Apéndice A

A. CONCEPTOS BÁSICOS TERMODINÁMICA

A.1. INTRODUCCIÓN

Empezaremos describiendo algunos conceptos básicos de termodinámica, la rama de la física que describe los estados de equilibrio a nivel macroscópicos. Ésta postula dos leyes principales: ([6] [7])

- **PRIMERA LEY**. Conocida también como principio de conservación de la energía. Si se realiza un trabajo sobre un sistema o éste intercambia calor con otro, la energía interna del sistema varía. Dicho de otra manera, el incremento de energía que se produce en un sistema es igual a la cantidad de calor que se aporta al mismo menos el trabajo que realiza. Expresado matemáticamente:

$$\Delta u = Q - W \tag{A.1}$$

- **SEGUNDA LEY**. Marca la dirección en la que deben llevarse a cabo los procesos termodinámicos y, por tanto, la imposibilidad de que ocurran en sentido contrario. Expresado para el caso que nos ocupa podemos decir que la transferencia de calor entre dos cuerpos siempre se va a producir desde el más caliente hacia el más frío hasta que se llegue al equilibrio térmico.

A.2. MAGNITUDES TERMODINÁMICAS

Las más importantes desde el punto de vista del ciclo de refrigeración por compresión de vapor son las siguientes:

- **Entalpía (H)**. Es una *función de estado* (el cambio de valor entre dos estados es independiente del proceso seguido para llegar de uno a otro) cuya variación expresa una medida de la cantidad de energía absorbida o cedida por un sistema termodinámico. Dicho de otra manera, es la cantidad de energía que un sistema intercambia con su entorno. En el sistema internacional se mide en Julios. ([6] [7]).

- **Entalpía por unidad de masa (h)**. Es el mismo concepto que el anterior pero referido a la unidad de masa. En este caso se mide en Julio/g en el sistema internacional. ([6] [7])

- **Entropía (S)**. Es una *función de estado* que nos permite determinar la parte de la energía que no puede utilizarse para producir un trabajo. (Julio/K). ([6] [7])

- **Entropía por unidad de masa (s)**. Mismo concepto que la entropía pero por unidad de masa. Se mide en Julio/g·K. ([6] [7])

- **Energía interna (U)**. Es la suma de la energía cinética y de la energía potencial interna de un sistema. Se mide en Julios en el sistema internacional. Al igual que en el caso de la entalpía y entropía, también puede definirse por unidad de masa (u), en cuyo caso sus unidades son Julios/g. ([6] [7])

- **Volumen específico (v)**. Es el volumen ocupado por una unidad de masa de un material. (inverso de la densidad). No depende de la cantidad de materia. ([6] [7])

Apéndice B

B. REPRESENTACIONES GRÁFICAS

B.1. ÍNDICE

VARIABLE	50%	90%	98%
Caudal ($\dot{m}$)	Figura B.1	Figura B.2	Figura B.3
Presión evaporador (P_1)	Figura B.4	Figura B.5	Figura B.6
Presión condensador (P_2)	Figura B.7	Figura B.8	Figura B.9
Potencia frigorífica ($\dot{Q}_e$)	Figura B.10	Figura B.11	Figura B.12
Potencia compresor $\dot{W}$	Figura B.13	Figura B.14	Figura B.15
Coefficient of Performance (COP)	Figura B.16	Figura B.17	Figura B.18
COP de Carnot	Figura B.19	Figura B.20	Figura B.21
Rendimiento de Carnot (η_{Carnot})	Figura B.22	Figura B.23	Figura B.24
Temperatura entrada compresor (T_1)	Figura B.25	Figura B.26	Figura B.27
Temperatura salida compresor (T_2)	Figura B.28	Figura B.29	Figura B.30
Temperatura entrada válvula expansión (T_3)	Figura B.31	Figura B.32	Figura B.33
Temperatura salida válvula expansión (T_4)	Figura B.34	Figura B.35	Figura B.36
Salto térmico evaporador (ΔT_e)	Figura B.37	Figura B.38	Figura B.39
Salto térmico condensador (ΔT_c)	Figura B.40	Figura B.41	Figura B.42
Sobrecalentamiento (ΔT_{sh})	Figura B.43	Figura B.44	Figura B.45
Subenfriamiento (ΔT_{sc})	Figura B.46	Figura B.47	Figura B.48
$f_1(T_3, \Delta T_c) = 5T_3 + 10\Delta T_c$	Figura B.49	Figura B.50	Figura B.51
$f_2(T_4, \Delta T_e) = 5T_4 + 10\Delta T_e$	Figura B.52	Figura B.53	Figura B.54

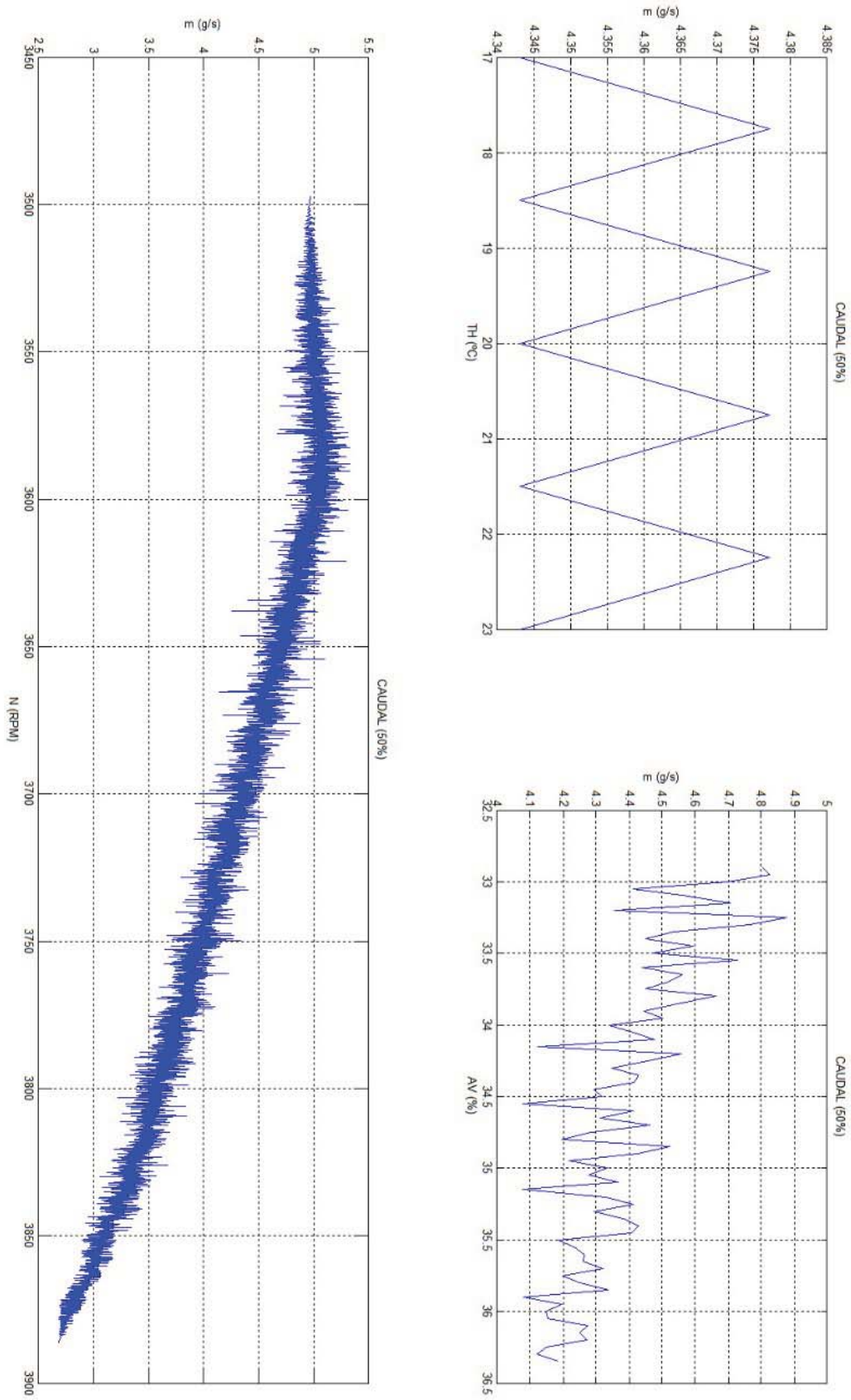


Figura B.1: Caudal m (50%)

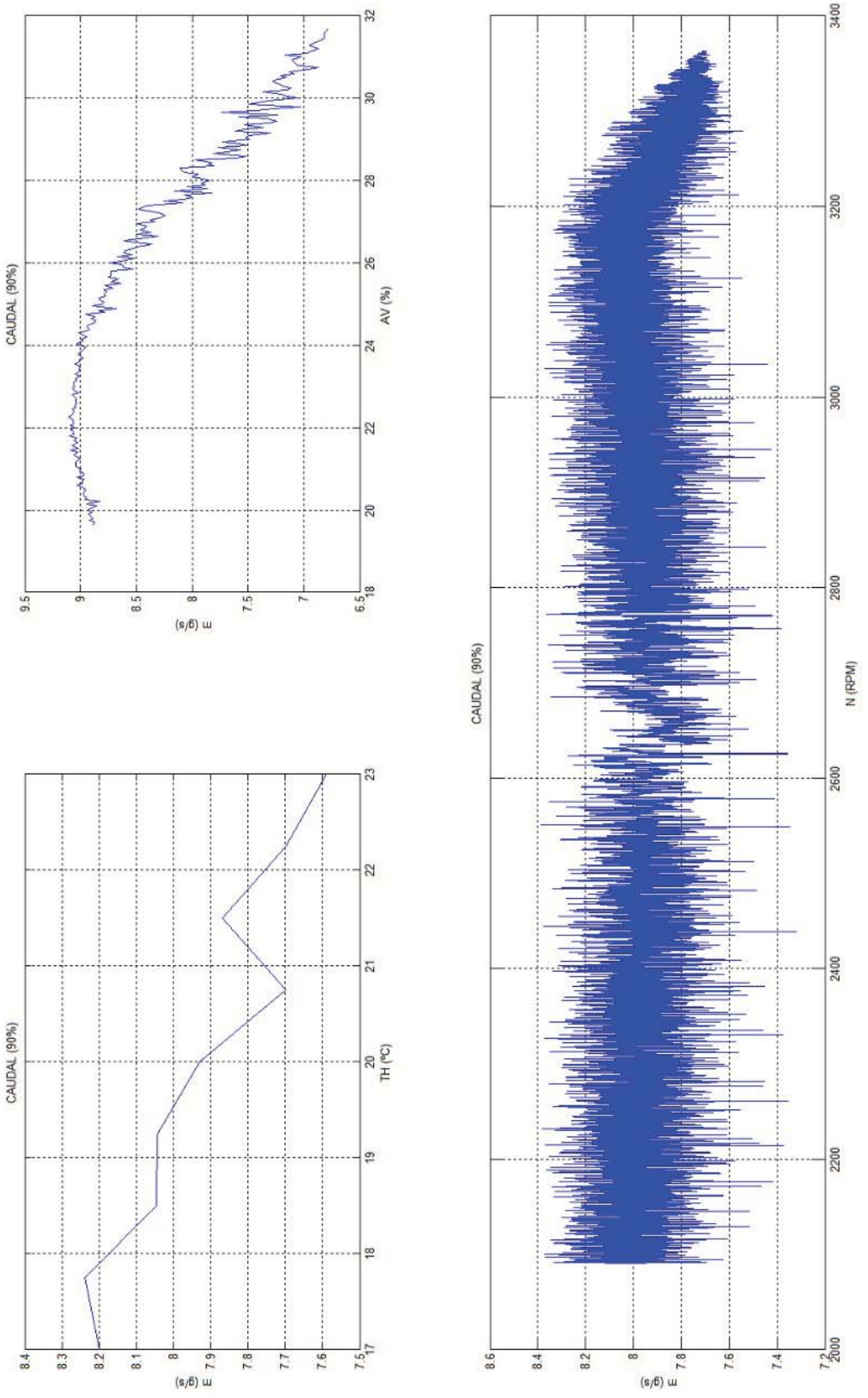


Figura B.2: Caudal m (90%)

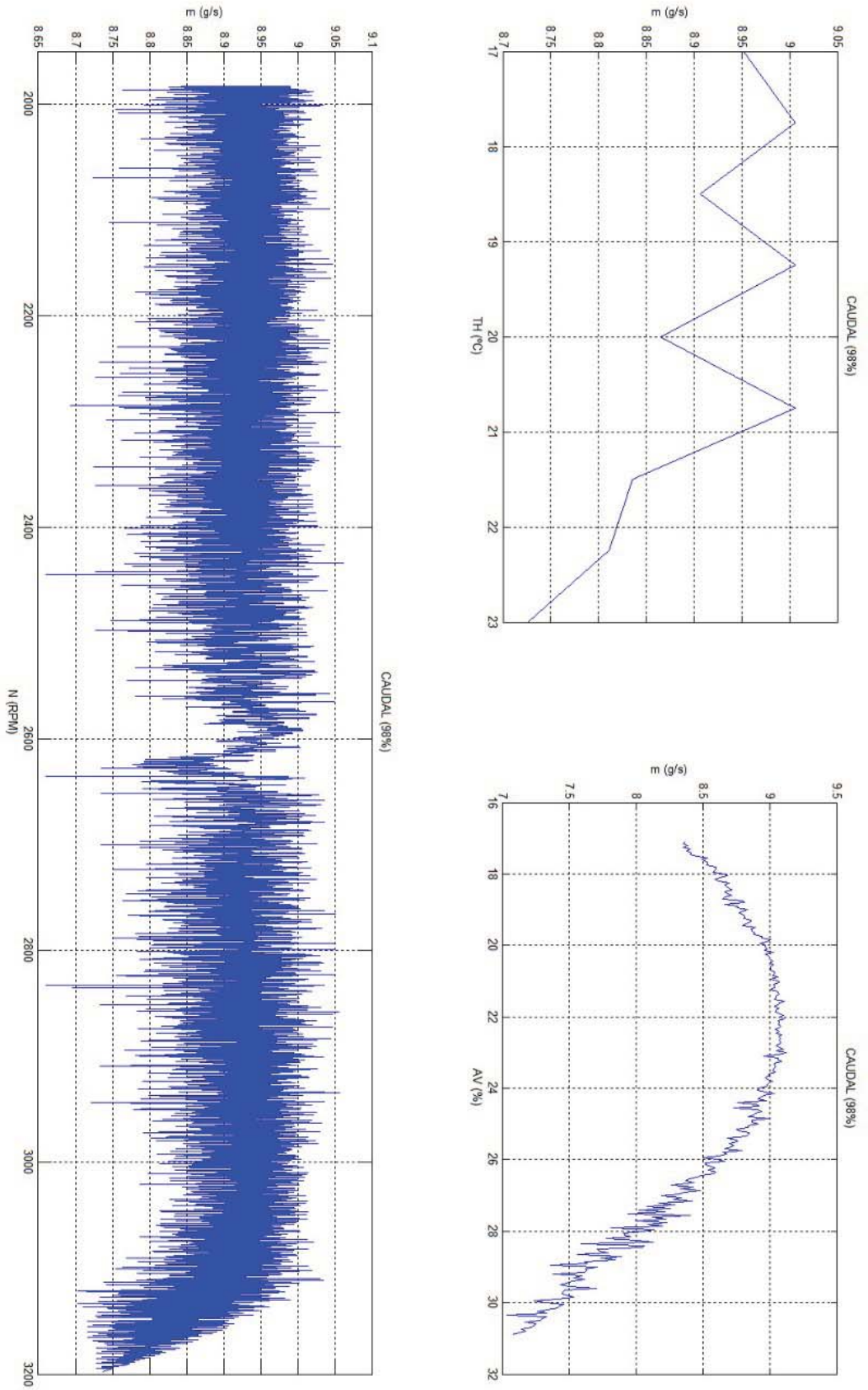


Figura B.3: Caudal m (98%)

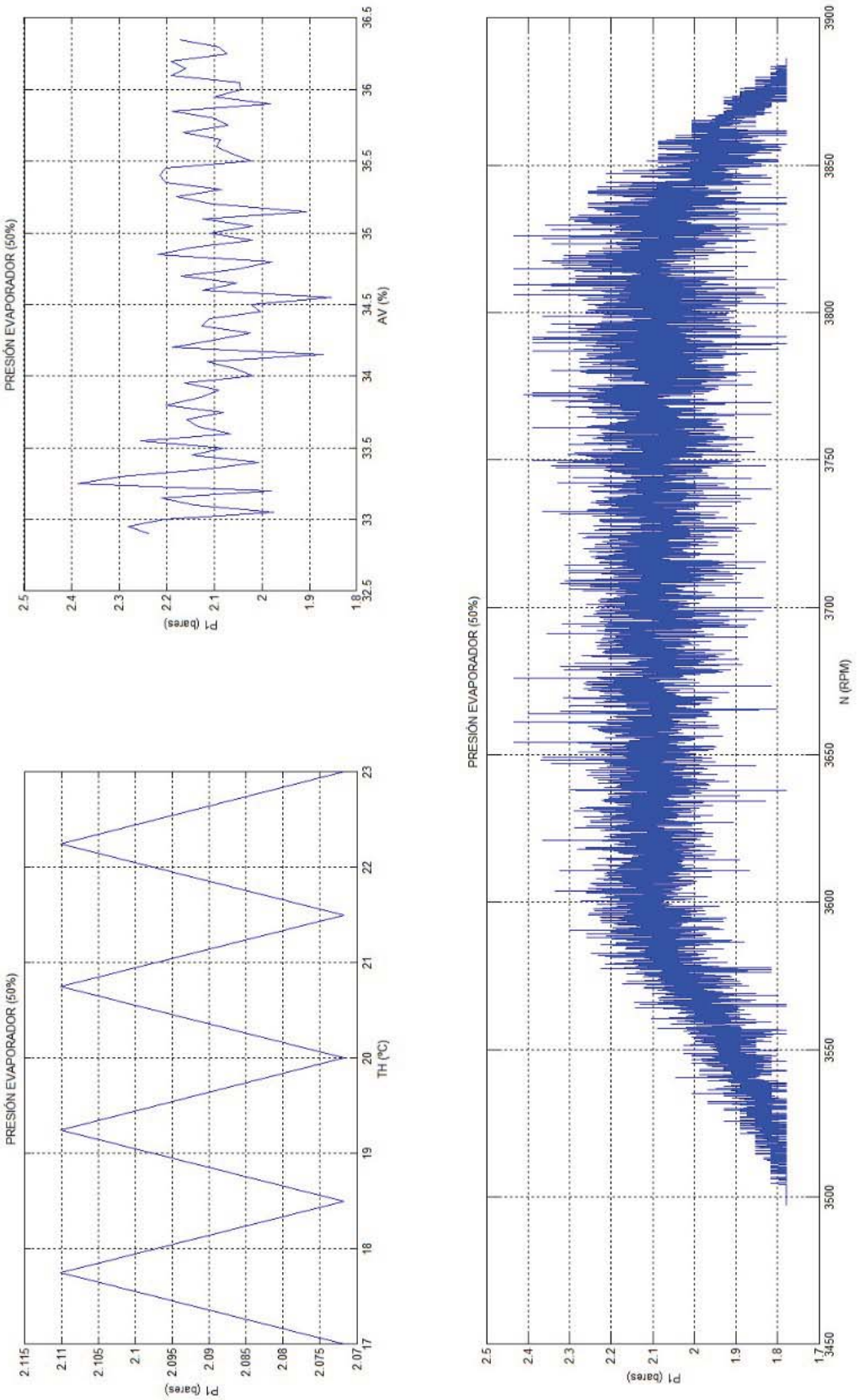


Figura B.4: Presión evaporador P_1 (50%)

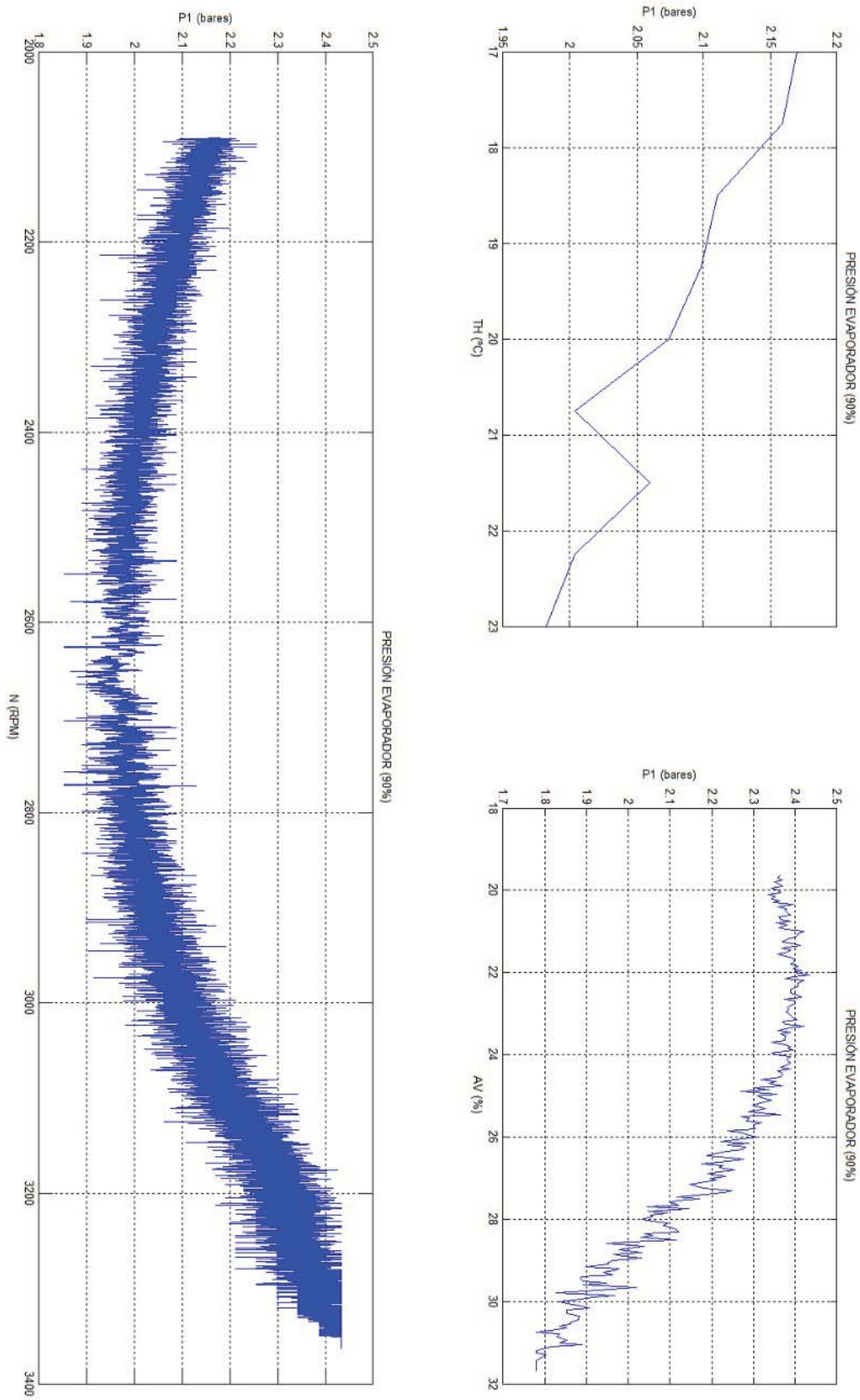


Figura B.5: Presión evaporador P_1 (90%)

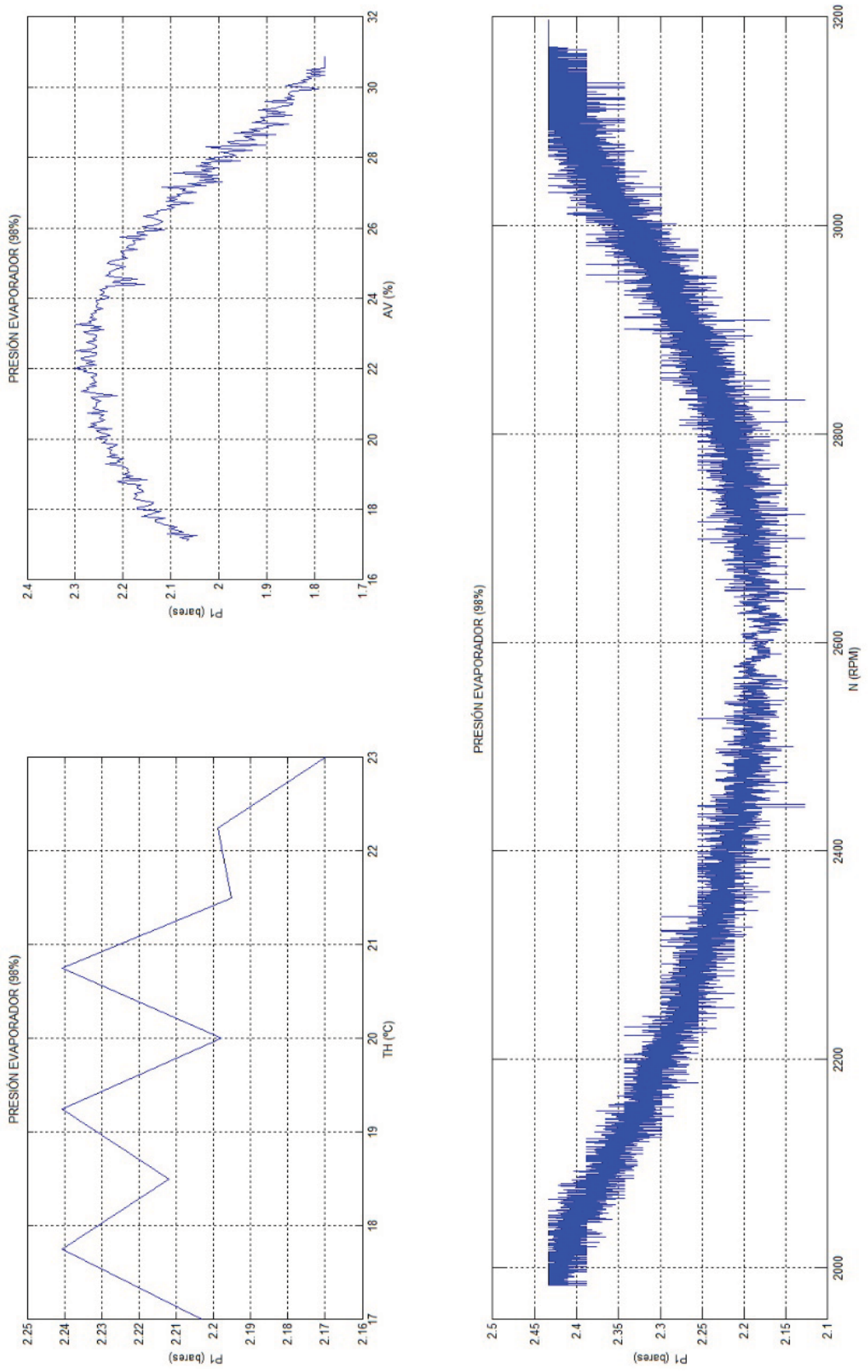


Figura B.6: Presión evaporador P_1 (98%)

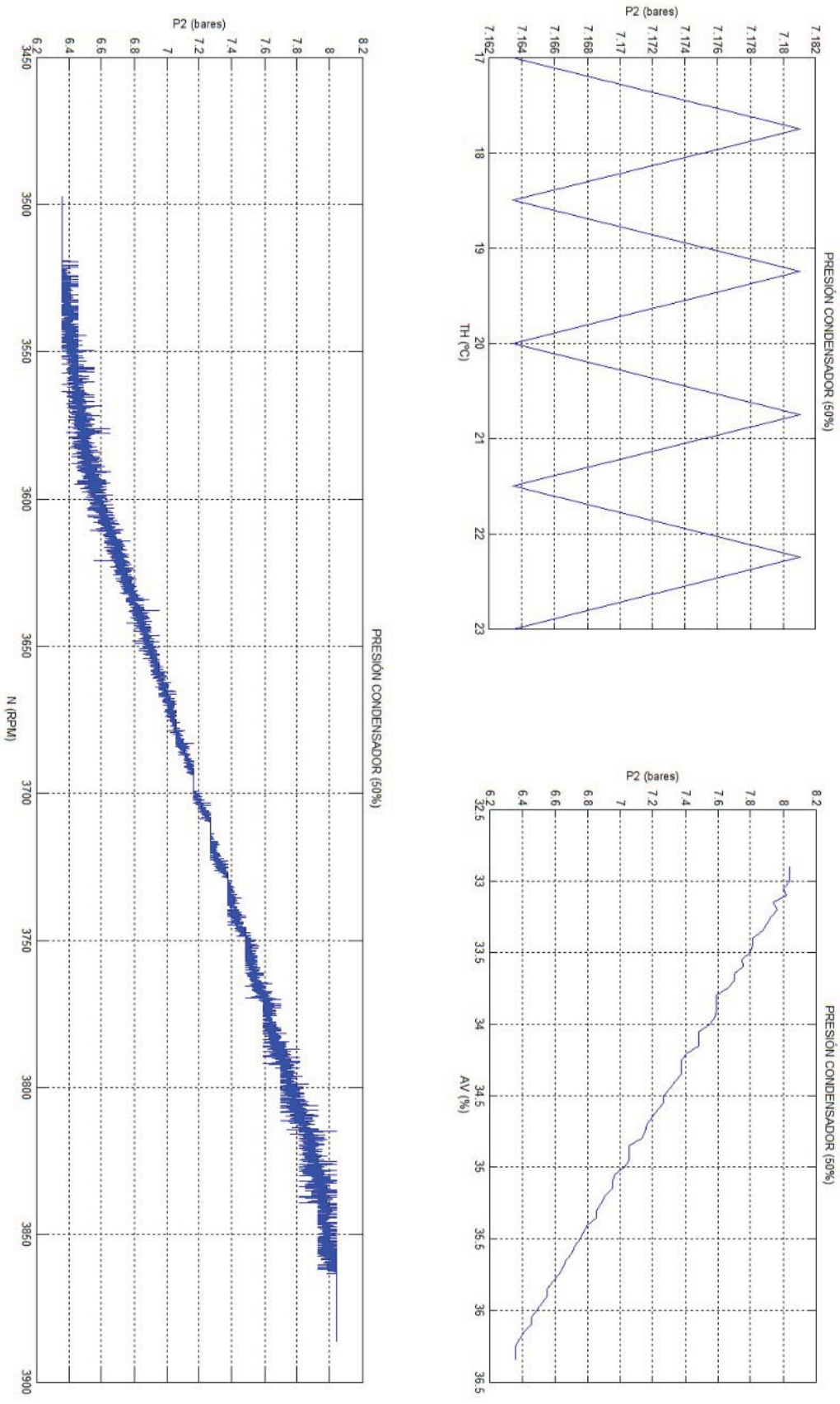


Figura B.7: Presión condensador P_2 (50%)

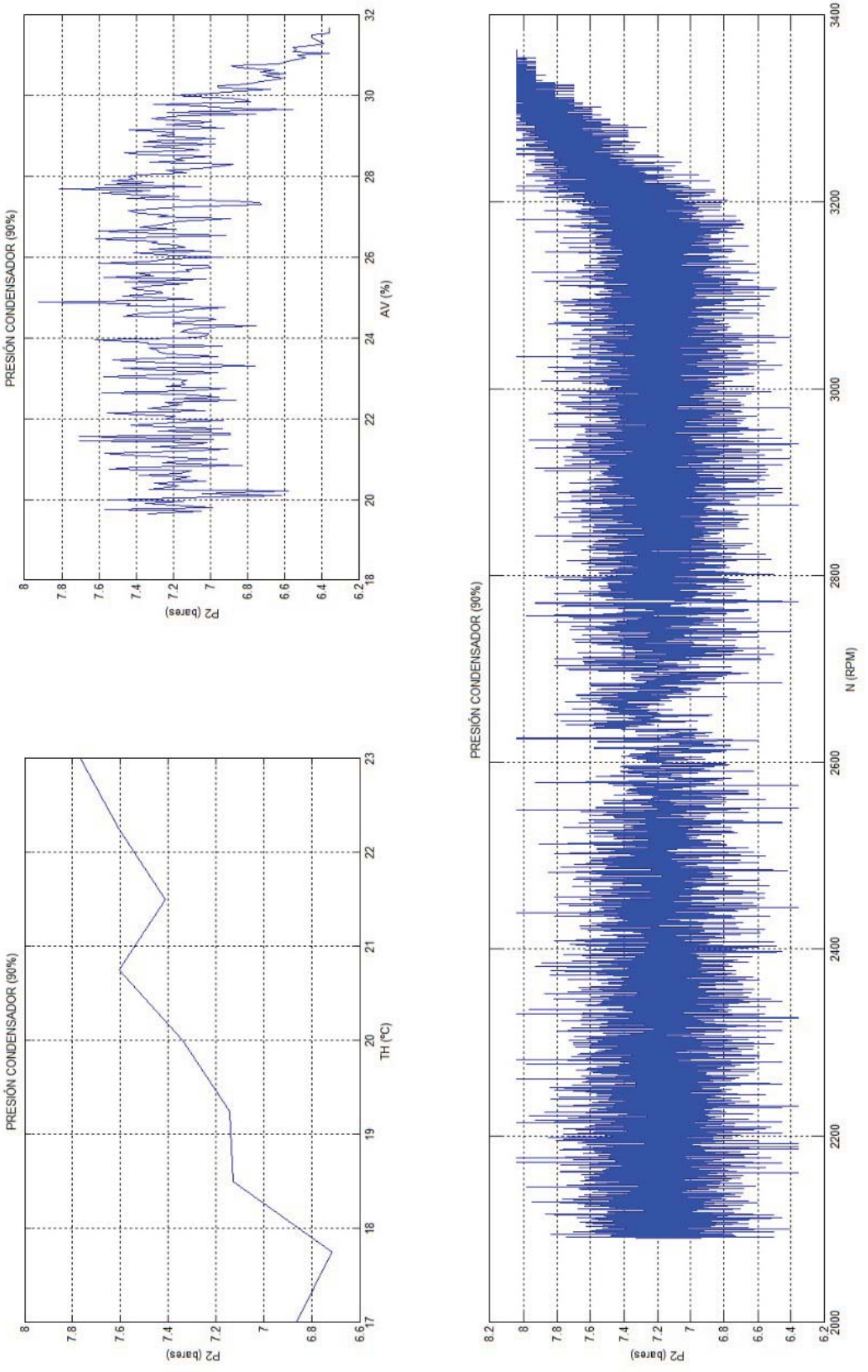


Figura B.8: Presión condensador P_2 (90%)

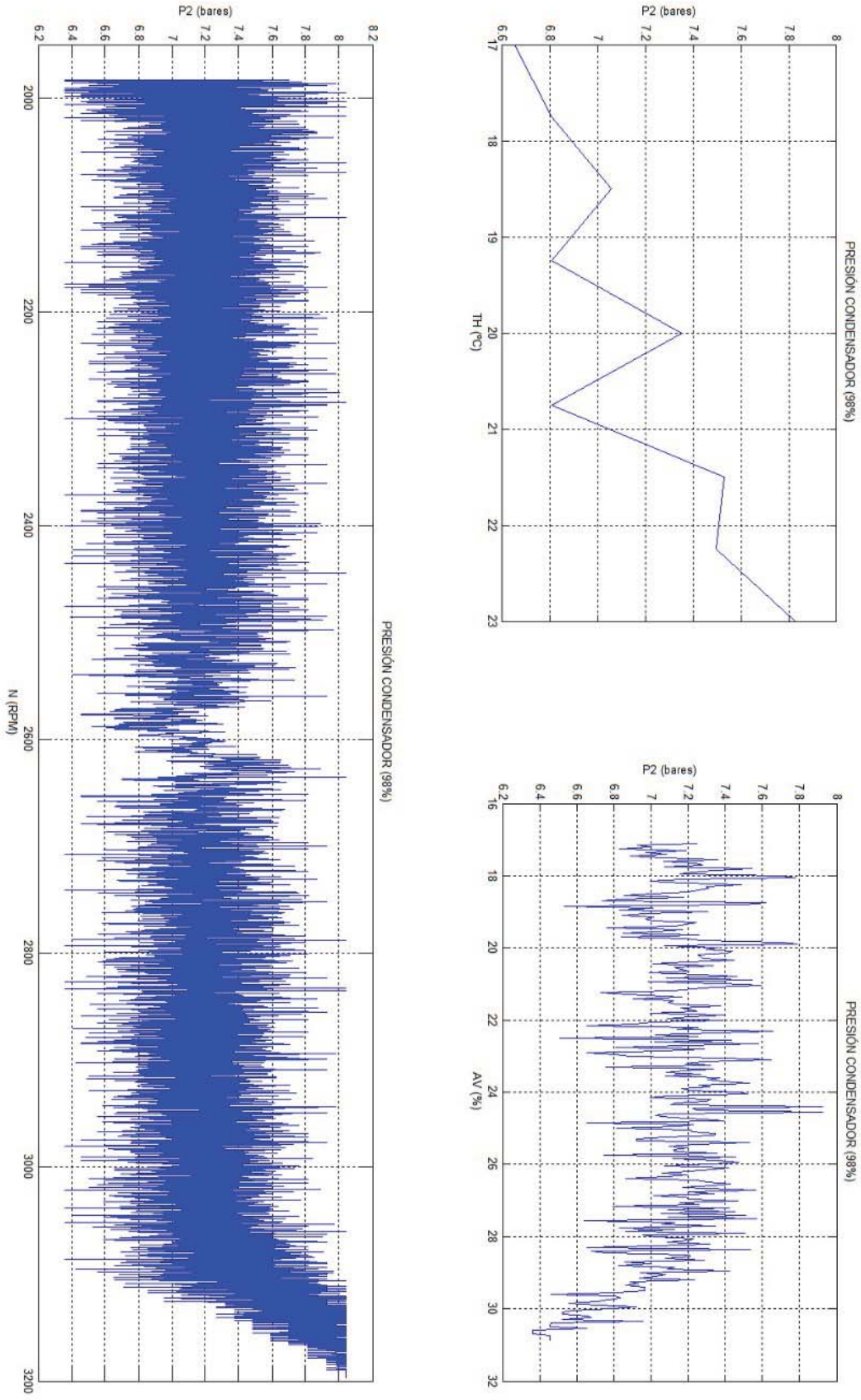


Figura B.9: Presión condensador P_2 (98%)

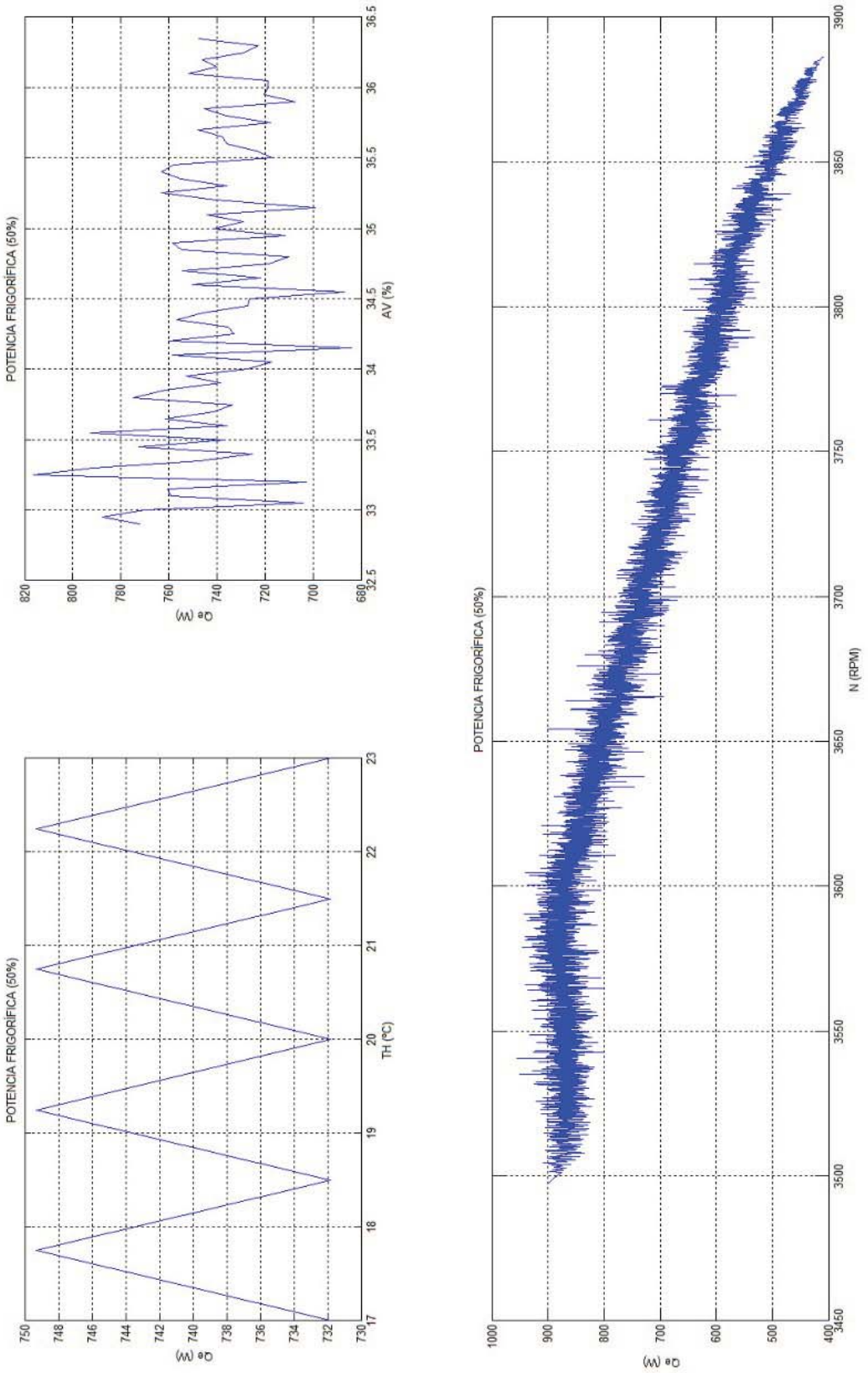


Figura B.10: Potencia frigorífica $\dot{Q}_e$ (50%)

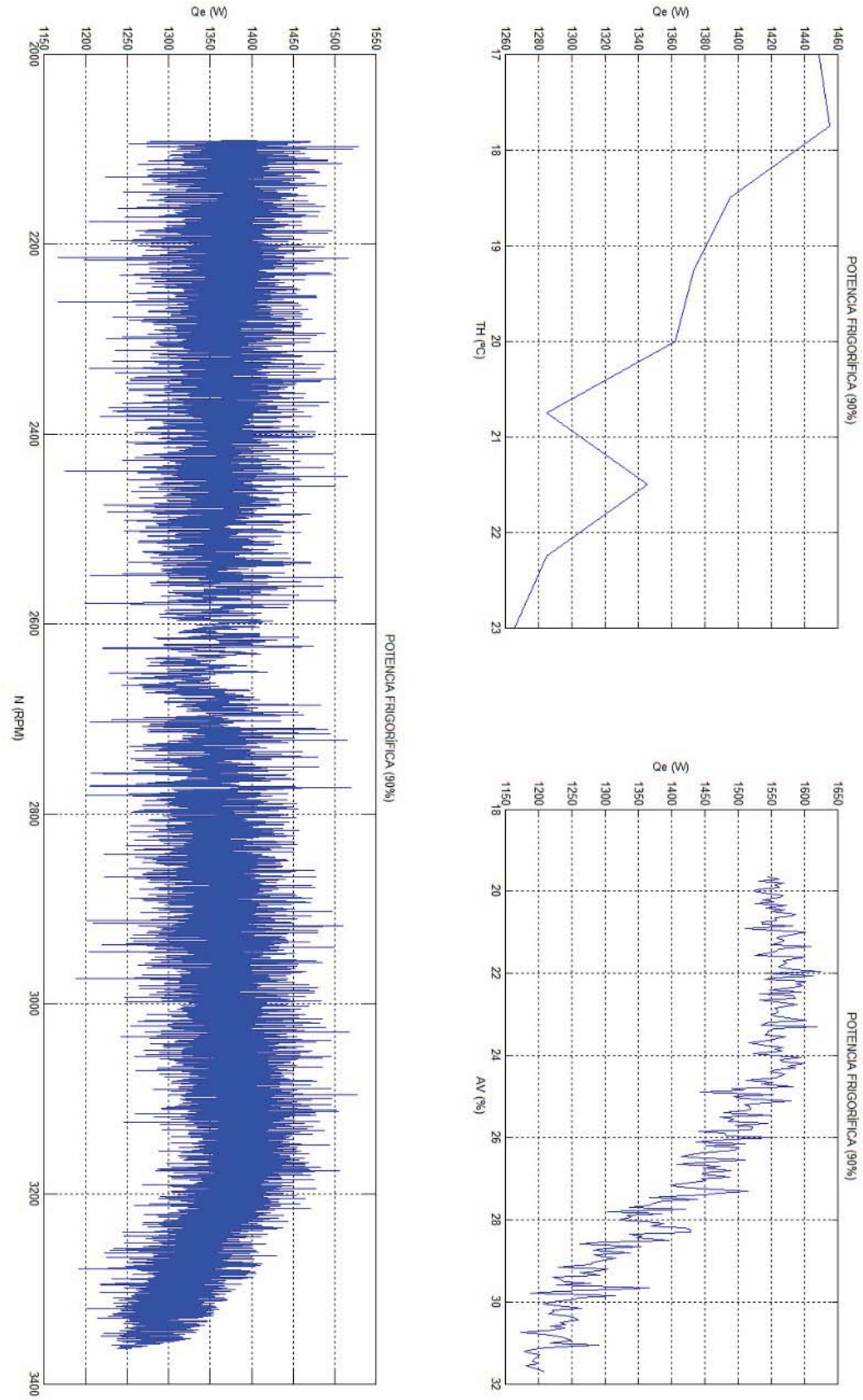


Figura B.11: Potencia frigorífica $\dot{Q}_e$ (90%)

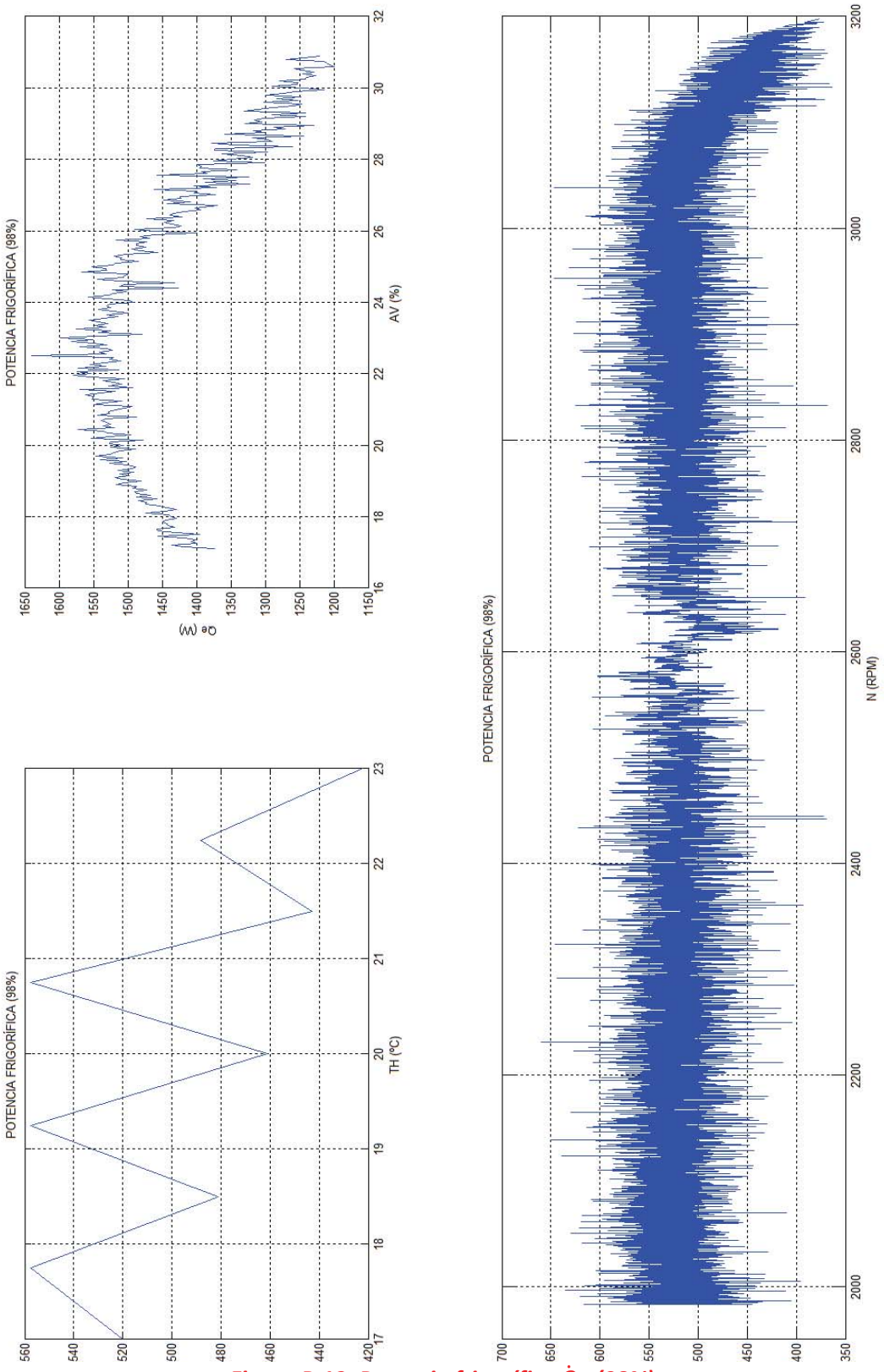


Figura B.12: Potencia frigorífica $\dot{Q}_e$ (98%)

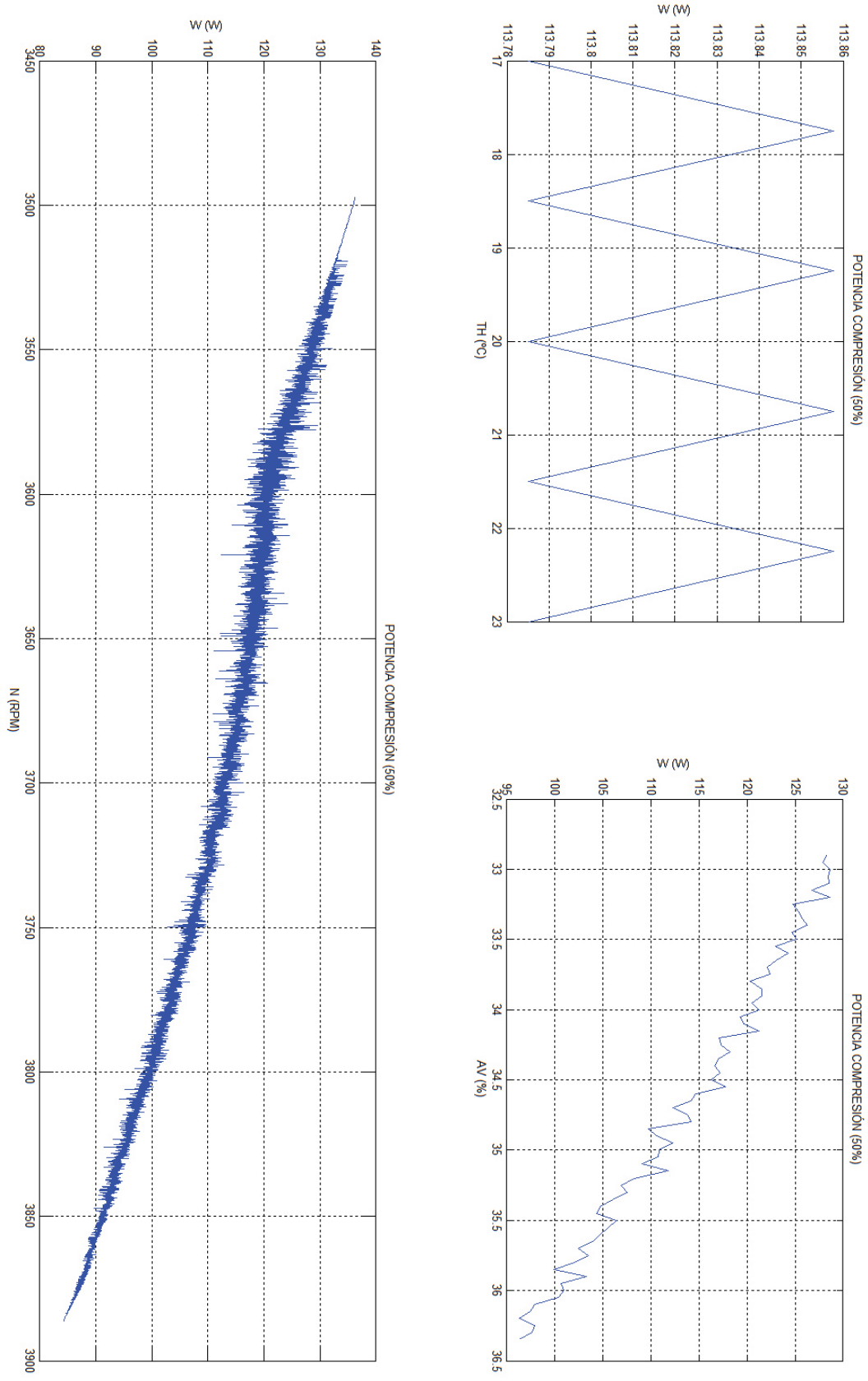


Figura B.13: Potencia compresión $\dot{W}$ (50%)

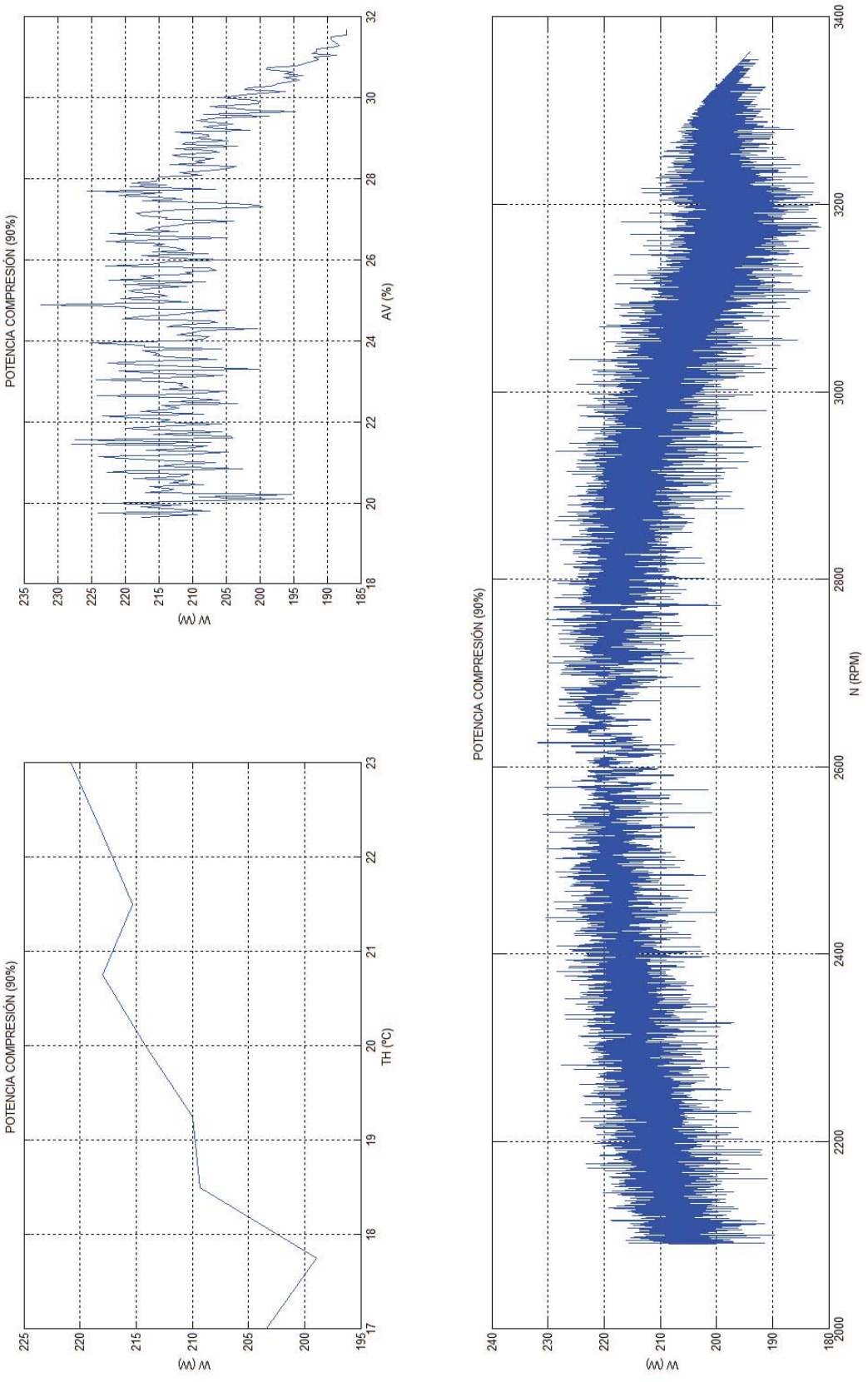


Figura B.14: Potencia compresión Ẇ (90%)

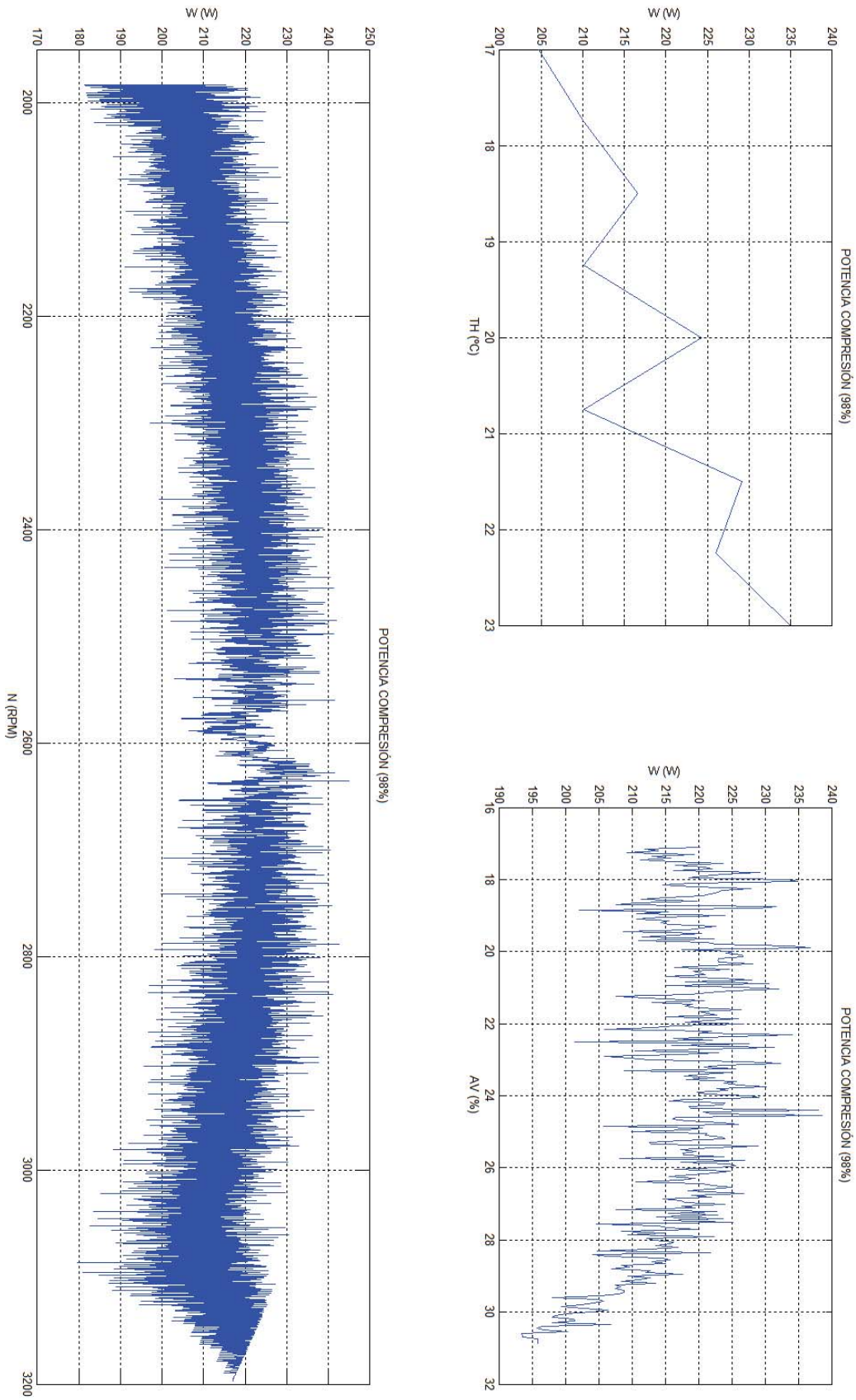


Figura B.15: Potencia compresión $\dot{W}$ (98%)

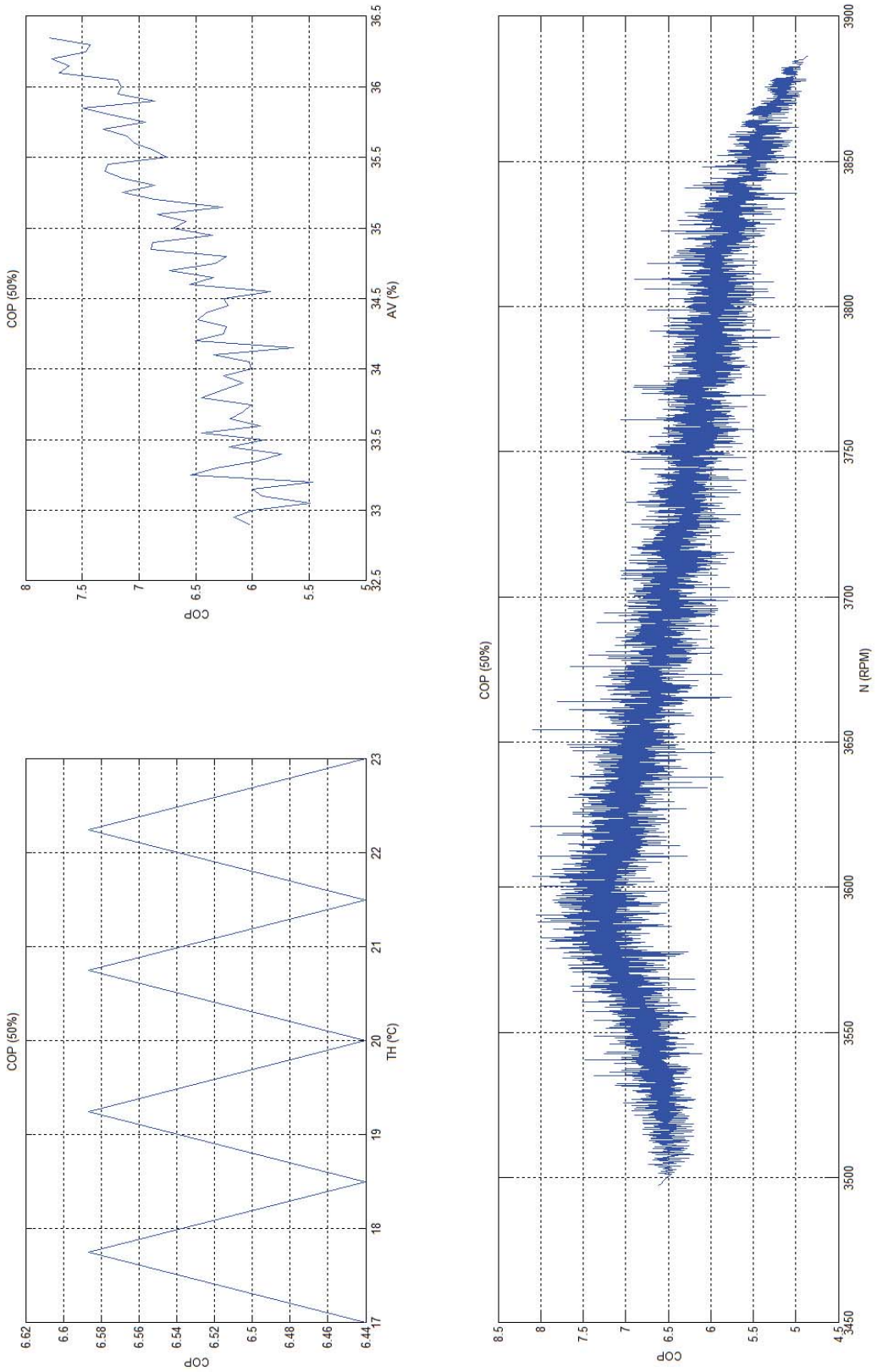


Figura B.16: COP (50%)

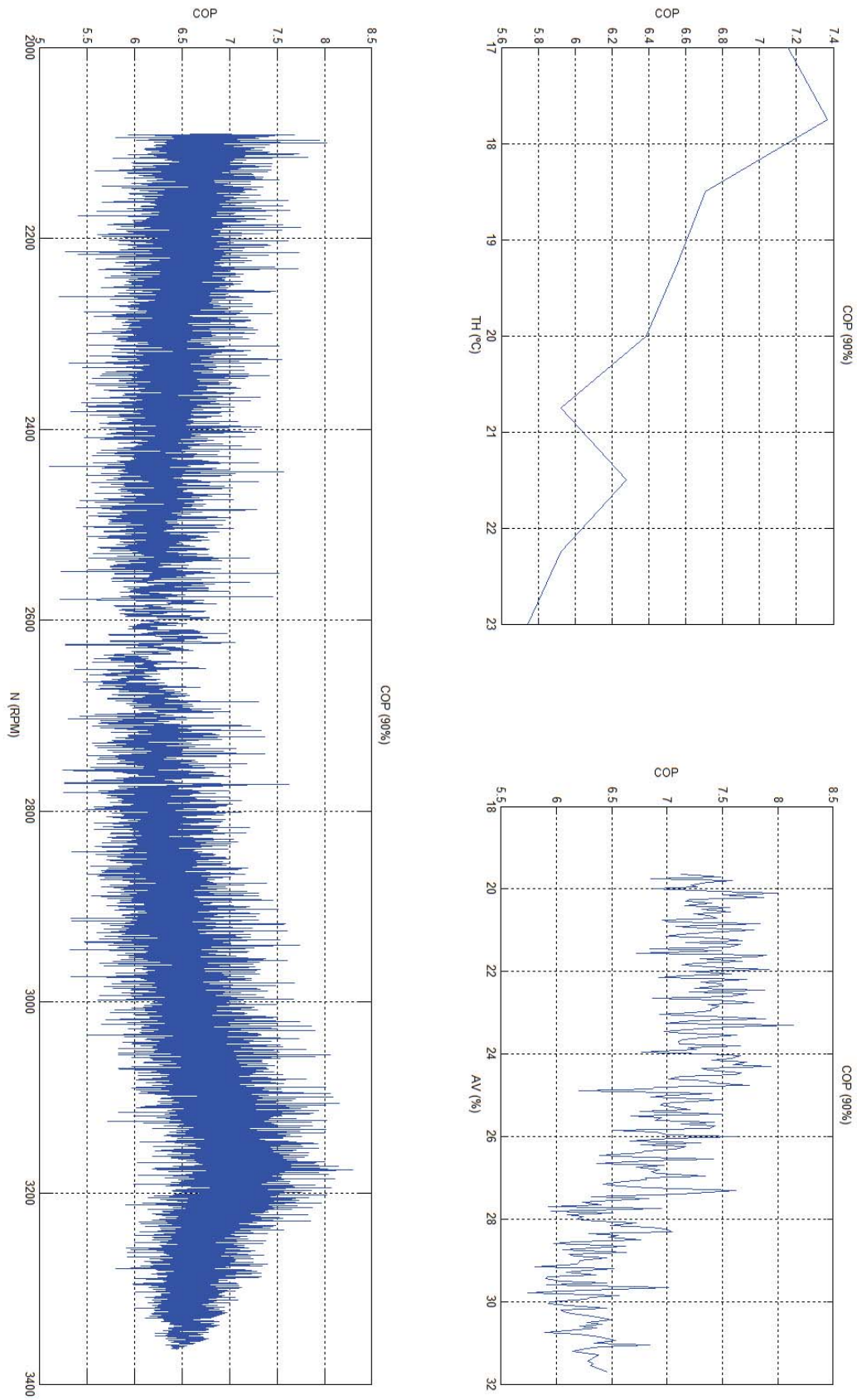


Figura B.17: COP (90%)

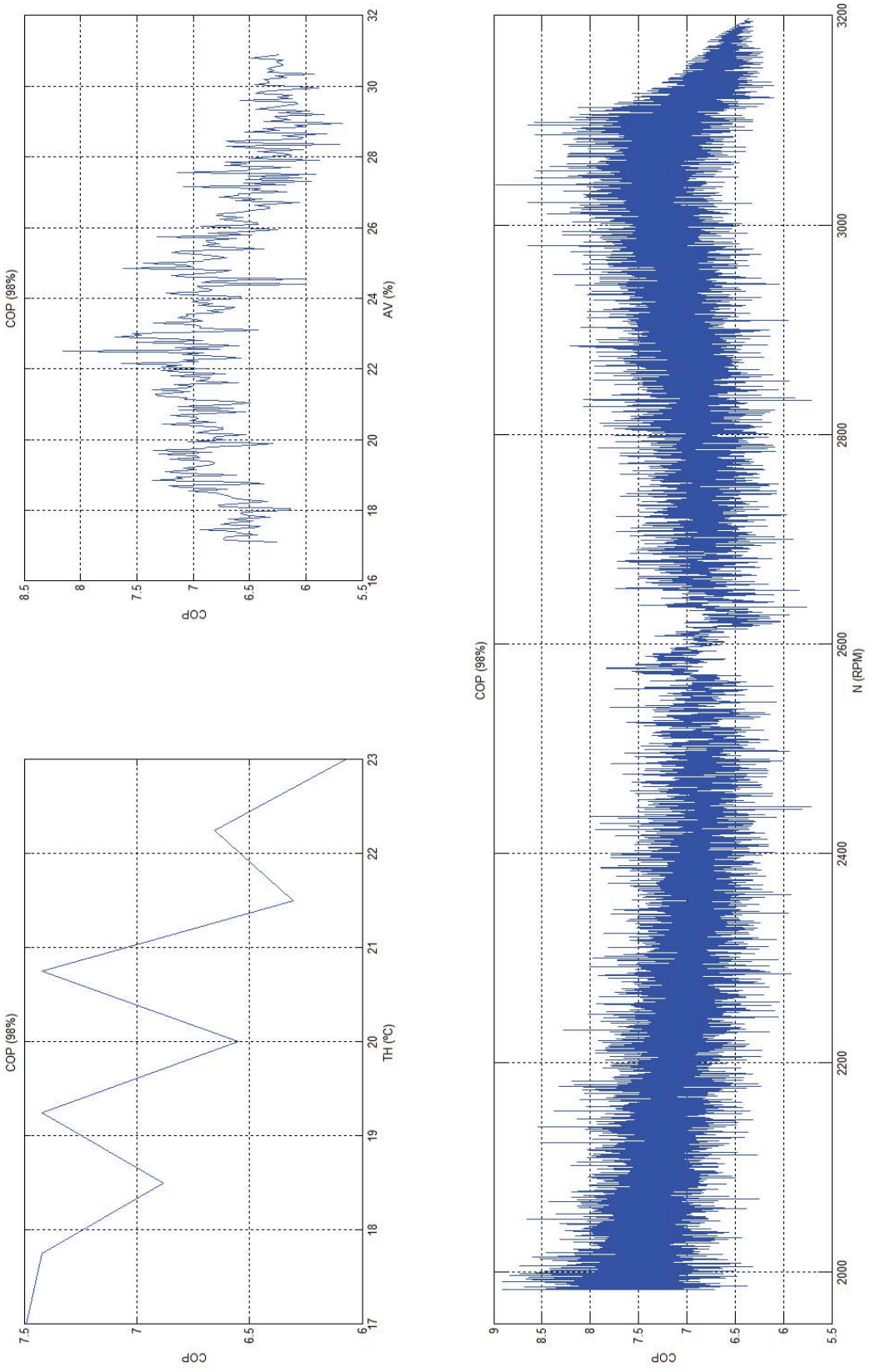


Figura B.18: COP (98%)

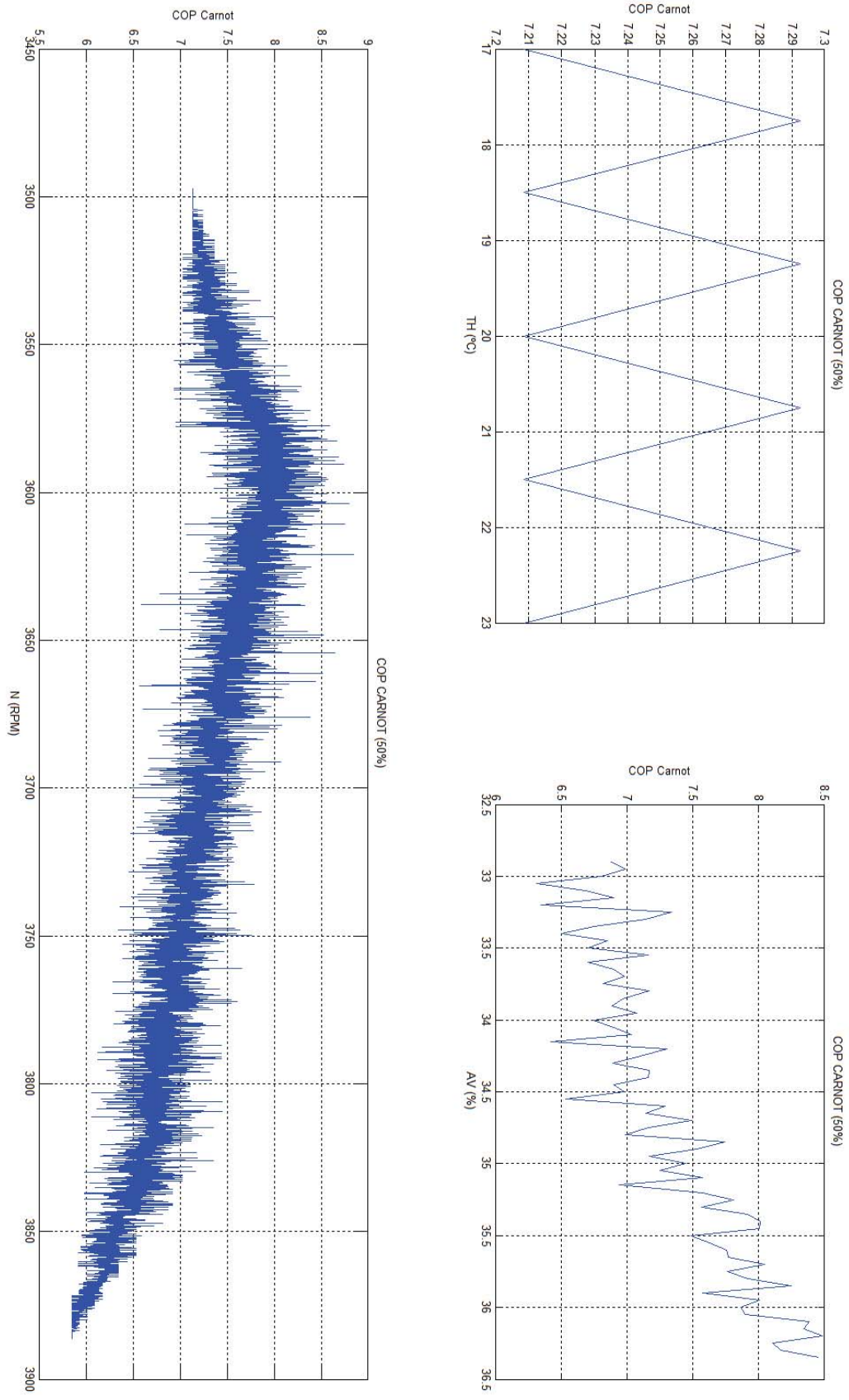


Figura B.19: COP Carnot (50%)

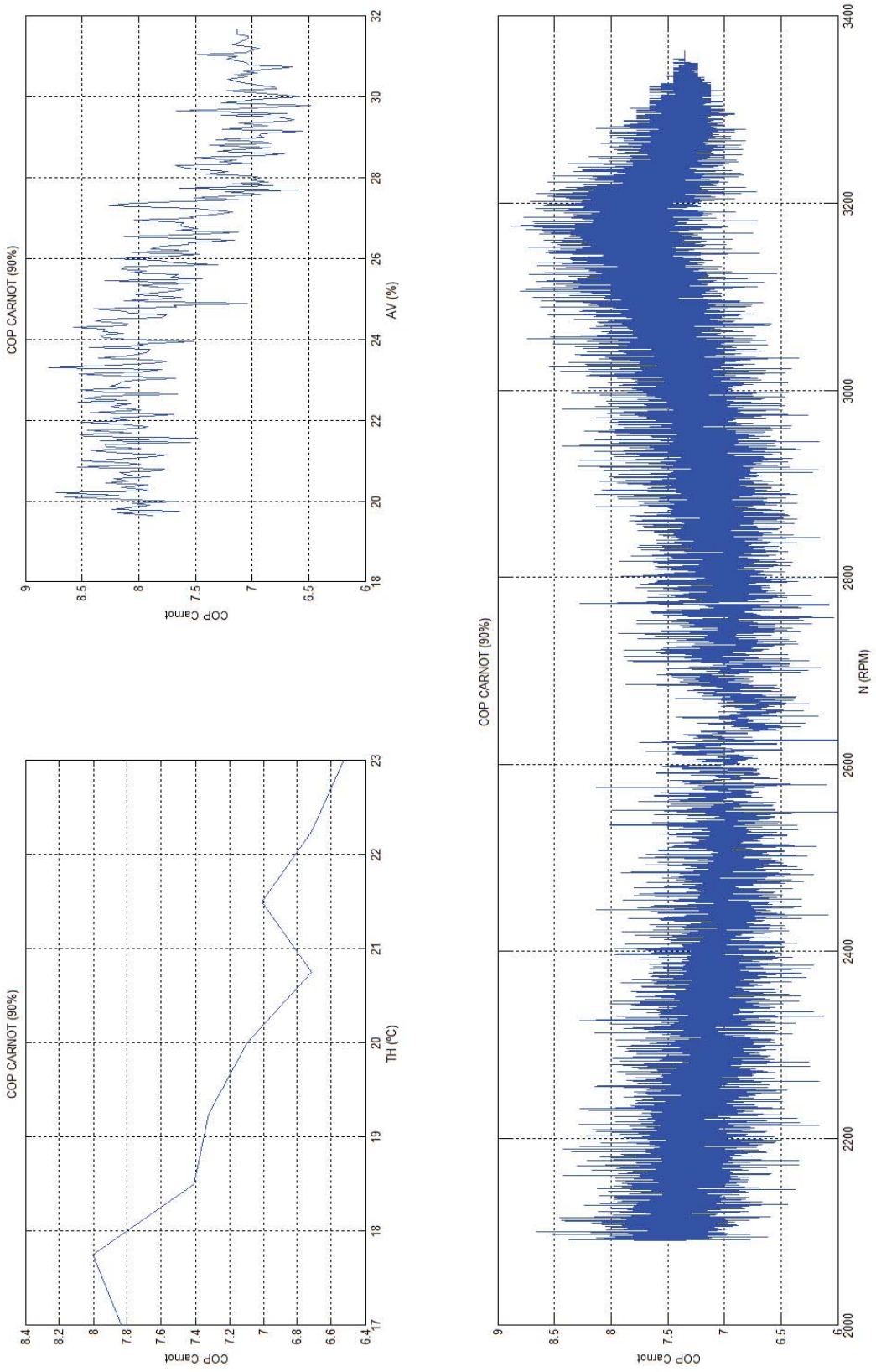


Figura B.20: COP Carnot (90%)

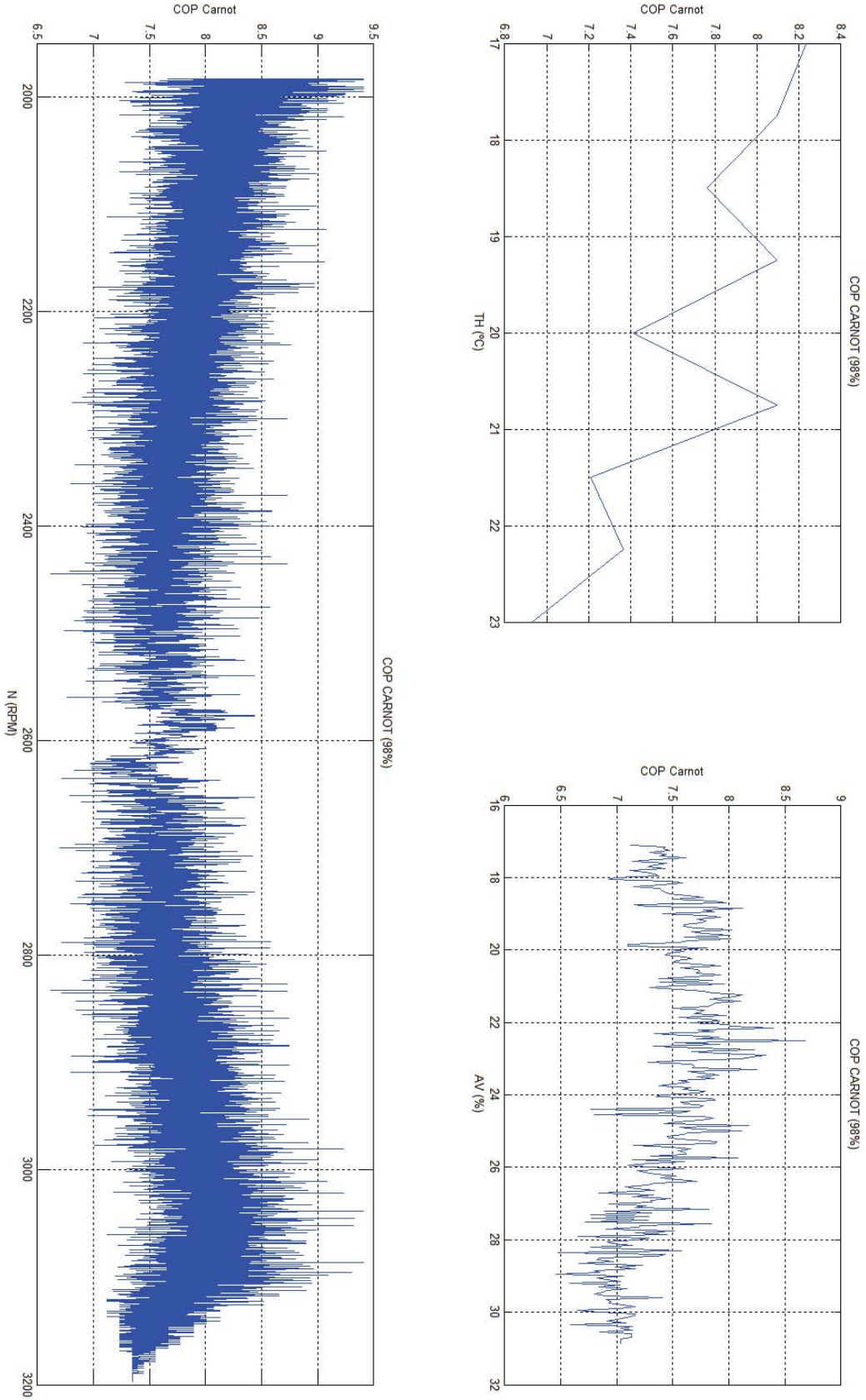


Figura B.21: COP Carnot (98%)

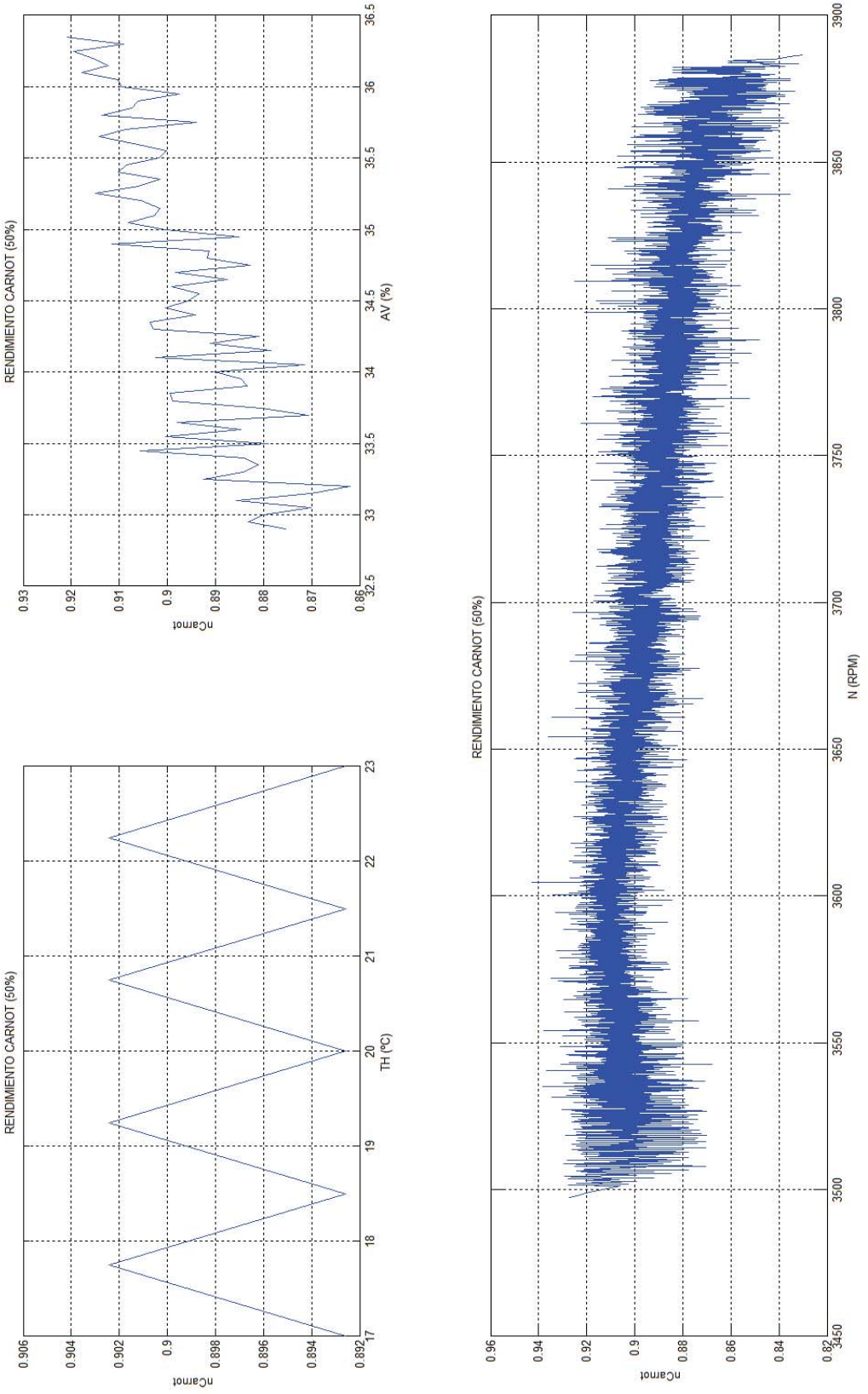


Figura B.22: Rendimiento de Carnot η (50%)

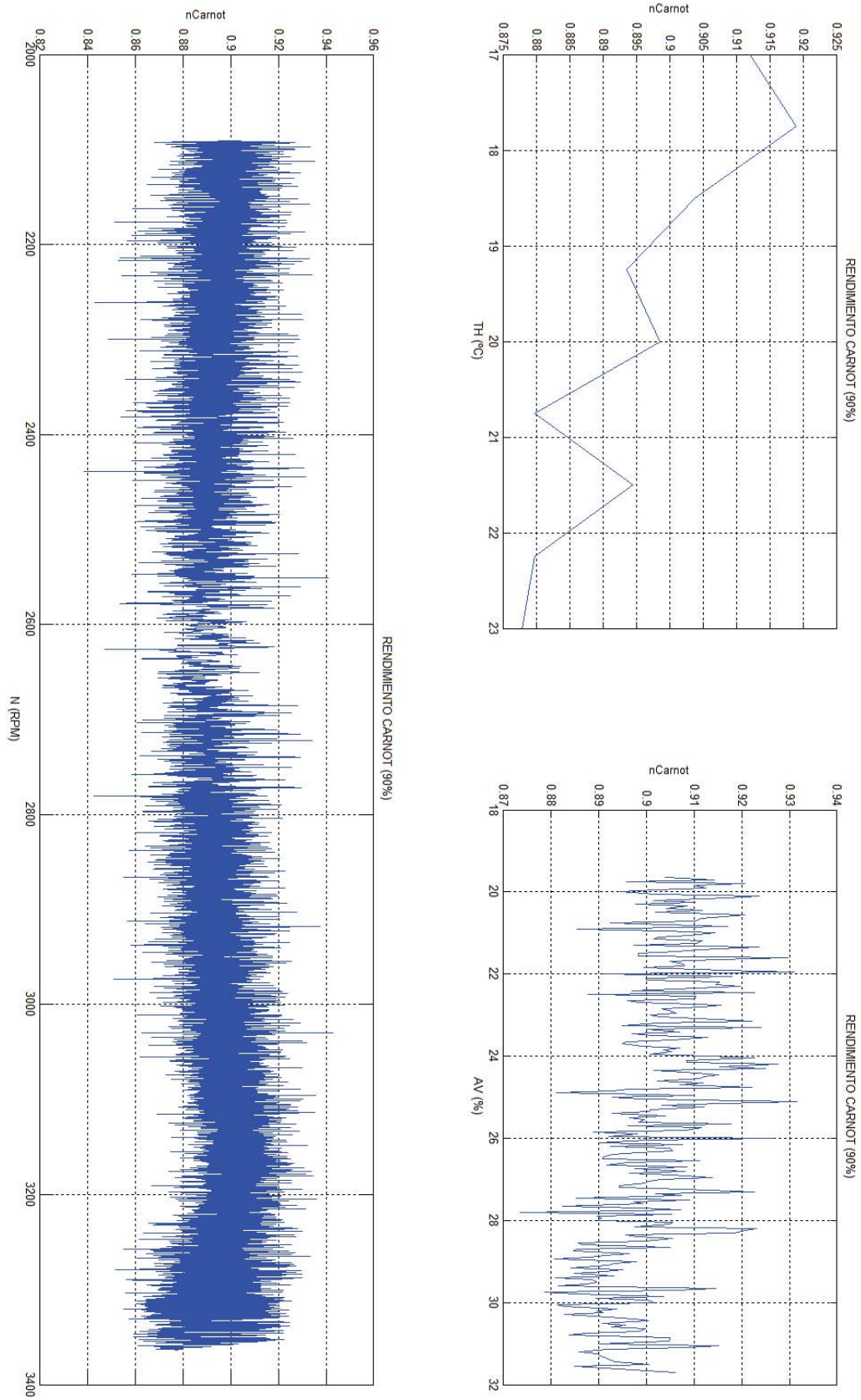


Figura B.23: Rendimiento de Carnot η (90%)

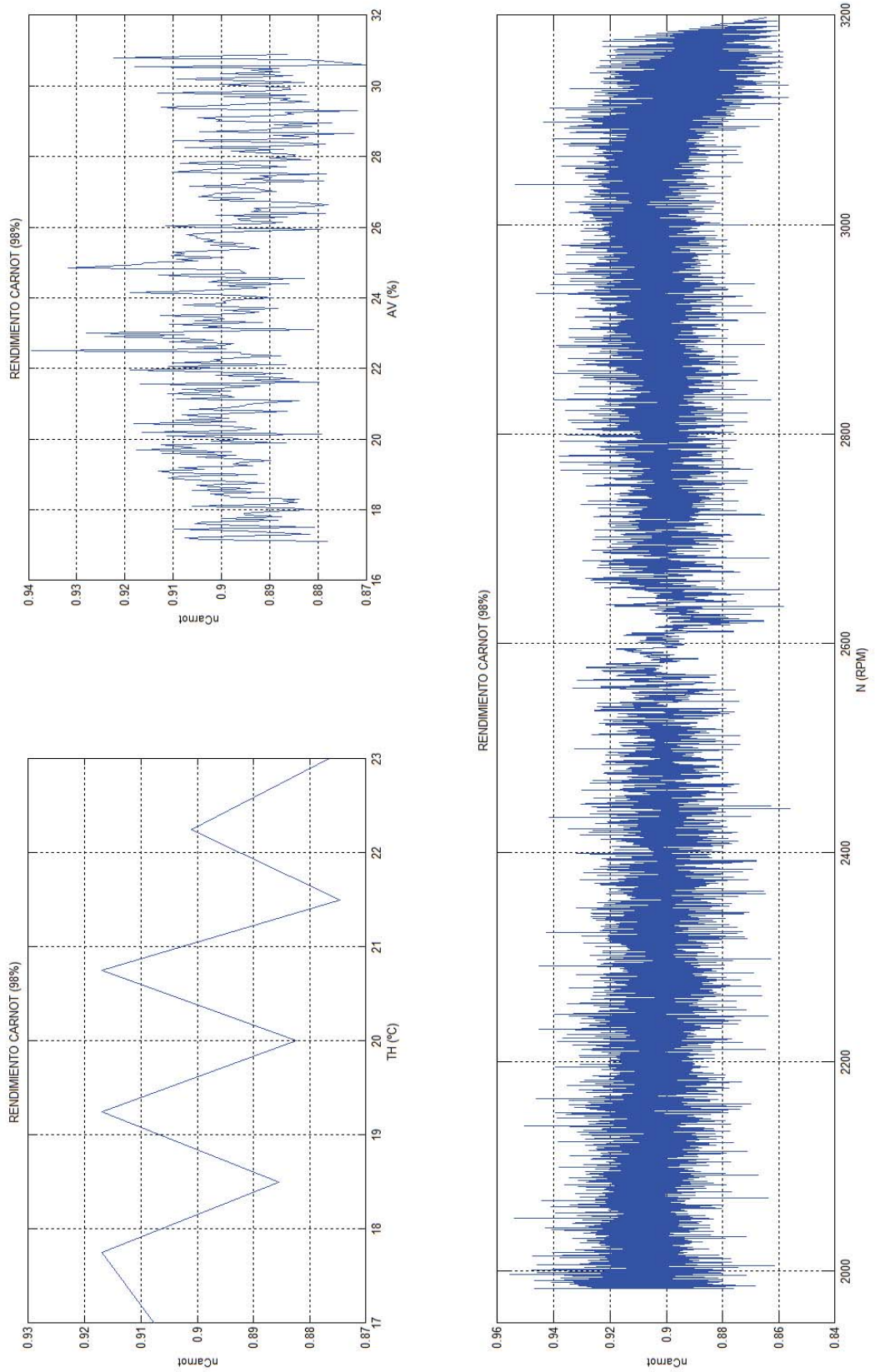


Figura B.24: Rendimiento de Carnot η (98%)

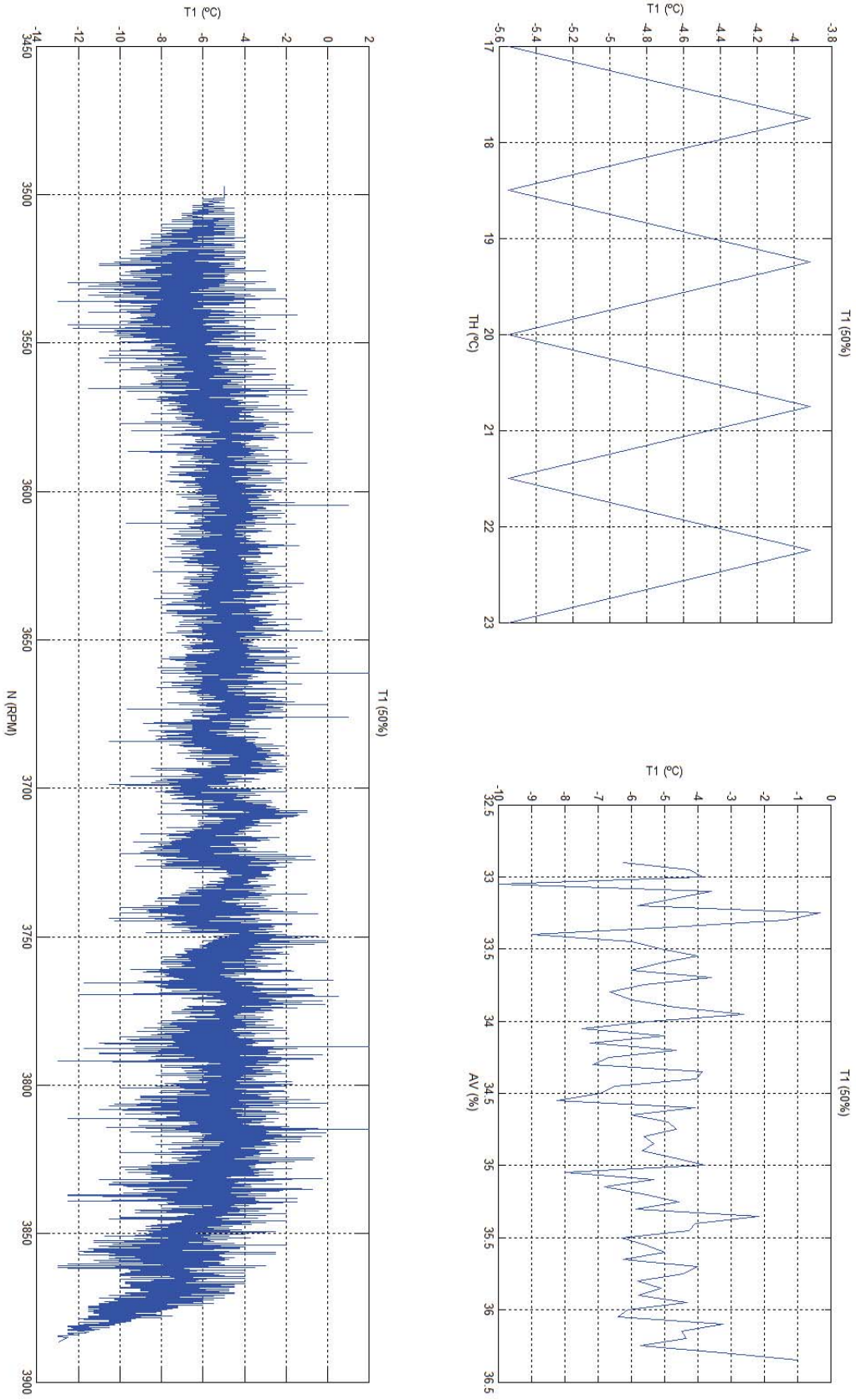


Figura B.25: Temperatura entrada compresor T_1 (50%)

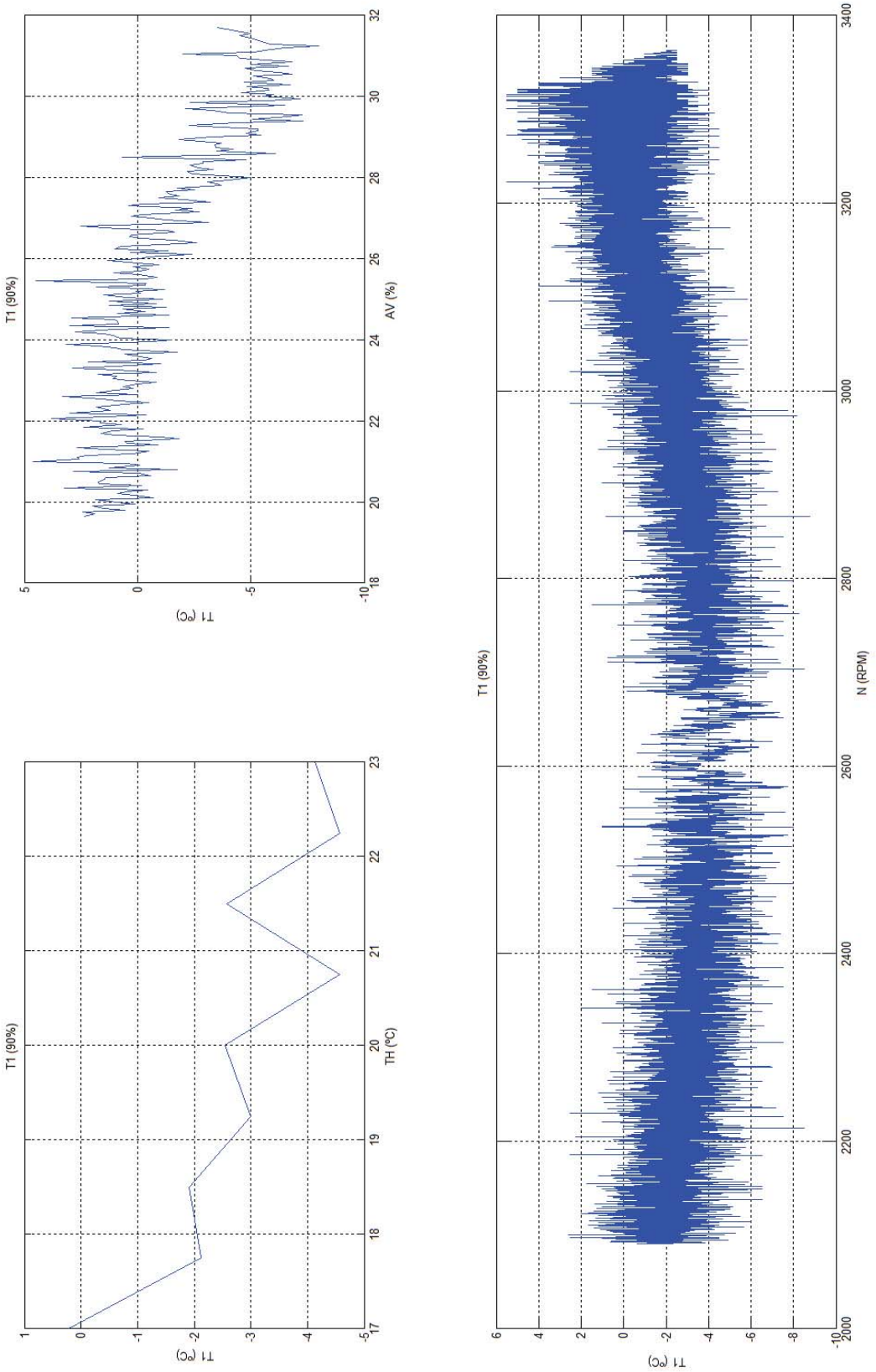


Figura B.26: Temperatura entrada compresor T_1 (90%)

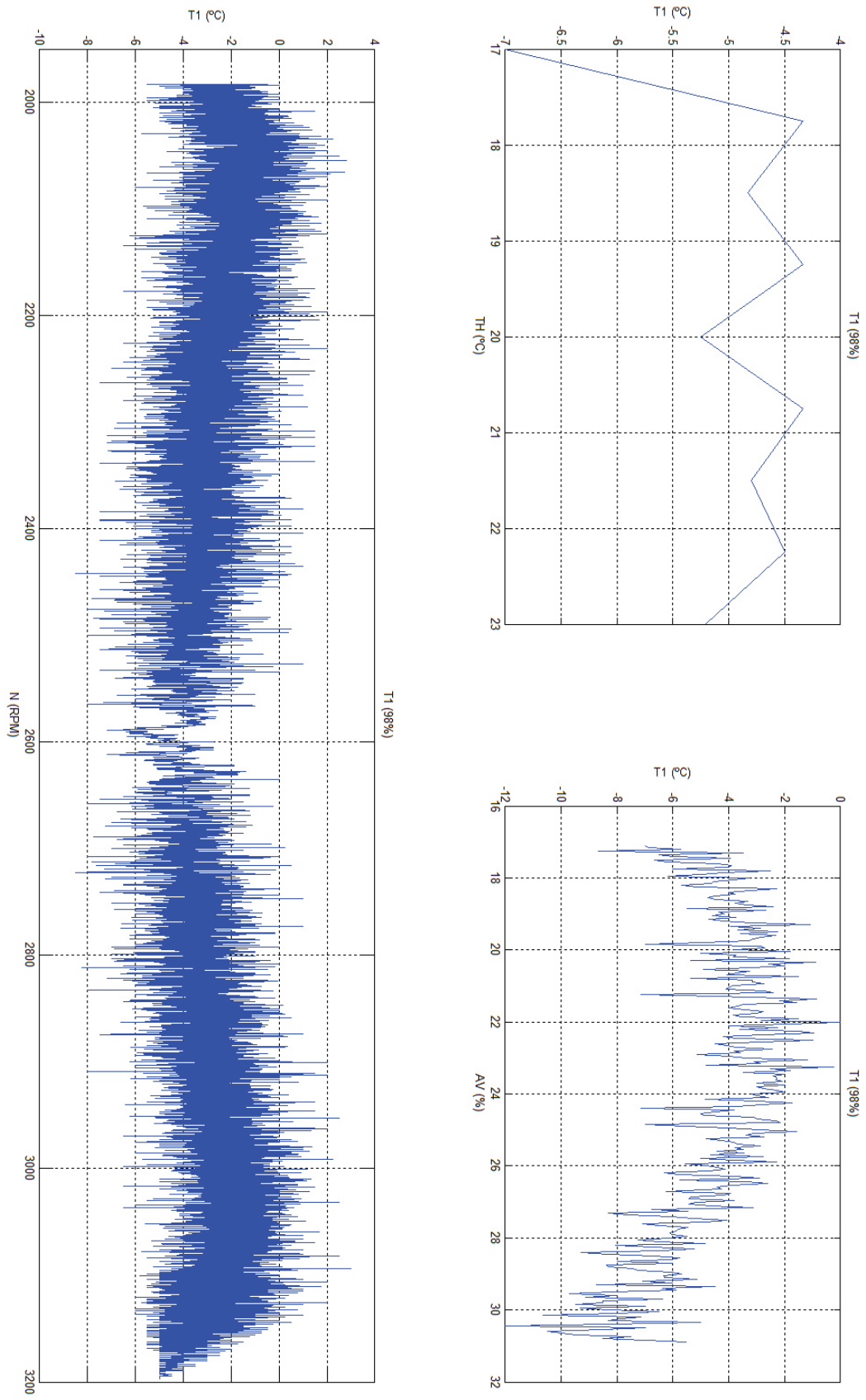


Figura B.27: Temperatura entrada compresor T_1 (98%)

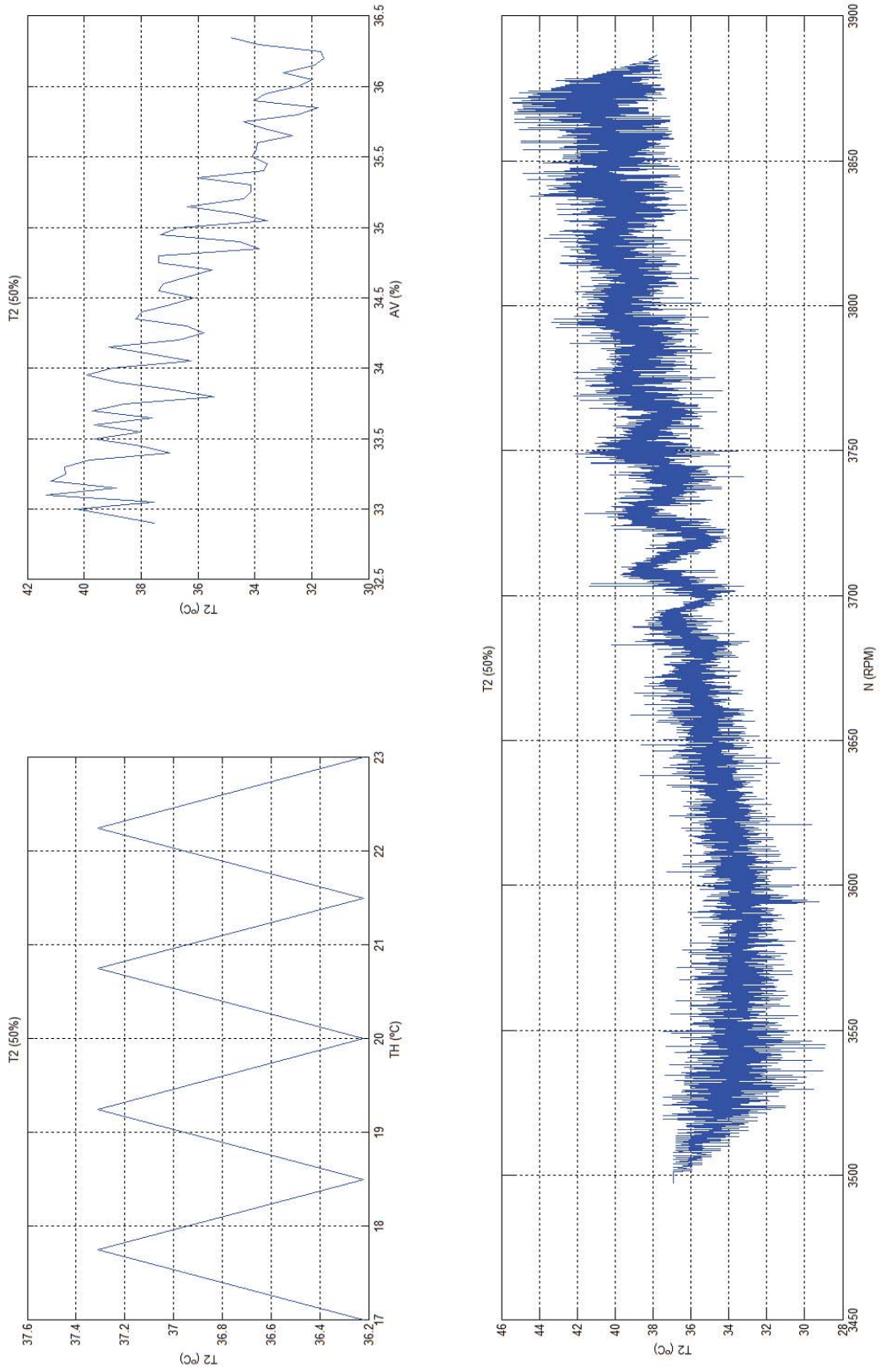


Figura B.28: Temperatura salida compresor T_2 (50%)

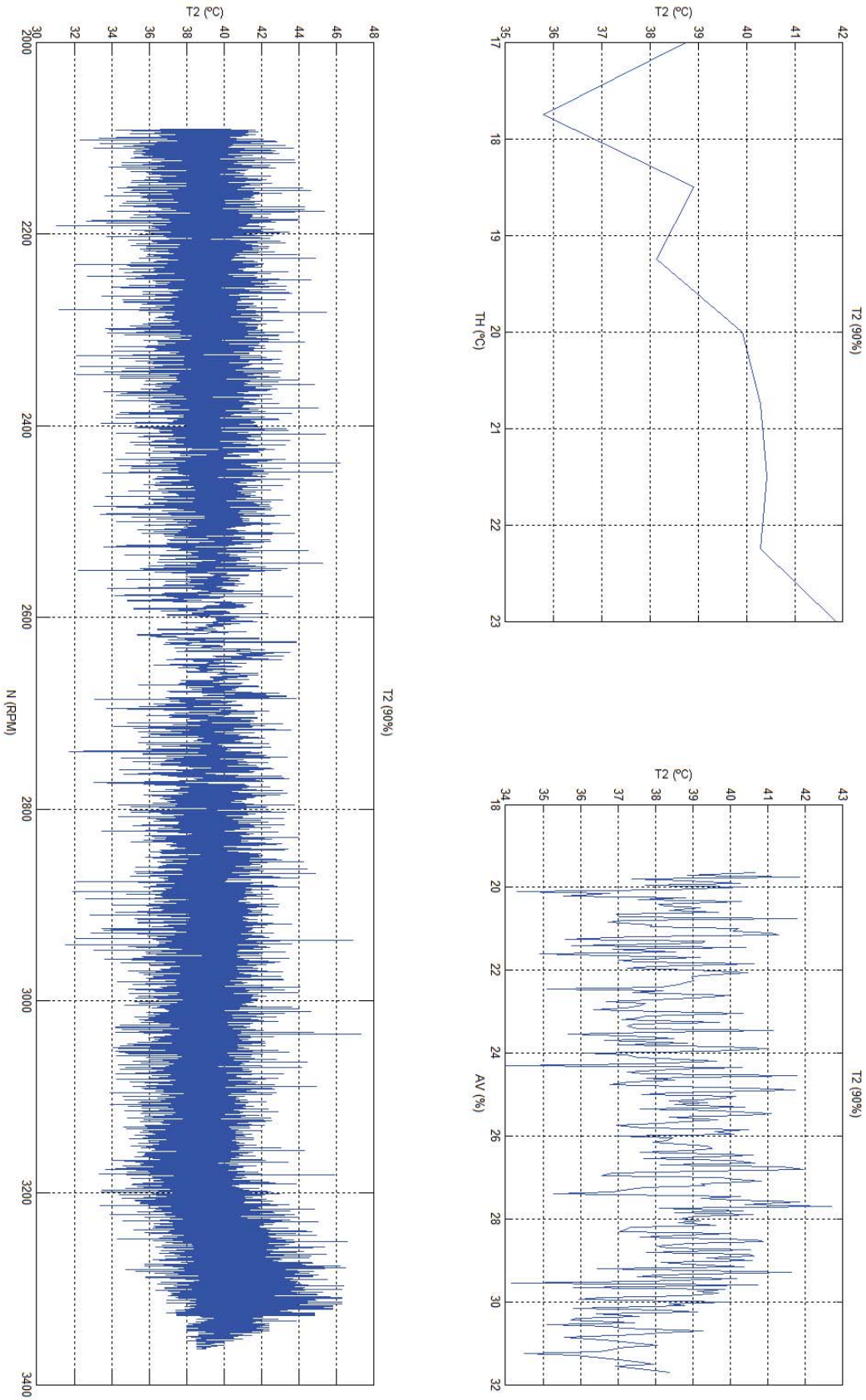


Figura B.29: Temperatura salida compresor T_2 (90%)

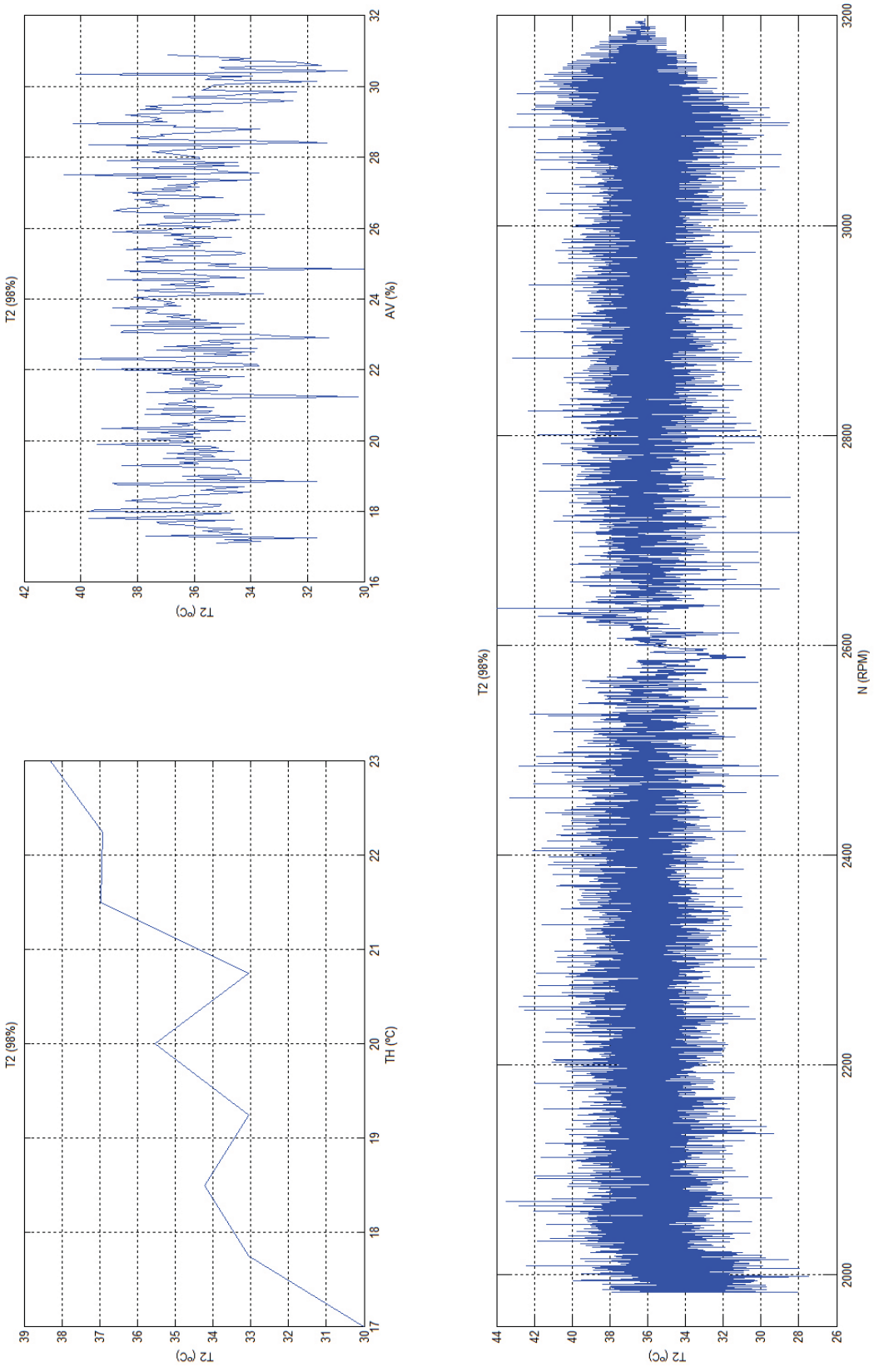


Figura B.30: Temperatura salida compresor T_2 (98%)

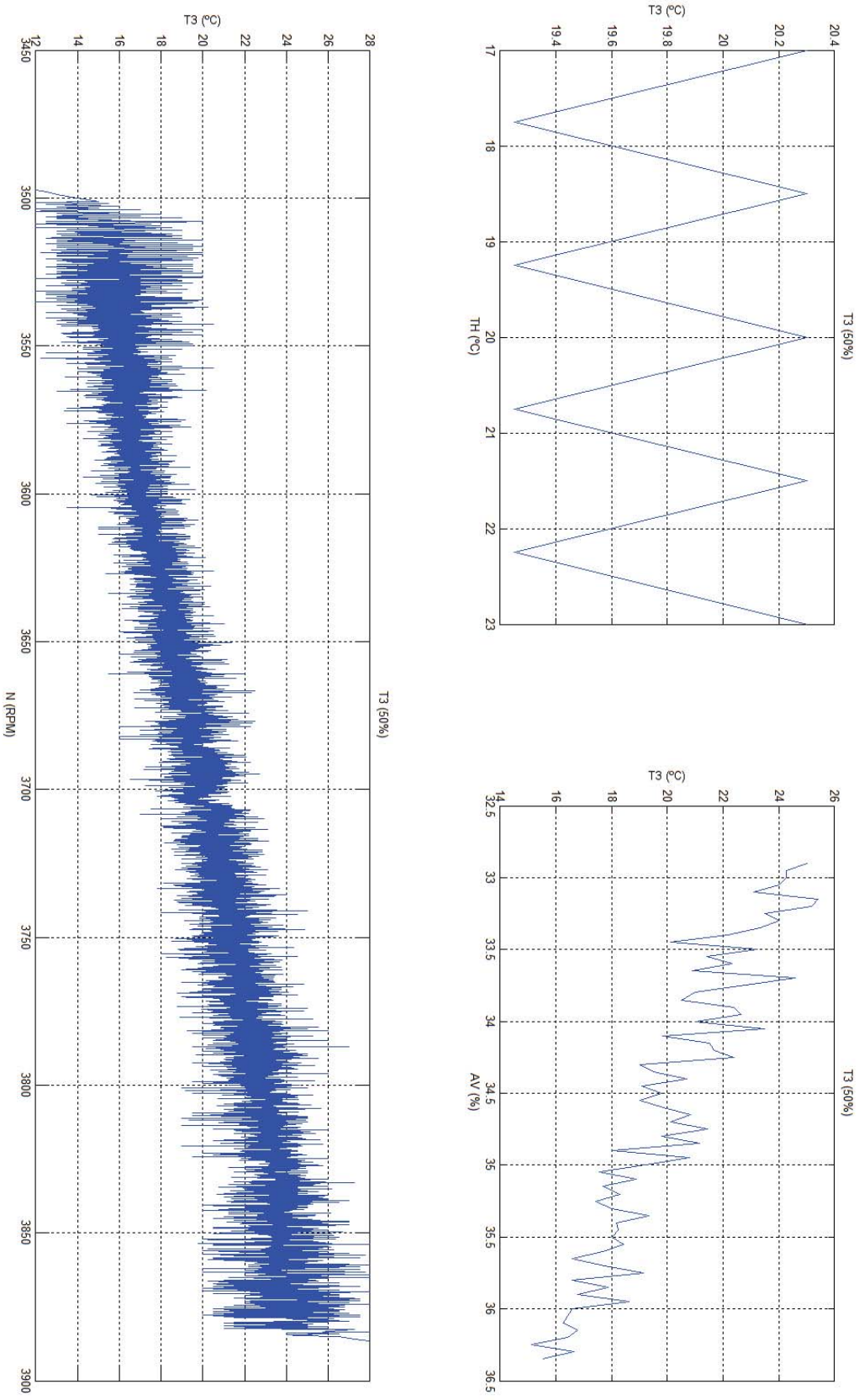


Figura B.31: Temperatura entrada válvula expansión T_3 (50%)

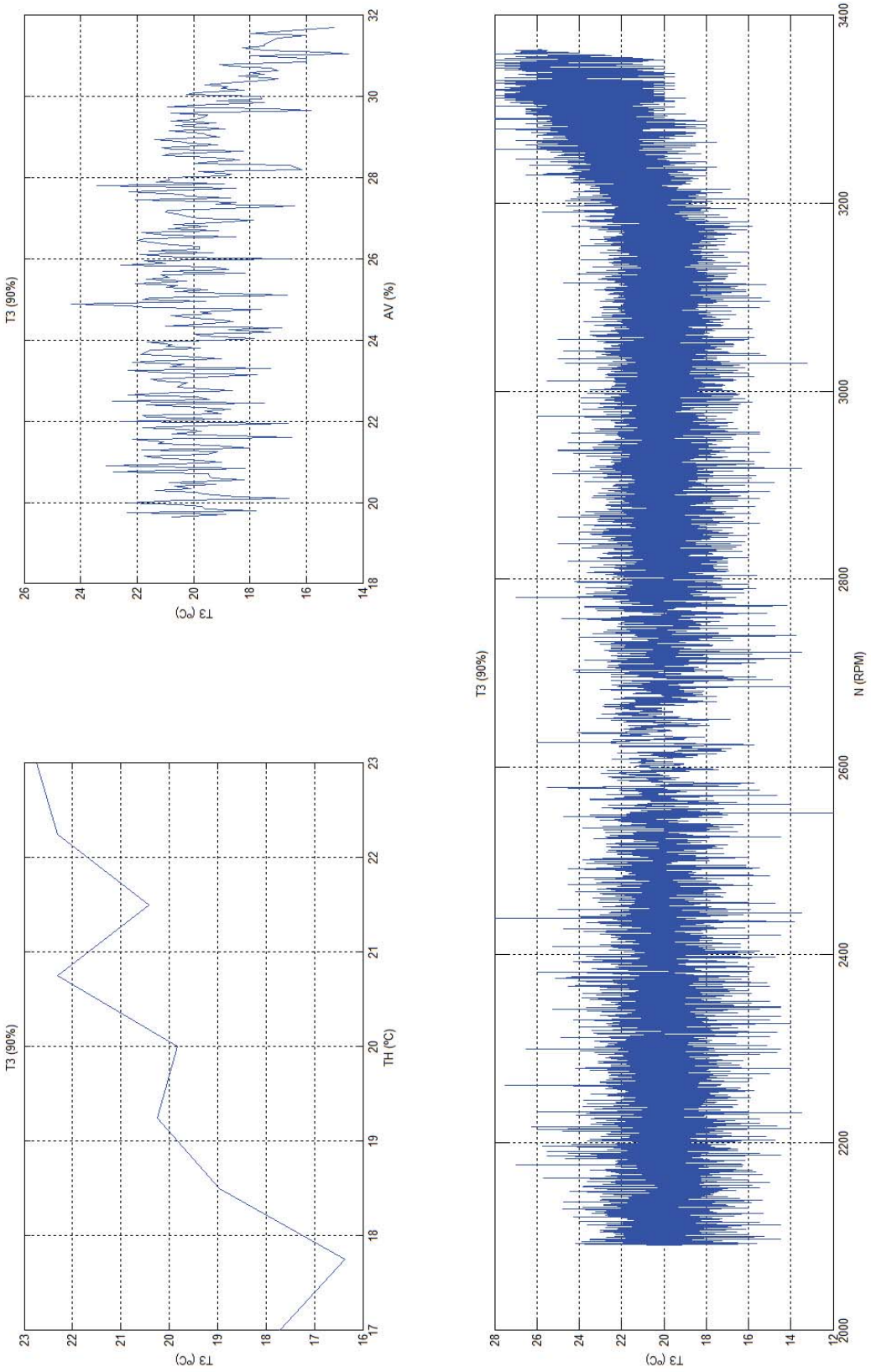


Figura B.32: Temperatura entrada válvula expansión T_3 (90%)

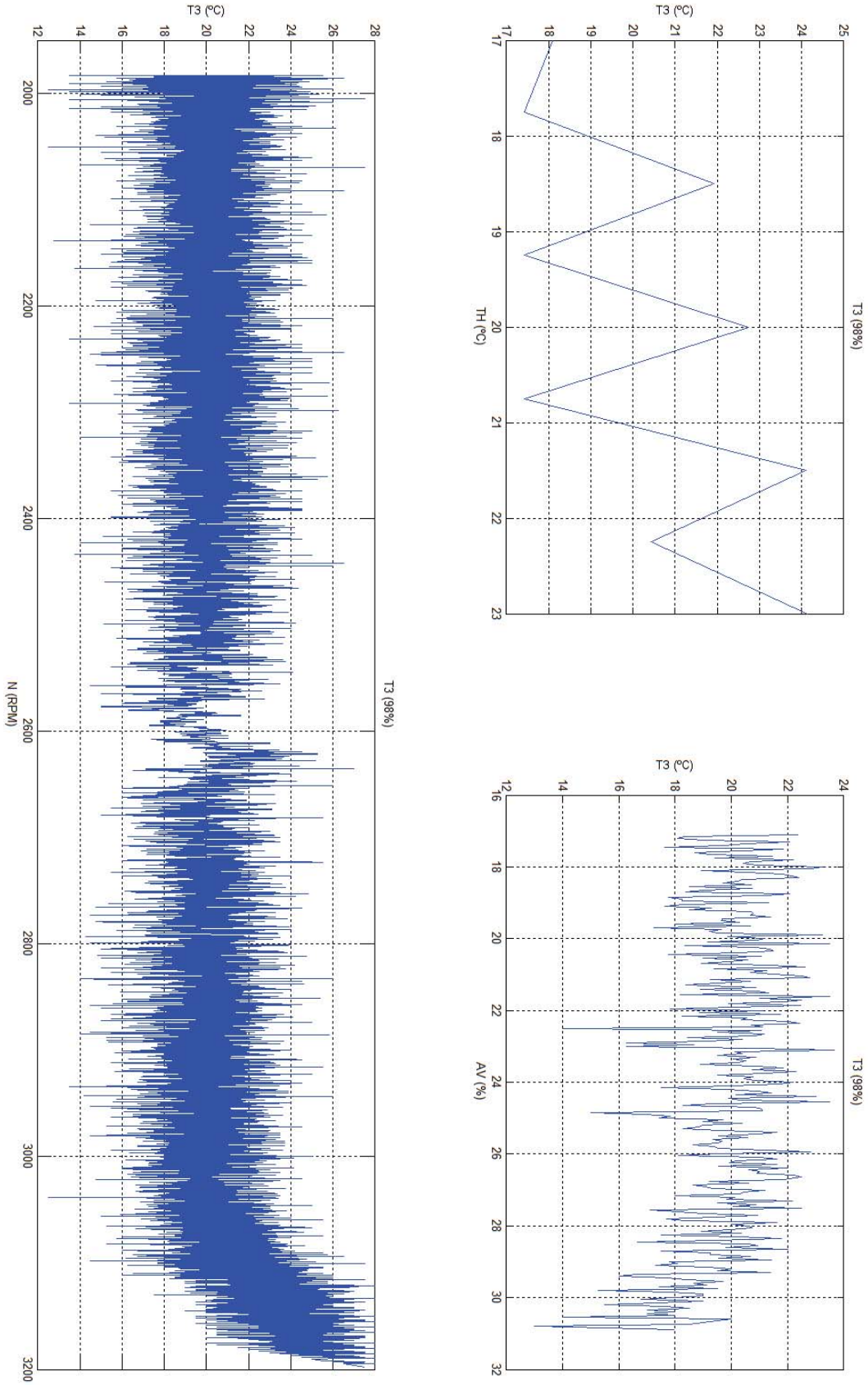


Figura B.33: Temperatura entrada válvula expansión T_3 (98%)

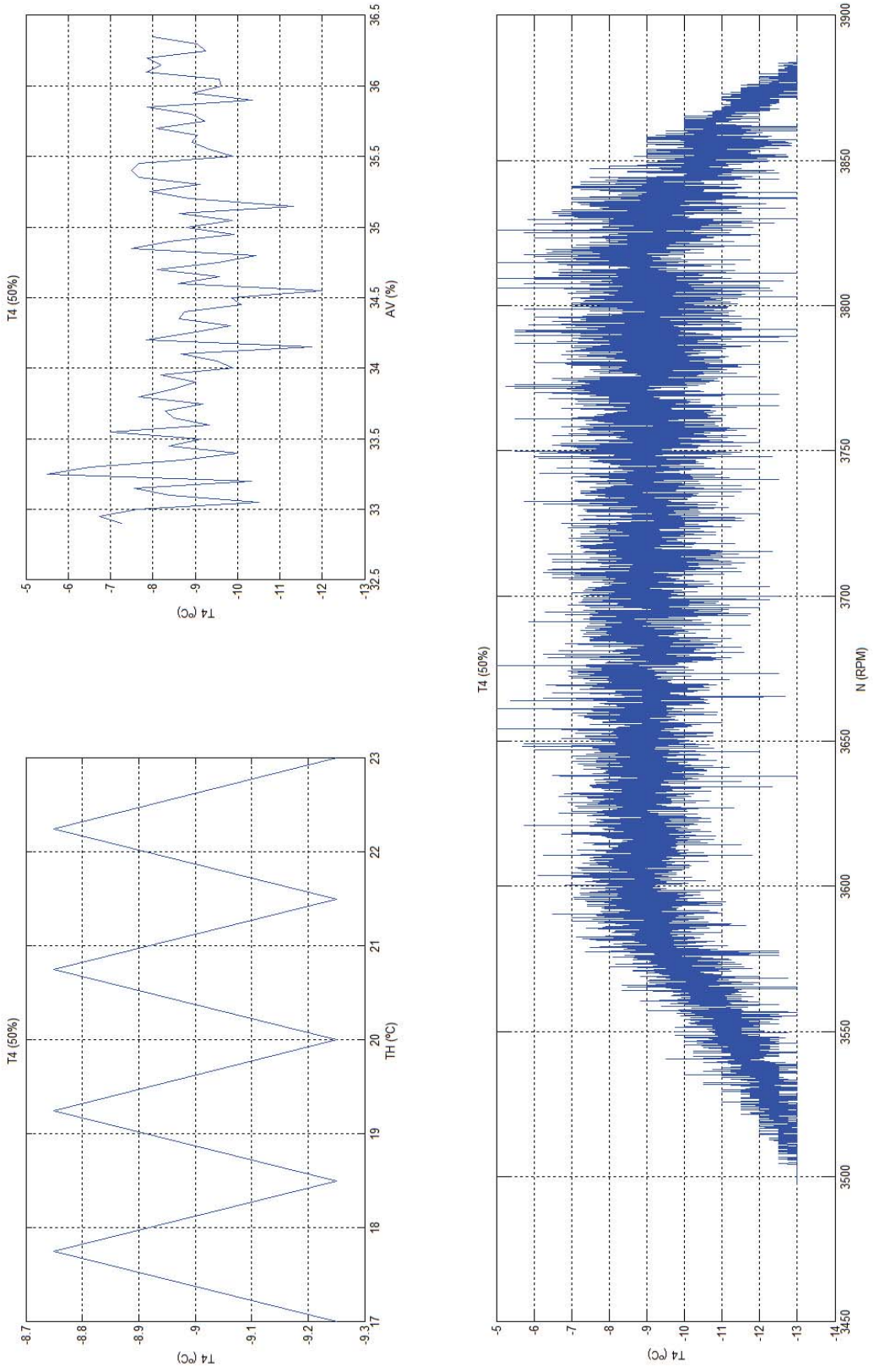


Figura B.34: Temperatura salida válvula expansión T_4 (50%)

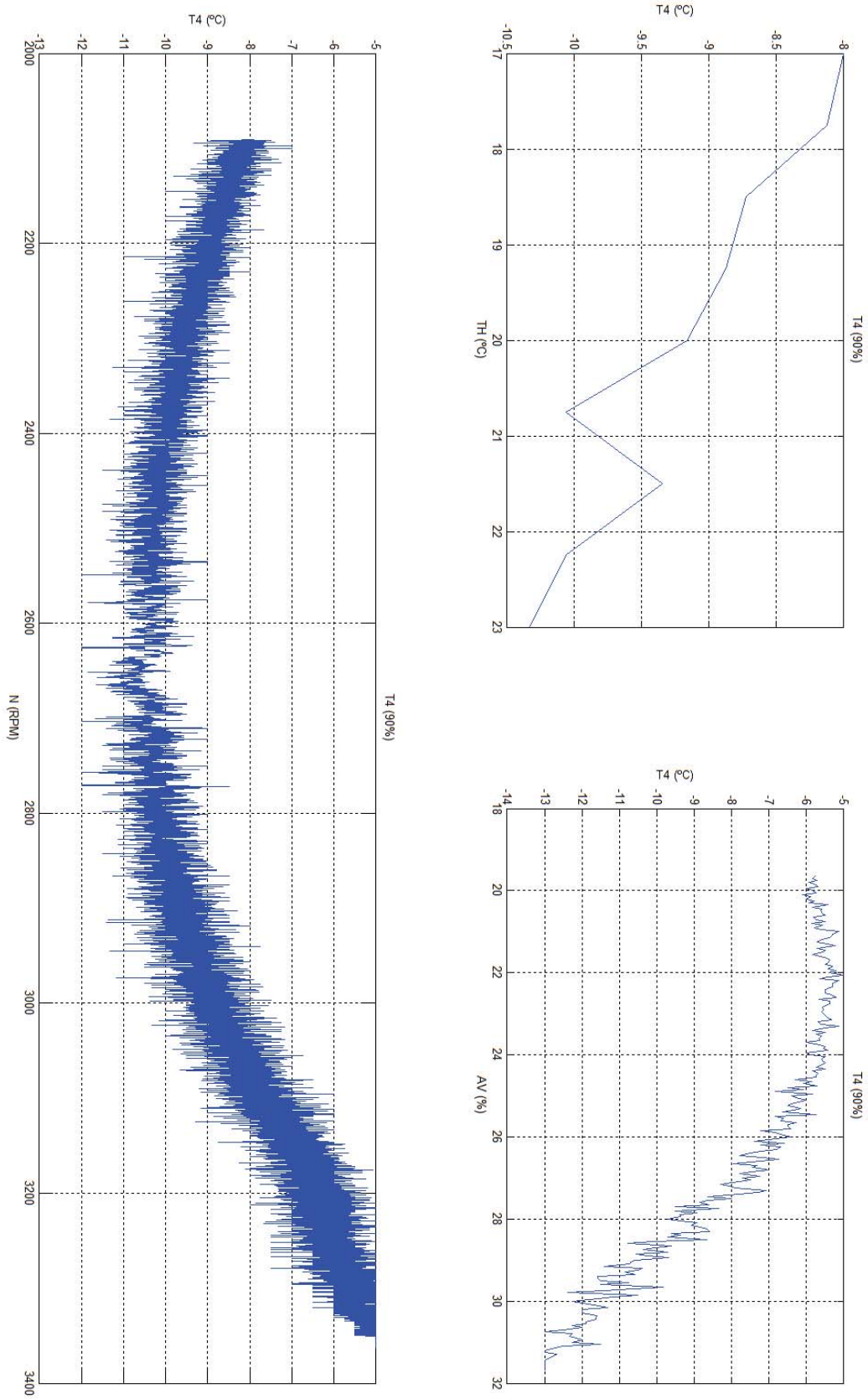


Figura B.35: Temperatura salida válvula expansión T_4 (90%)

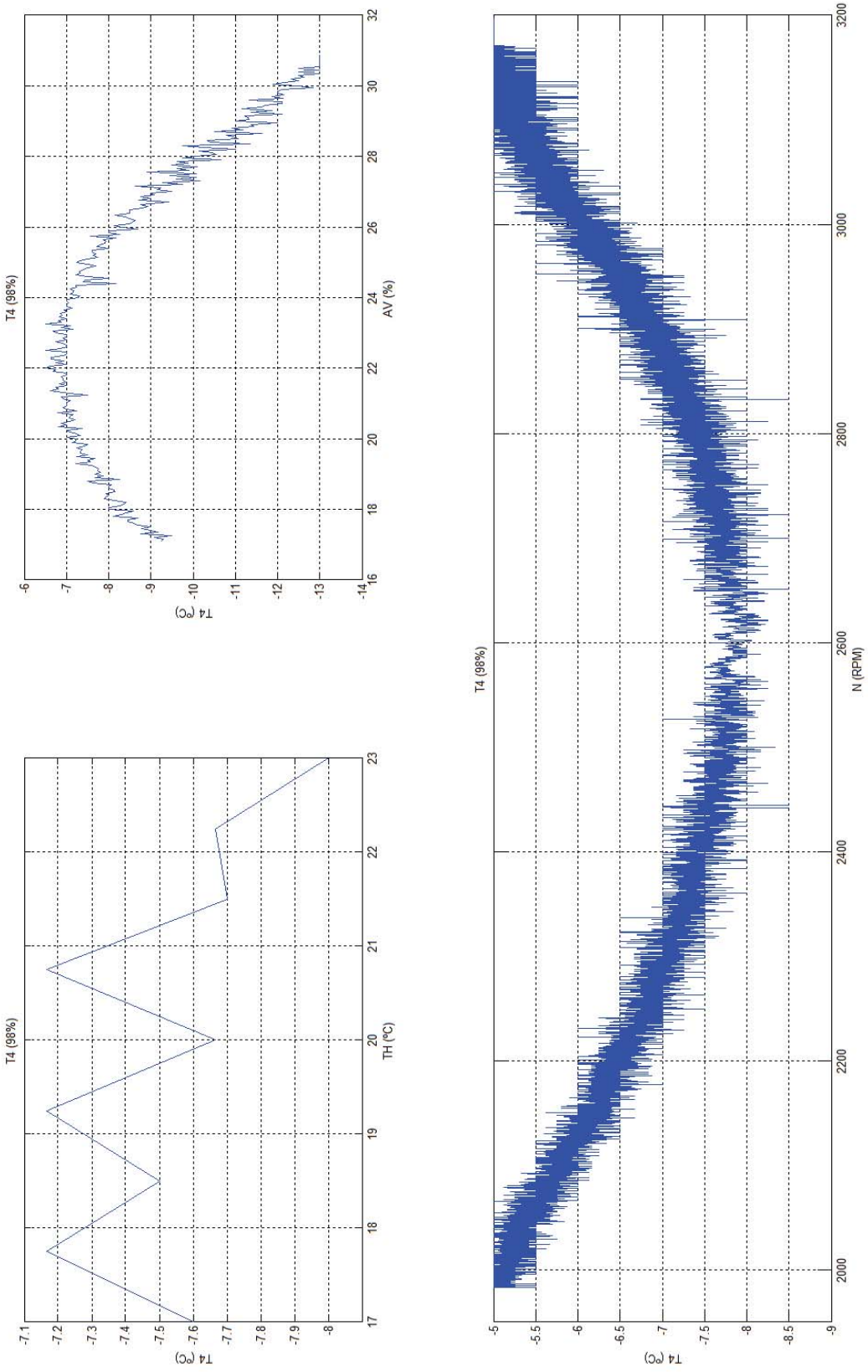


Figura B.36: Temperatura salida válvula expansión T_4 (98%)

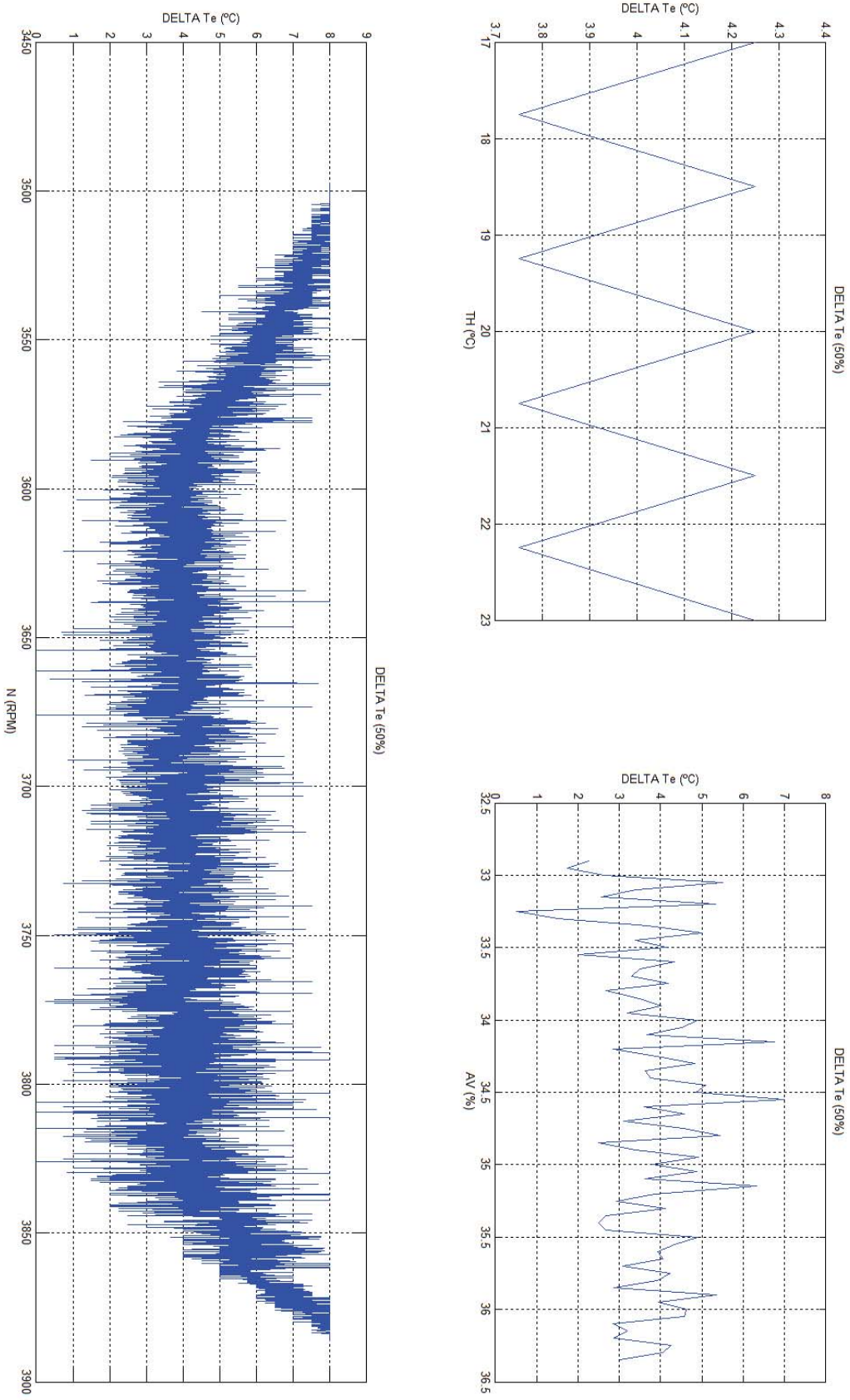


Figura B.37: Salto térmico evaporador ΔT_e (50%)

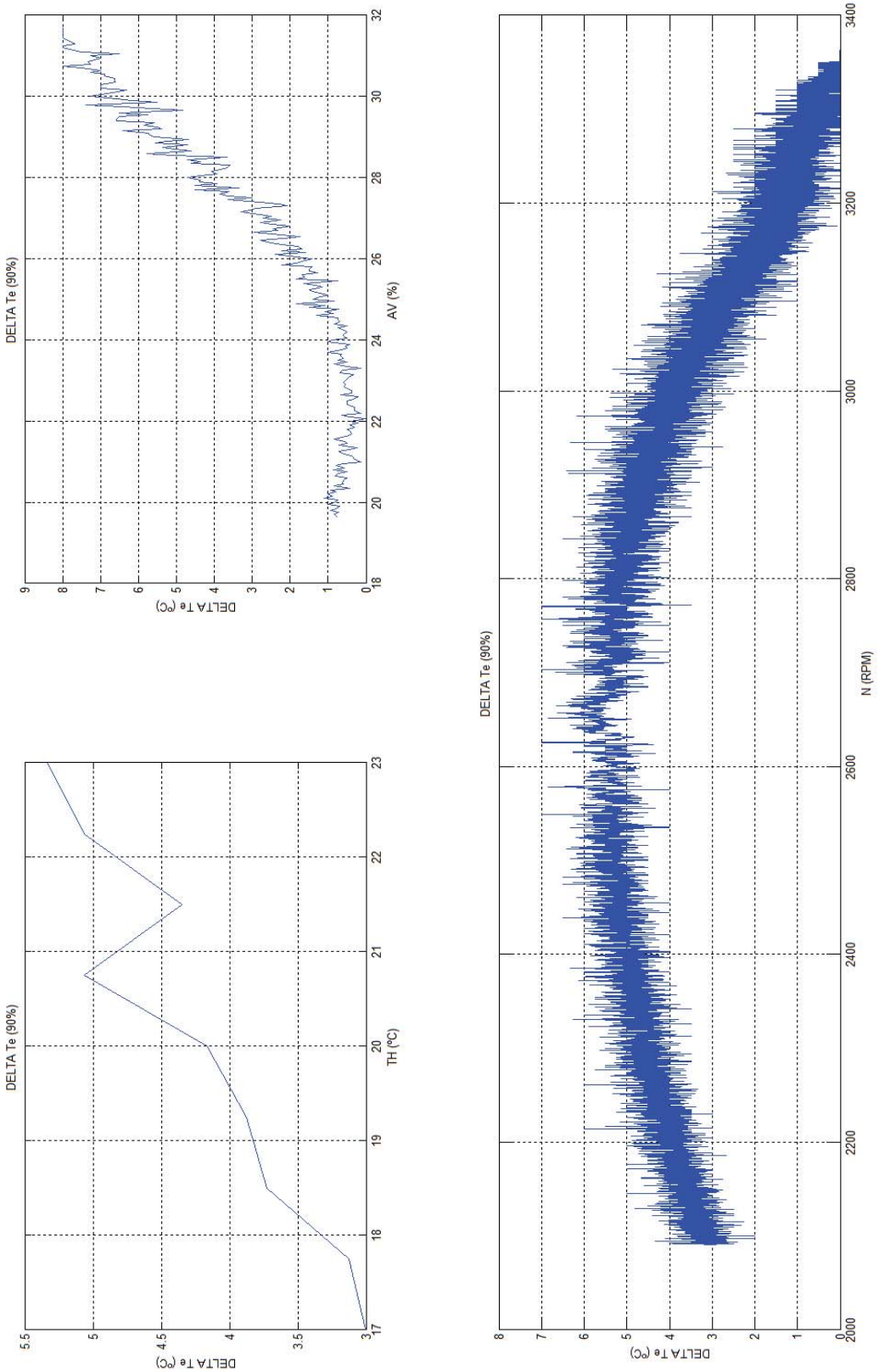


Figura B.38: Salto térmico evaporador ΔT_e (90%)

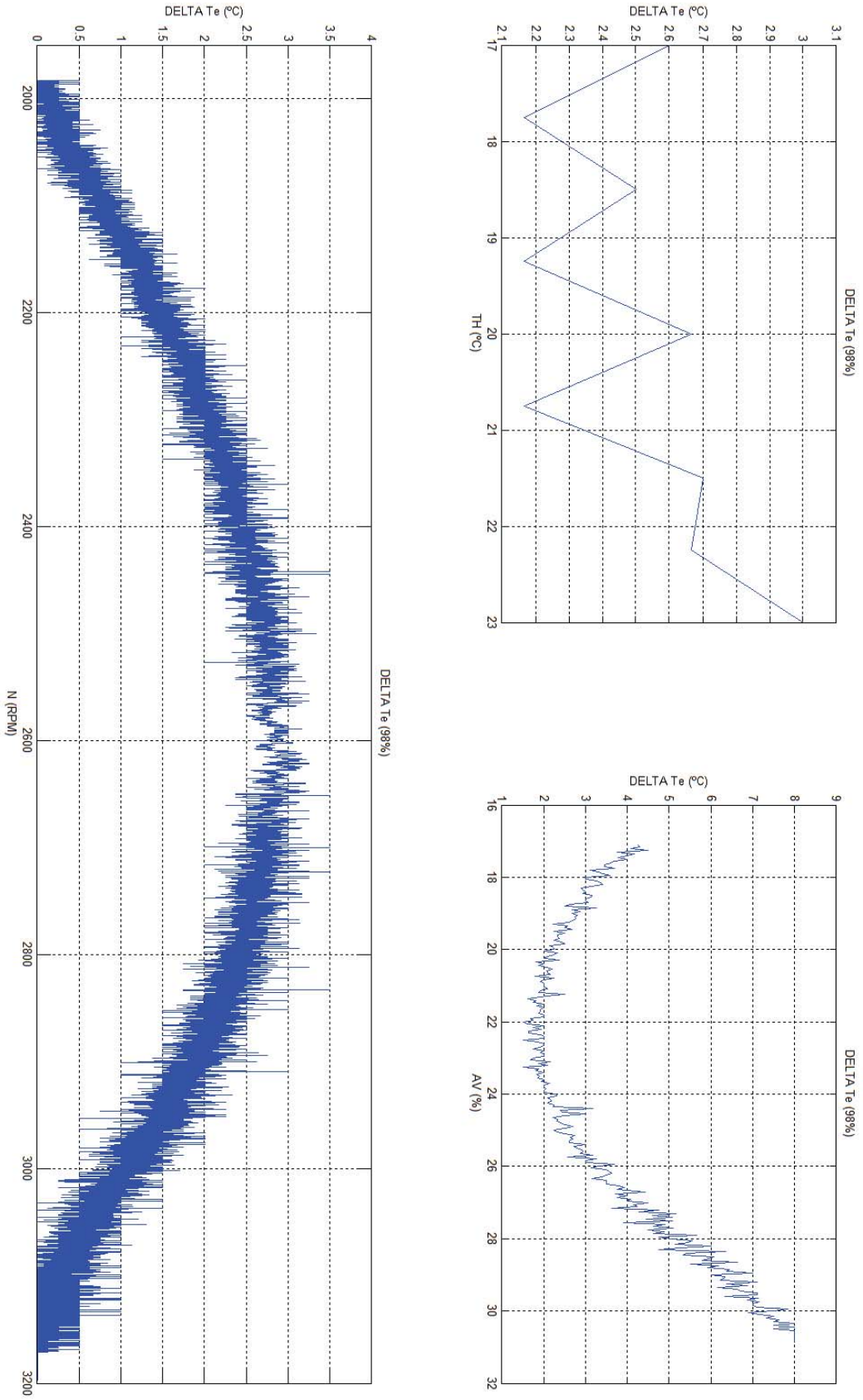


Figura B.39: Salto térmico evaporador ΔT_e (98%)

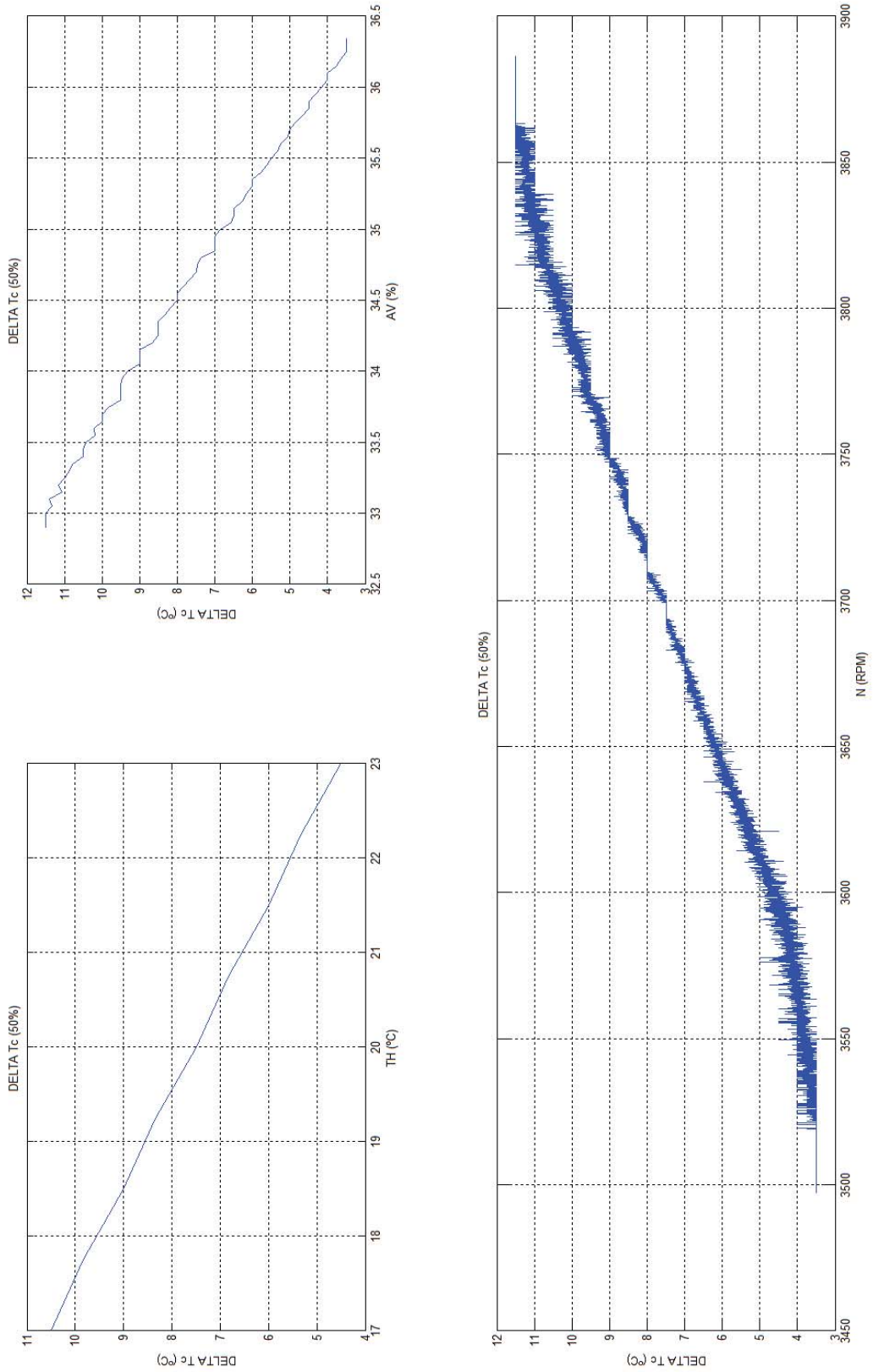


Figura B.40: Salto térmico condensador ΔT_c (50%)

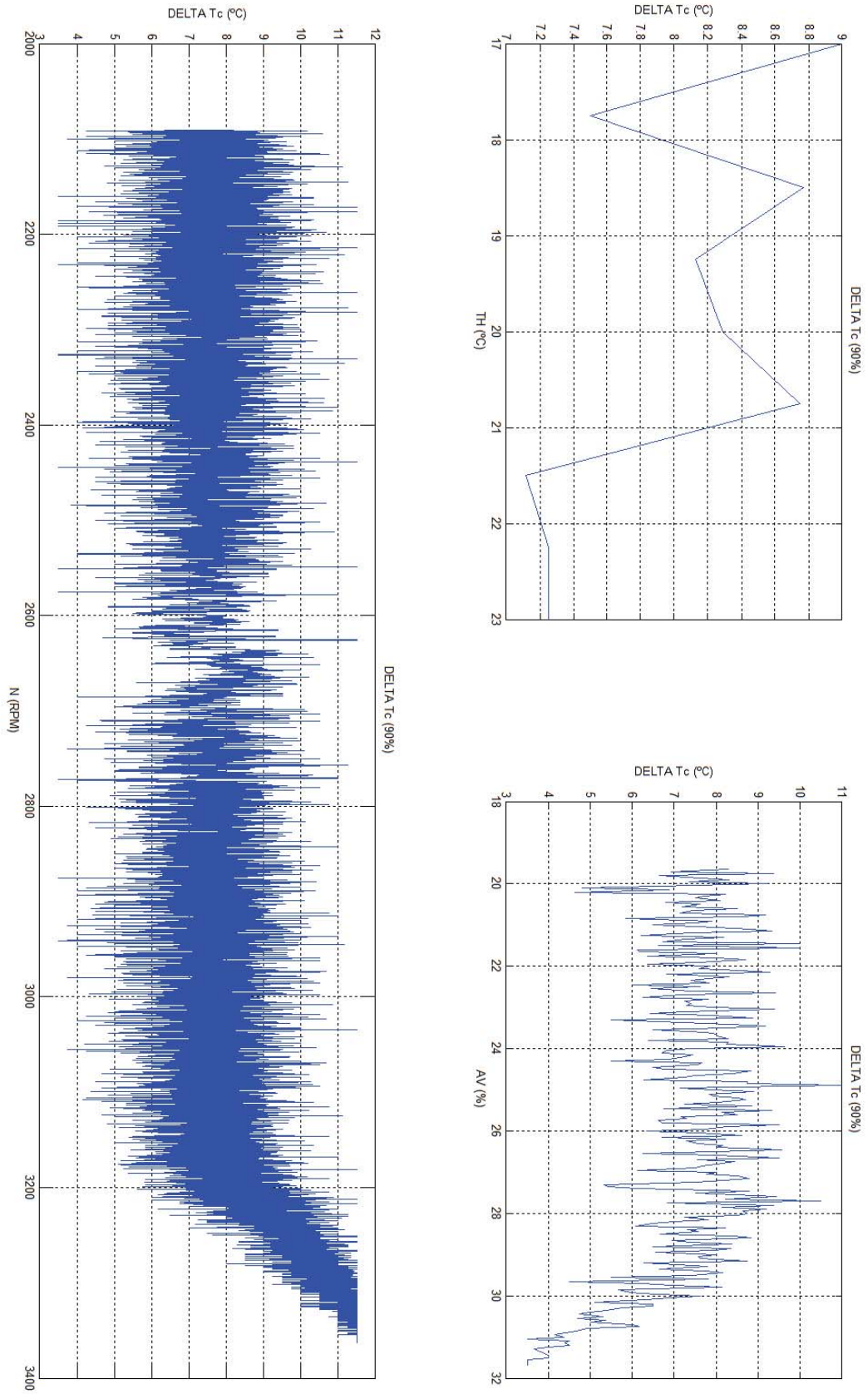


Figura B.41: Salto térmico condensador ΔT_c (90%)

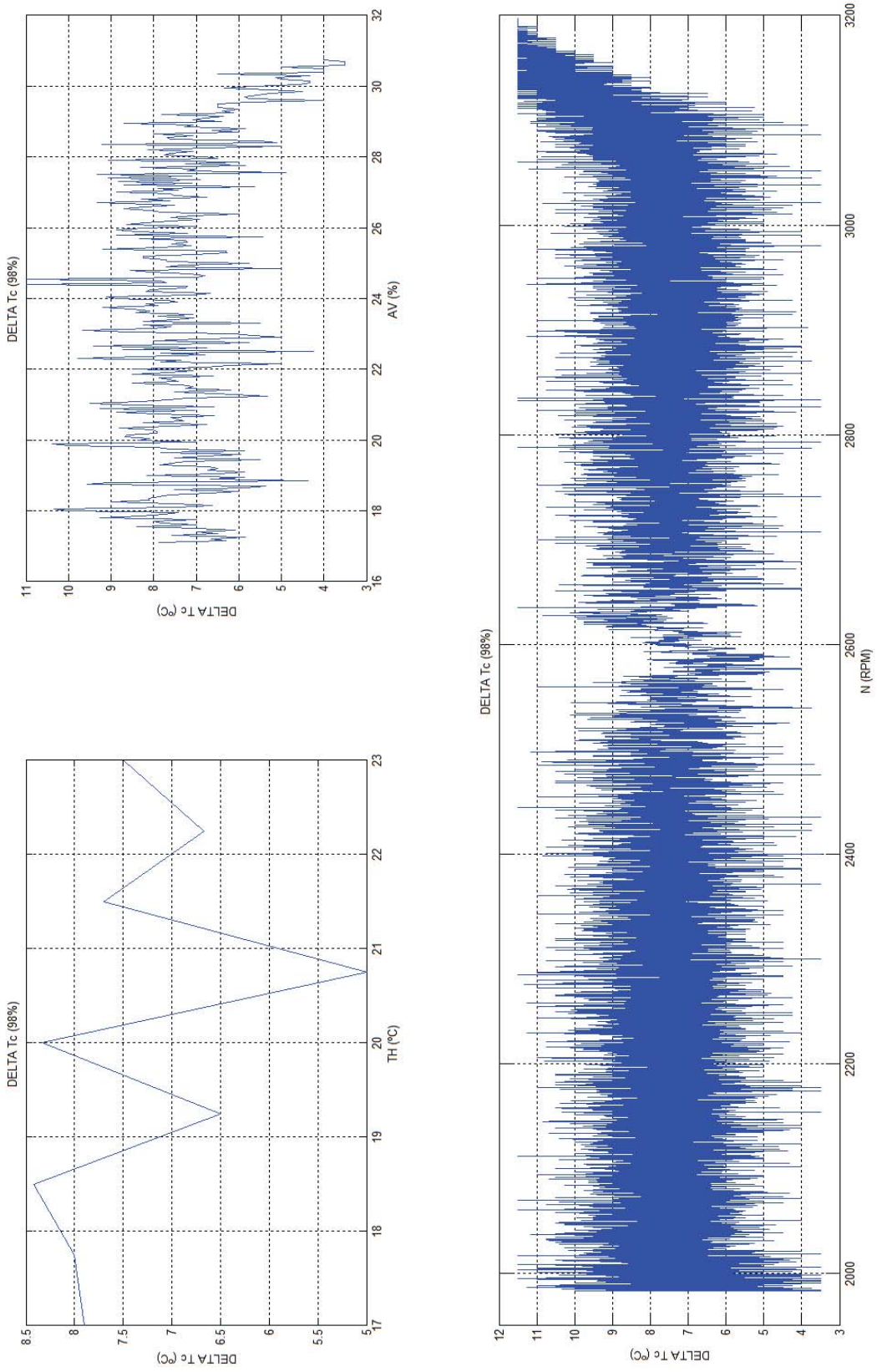


Figura B.42: Salto térmico condensador ΔT_c (98%)

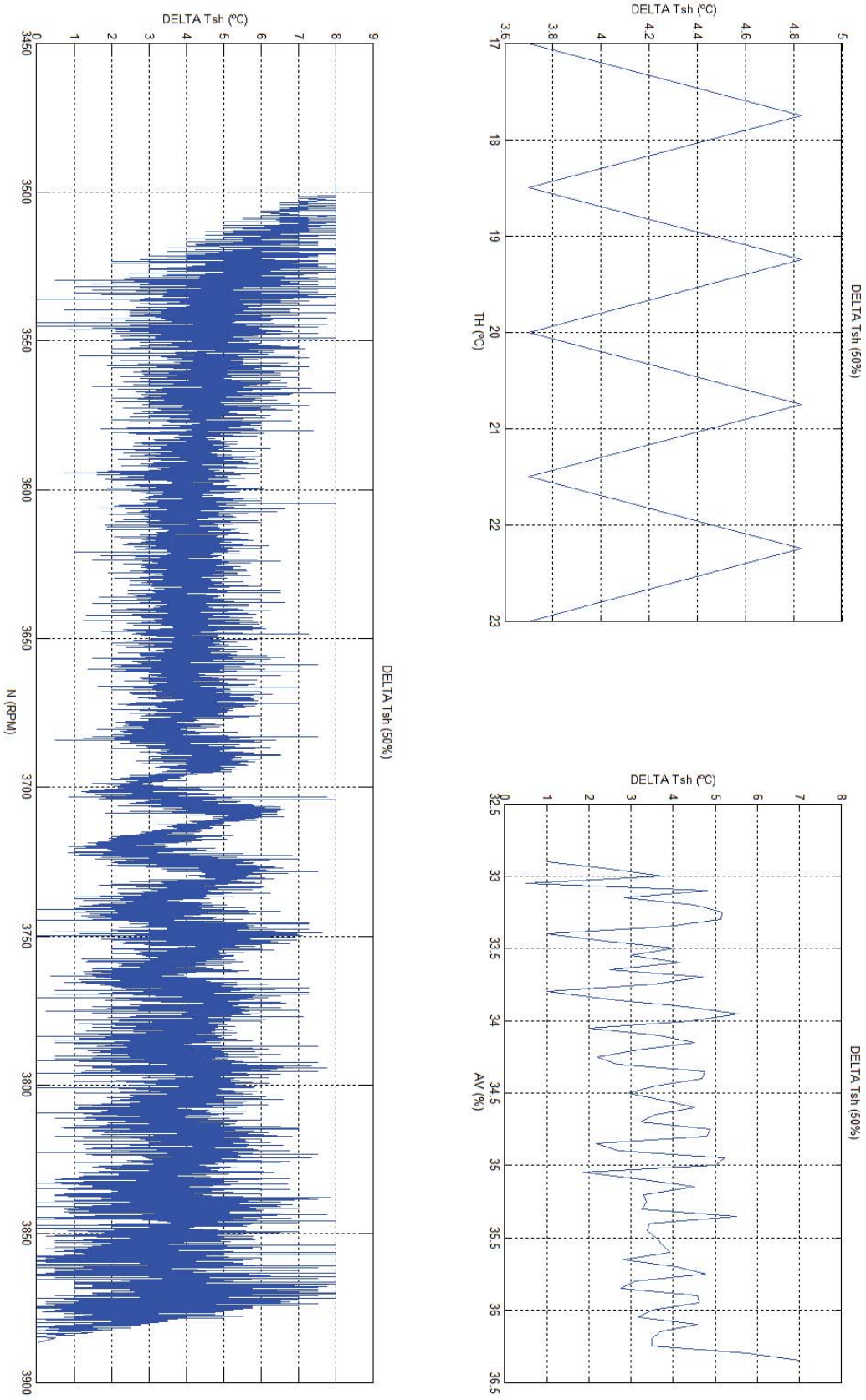


Figura B.43: Sobrecalentamiento ΔT_{sc} (50%)

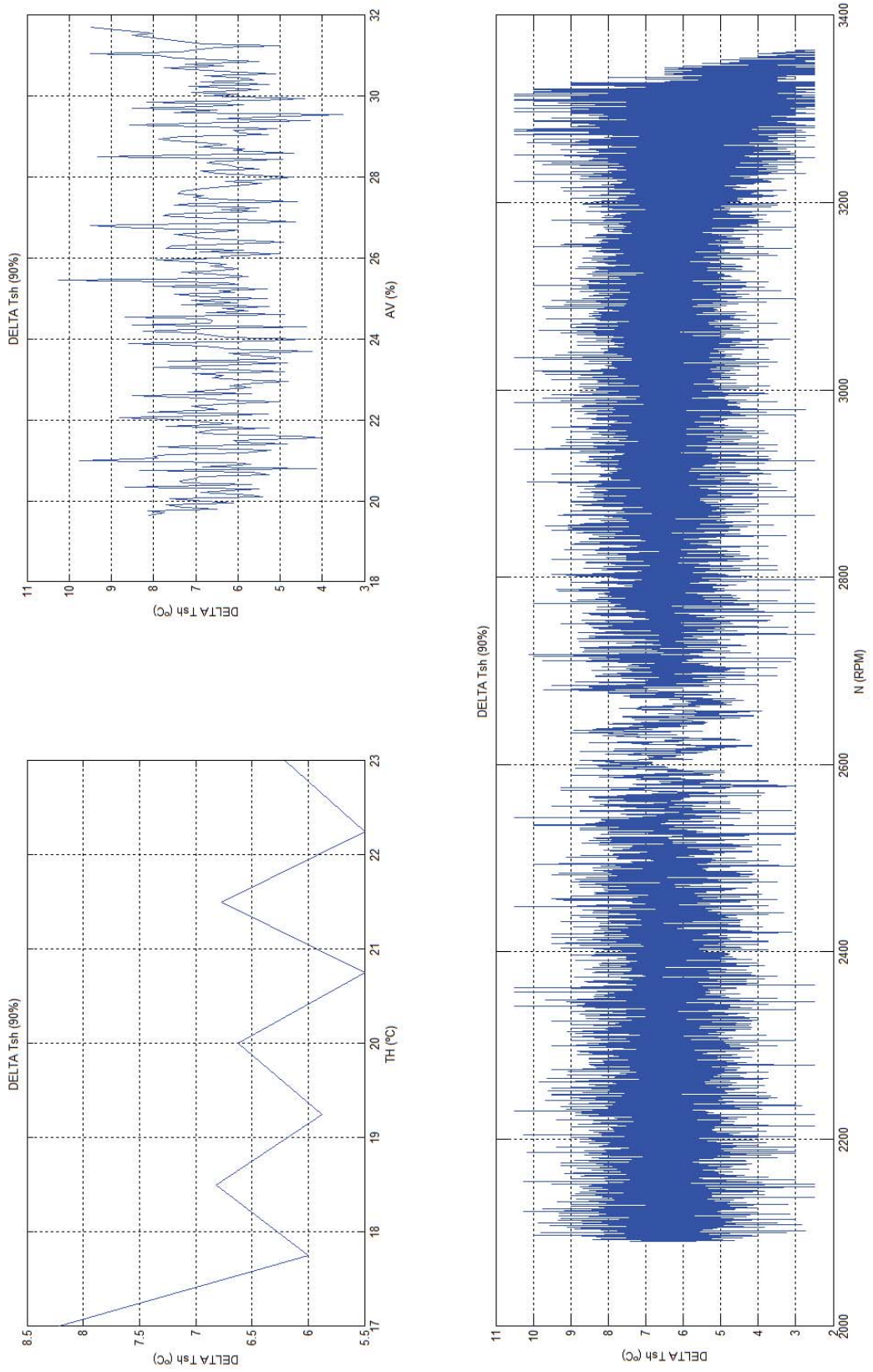


Figura B.44: Sobrecalentamiento ΔT_{sc} (90%)

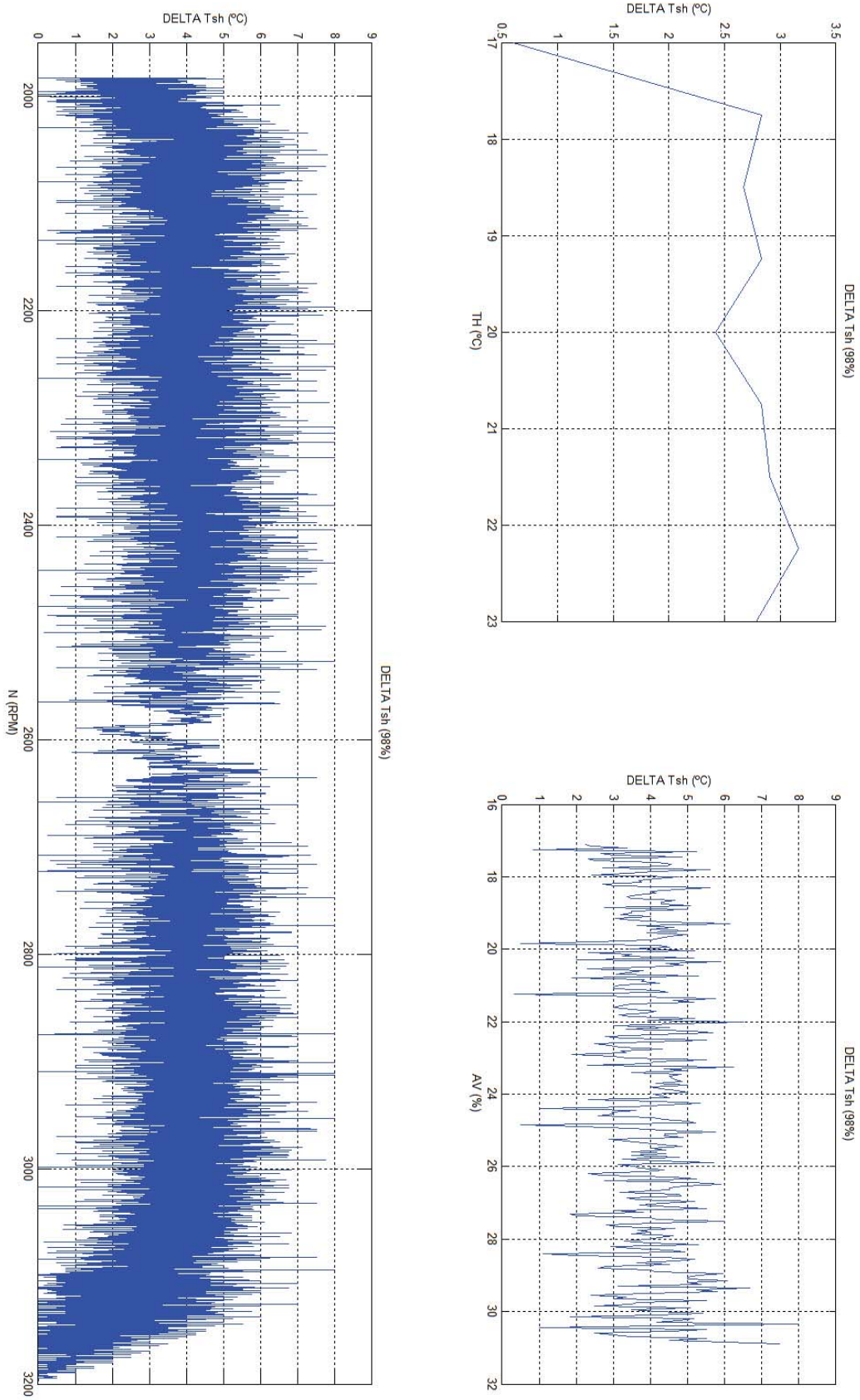


Figura B.45: Sobrecalentamiento ΔT_{sc} (98%)

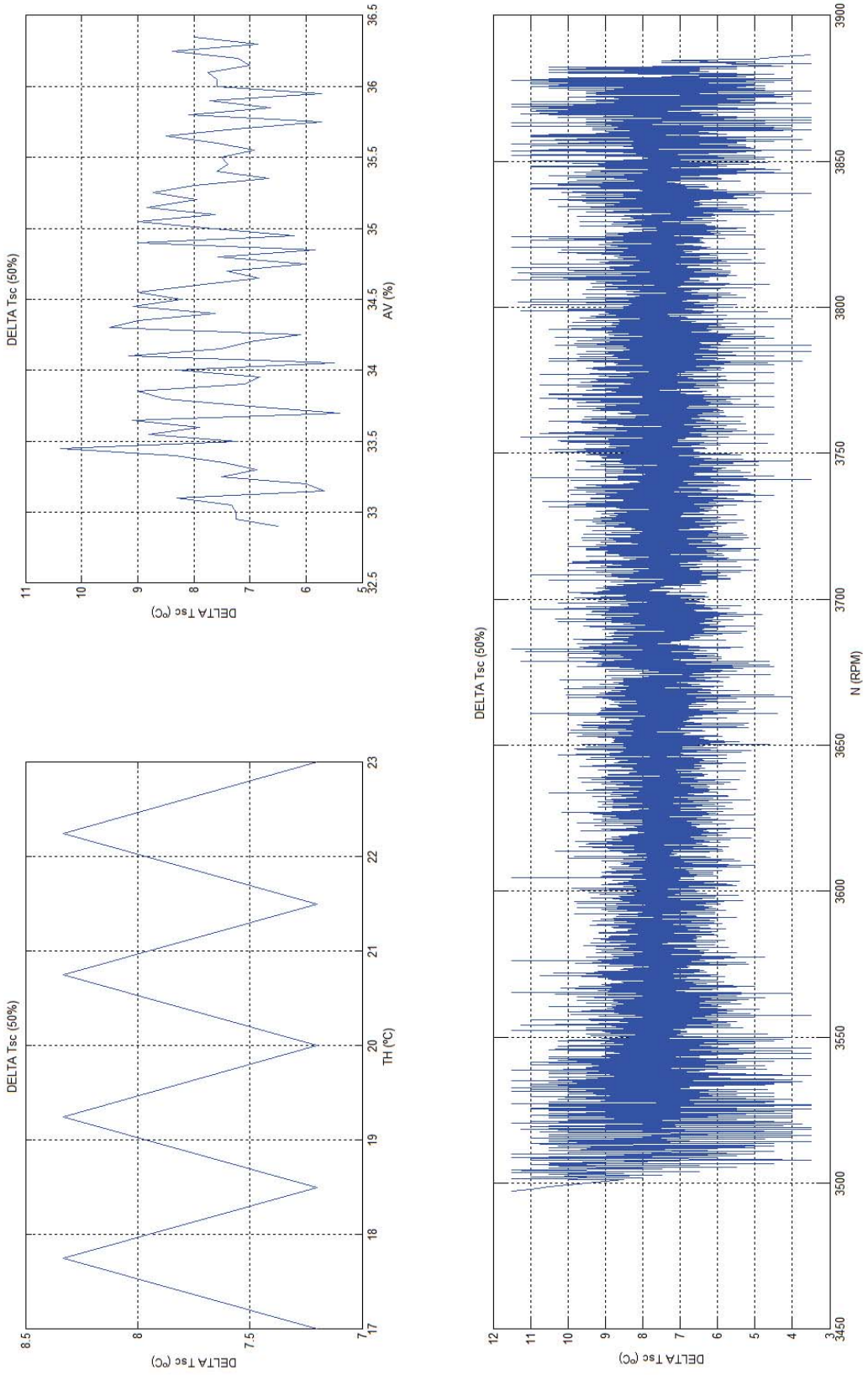


Figura B.46: Subenfriamiento ΔT_{sc} (50%)

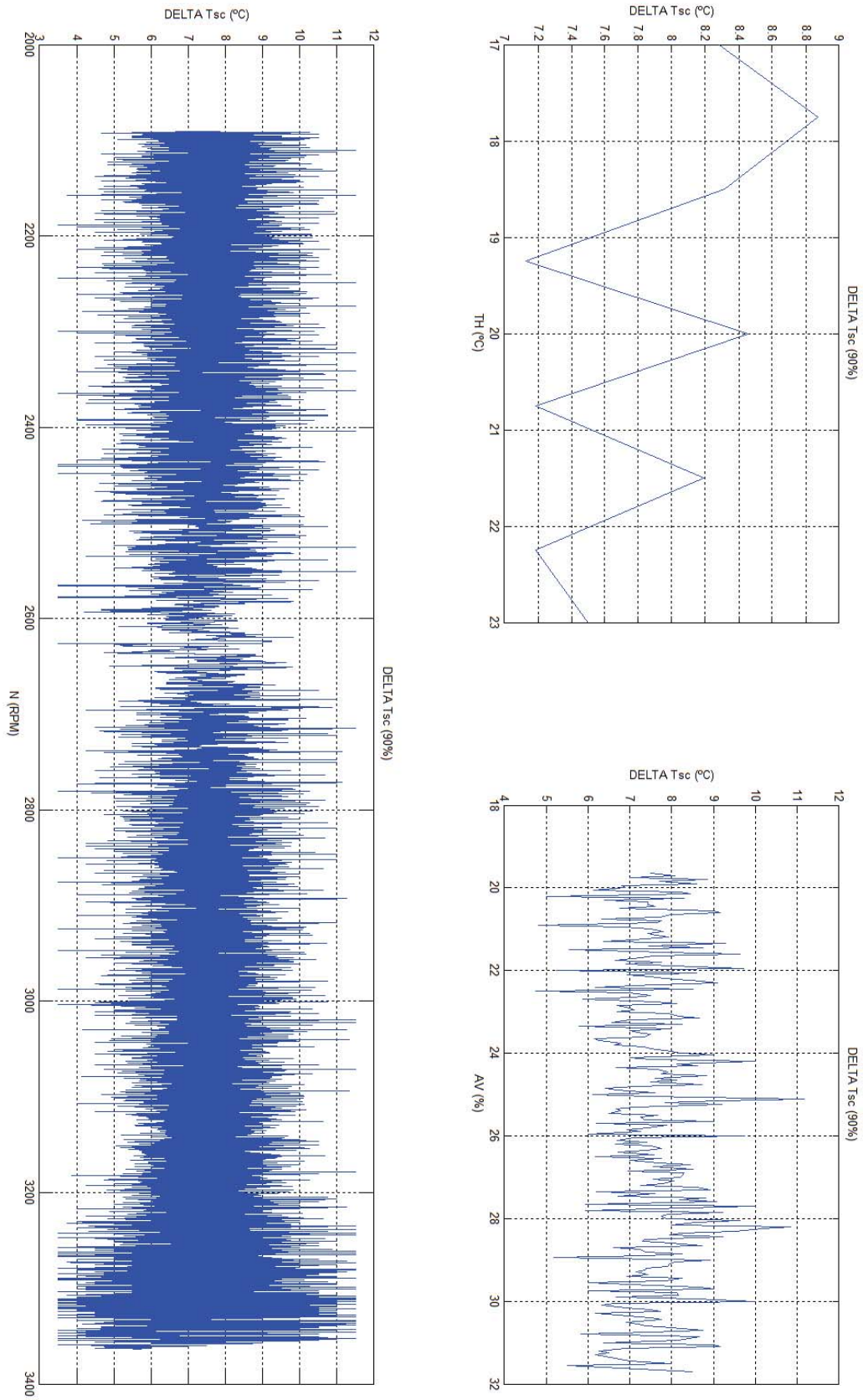


Figura B.47: Subenfriamiento ΔT_{sc} (90%)

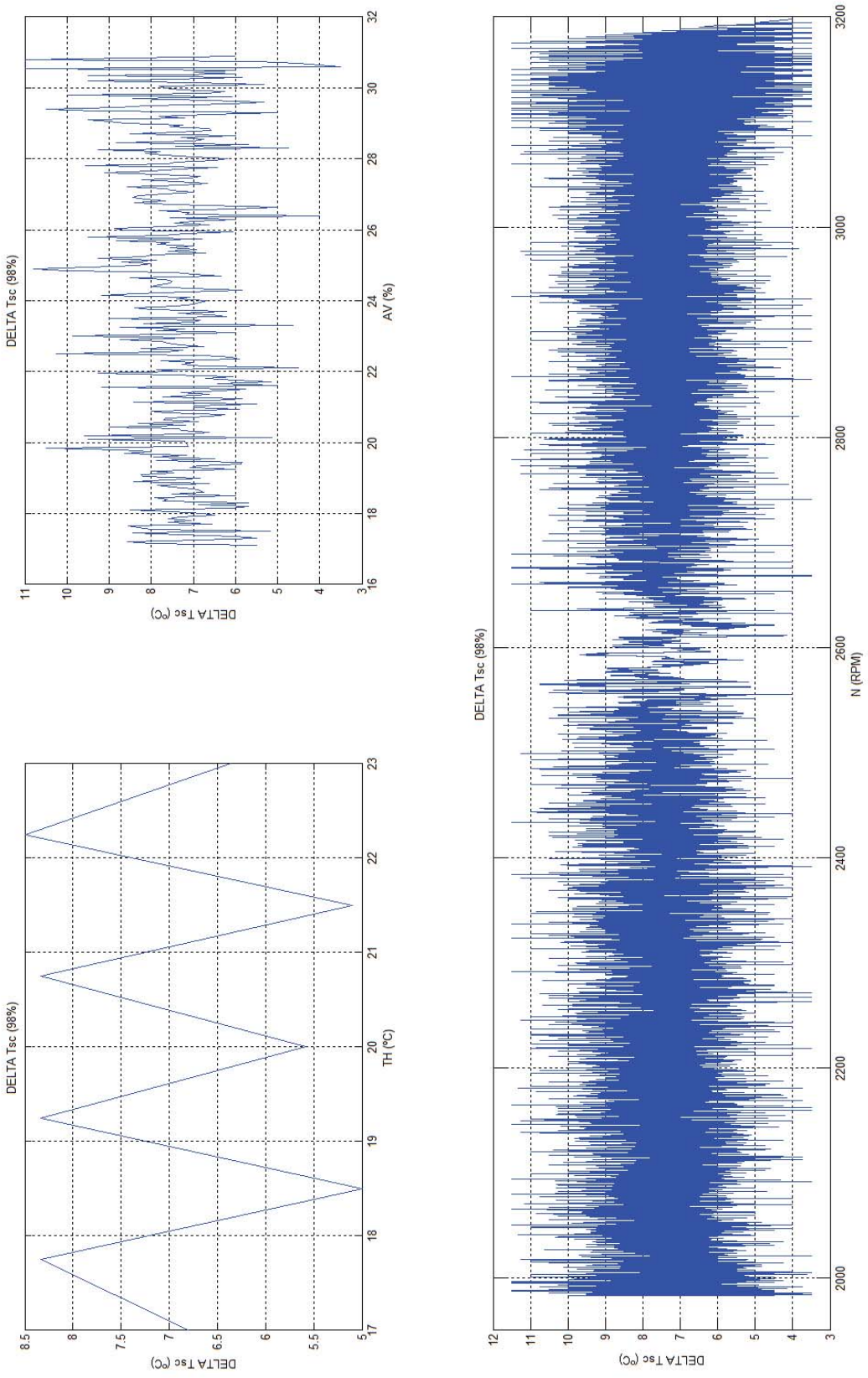


Figura B.48: Subenfriamiento ΔT_{sc} (98%)

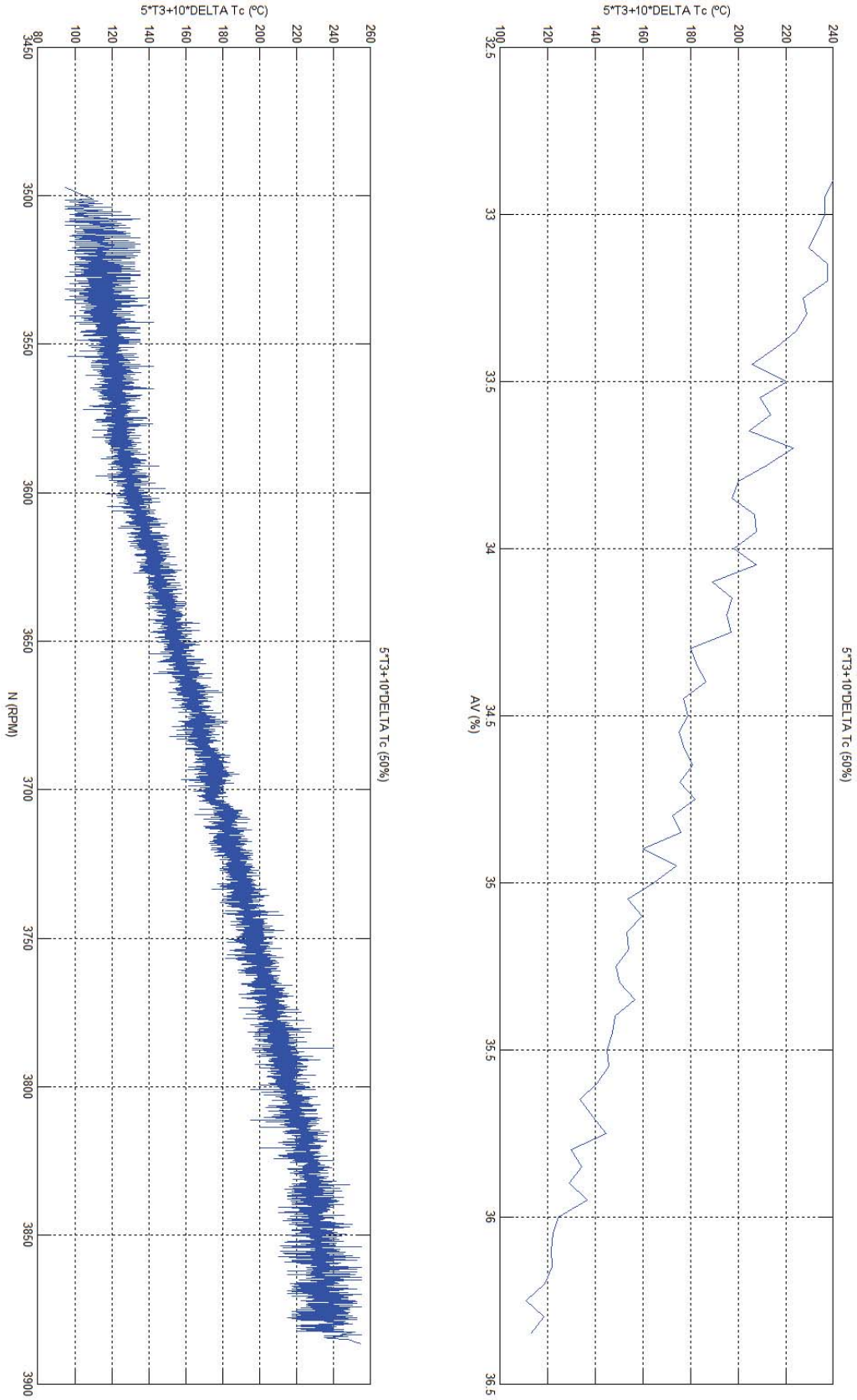


Figura B.49: $f_1(T_3, \Delta T_c) = 5T_3 + 10\Delta T_c$ *(50%)*

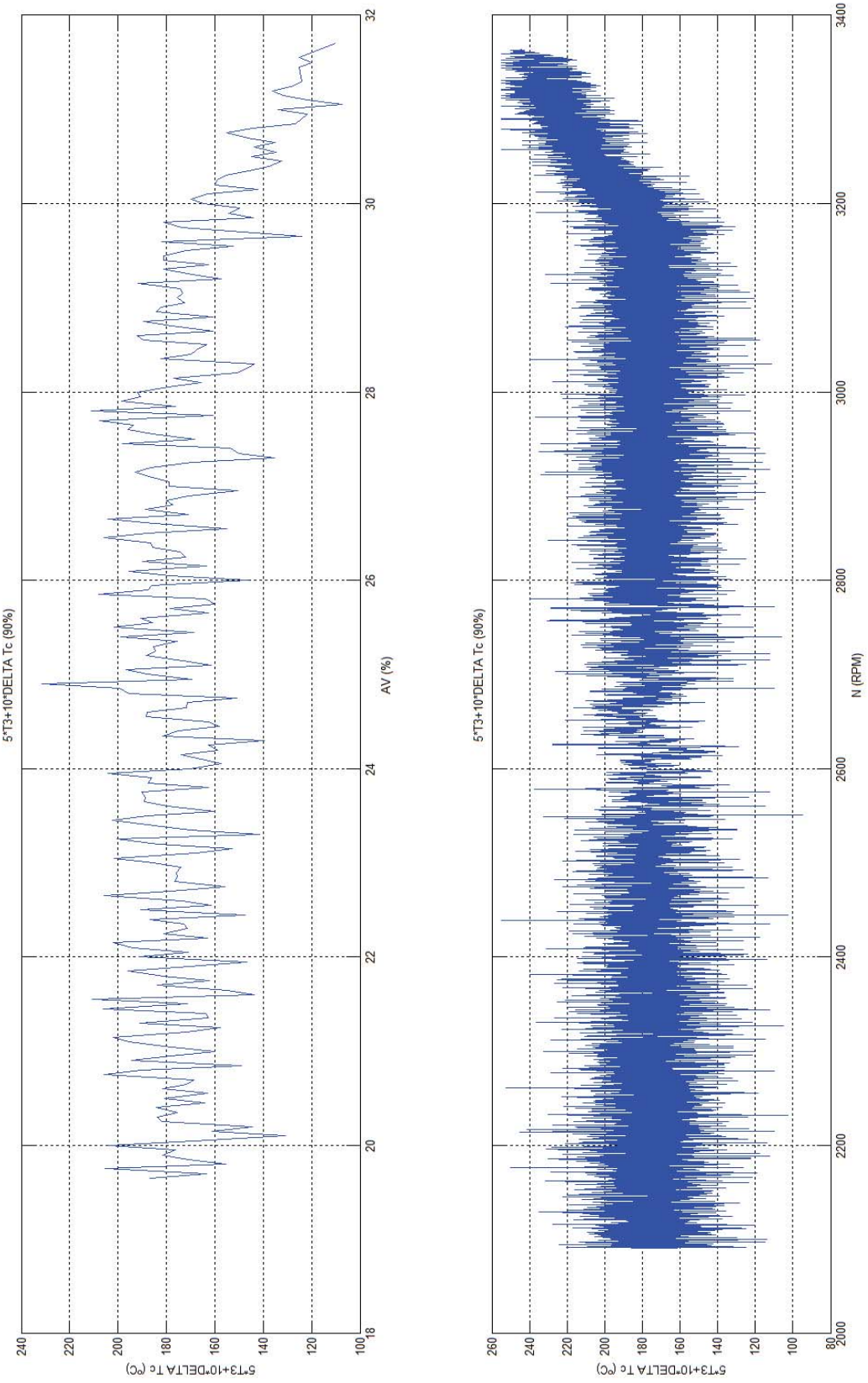


Figura B.50: $f_1(T_3, \Delta T_c) = 5T_3 + 10\Delta T_c$ *(90%)*

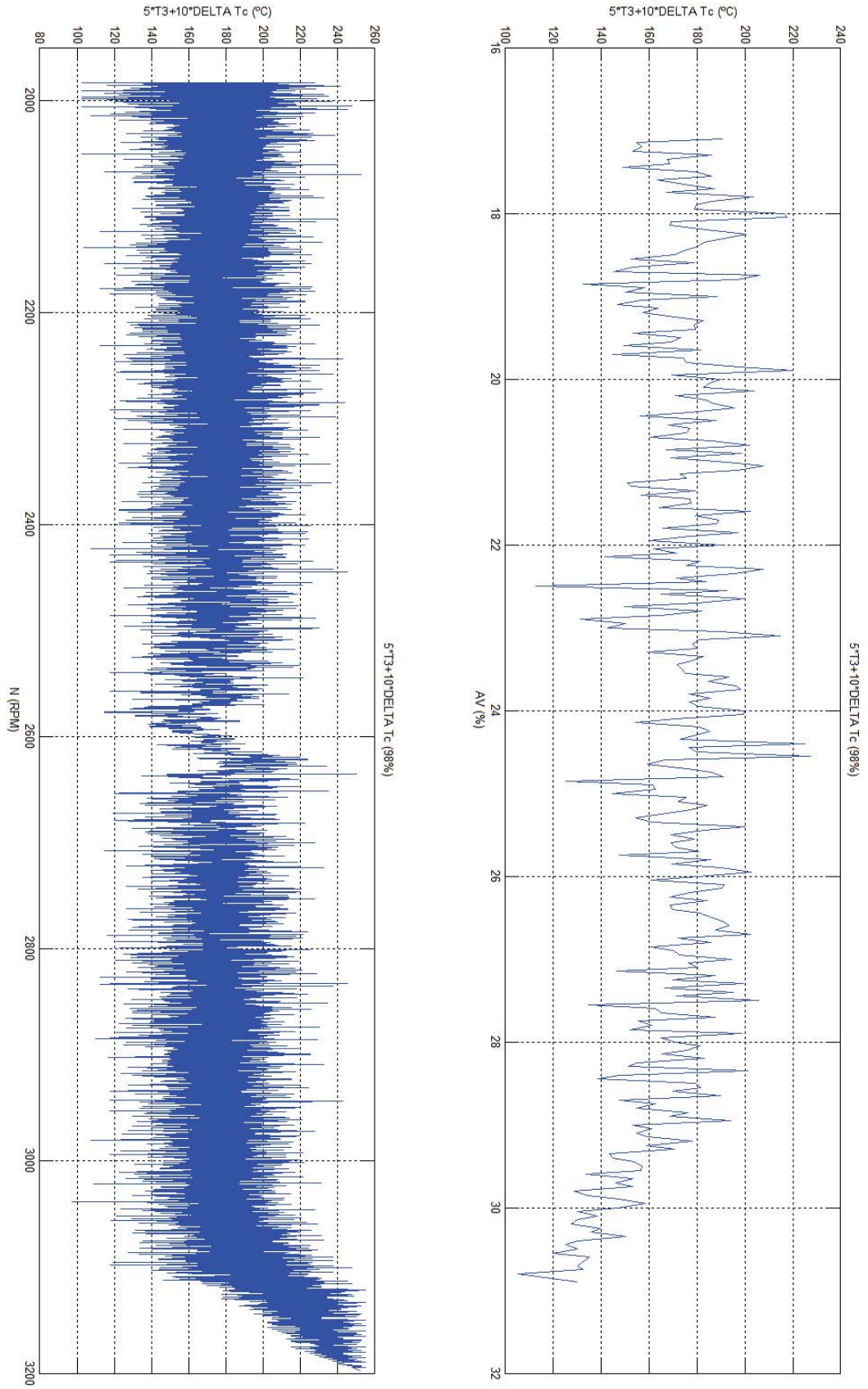


Figura B.51: $f_1(T_3, \Delta T_c) = 5T_3 + 10\Delta T_c$ *(98%)*

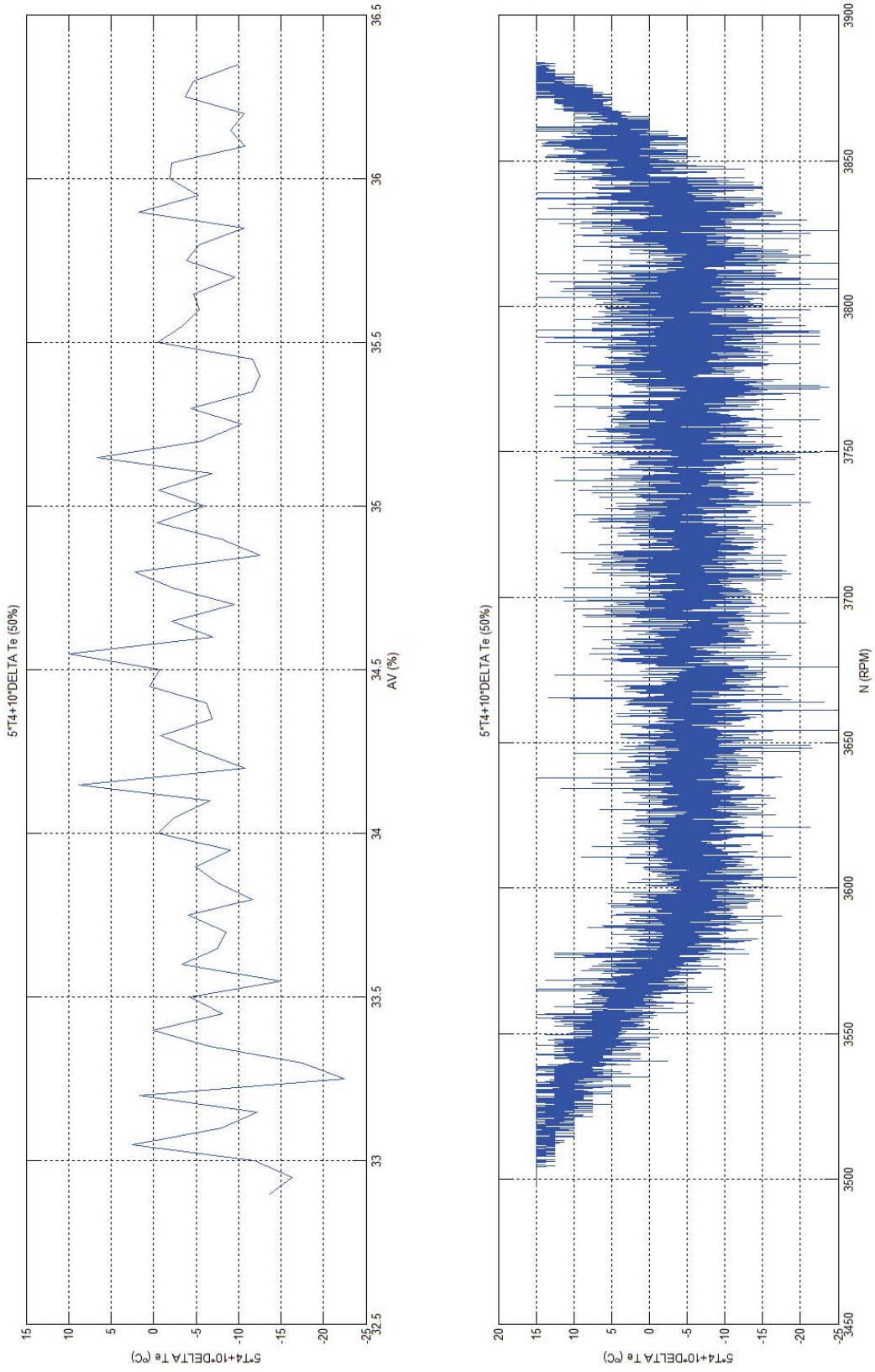


Figura B.52: $f_2(T_4, \Delta T_e) = 5T_4 + 10\Delta T_e$ *(50%)*

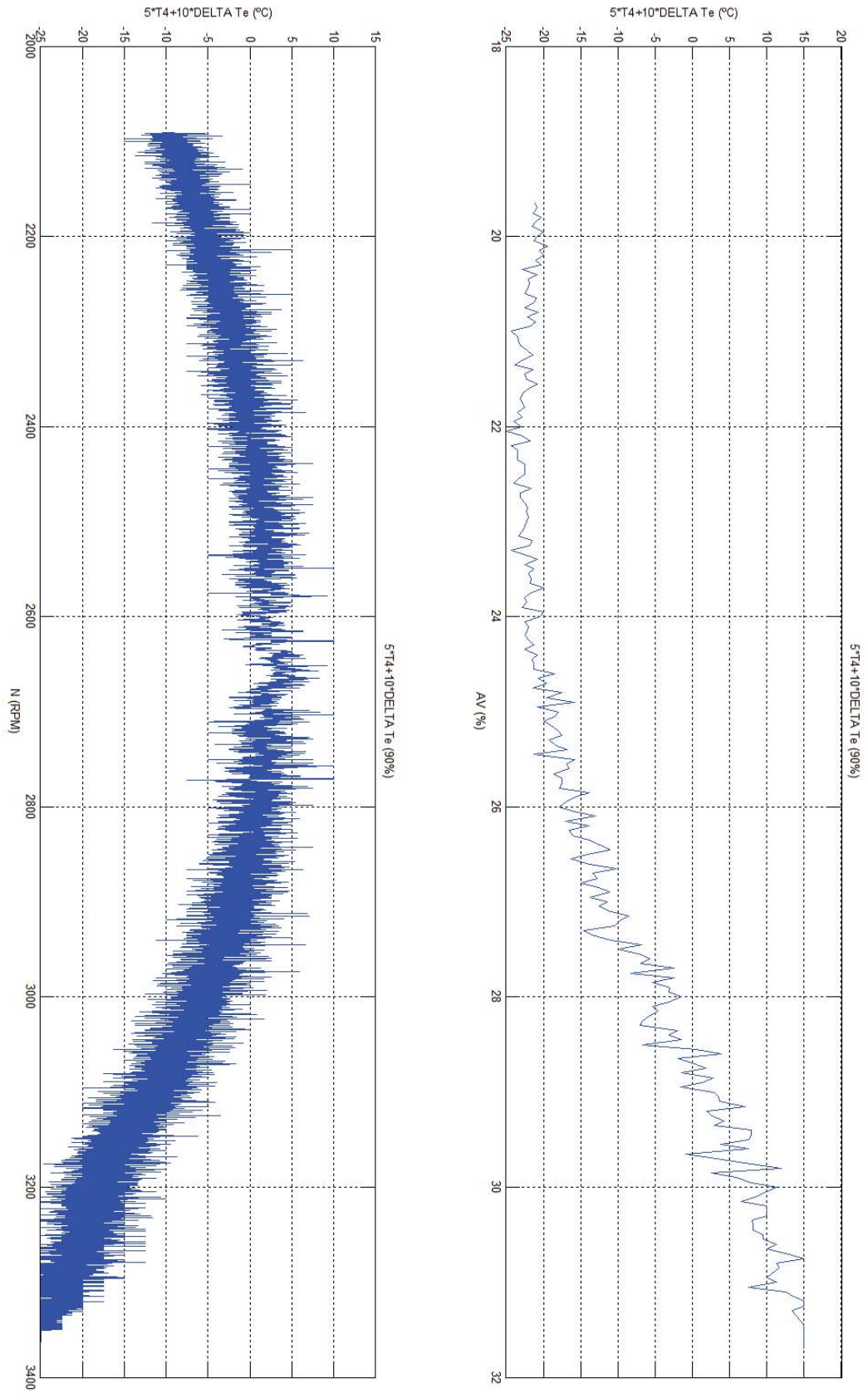


Figura B.53: $f_2(T_4, \Delta T_e) = 5T_4 + 10\Delta T_e$ *(90%)*

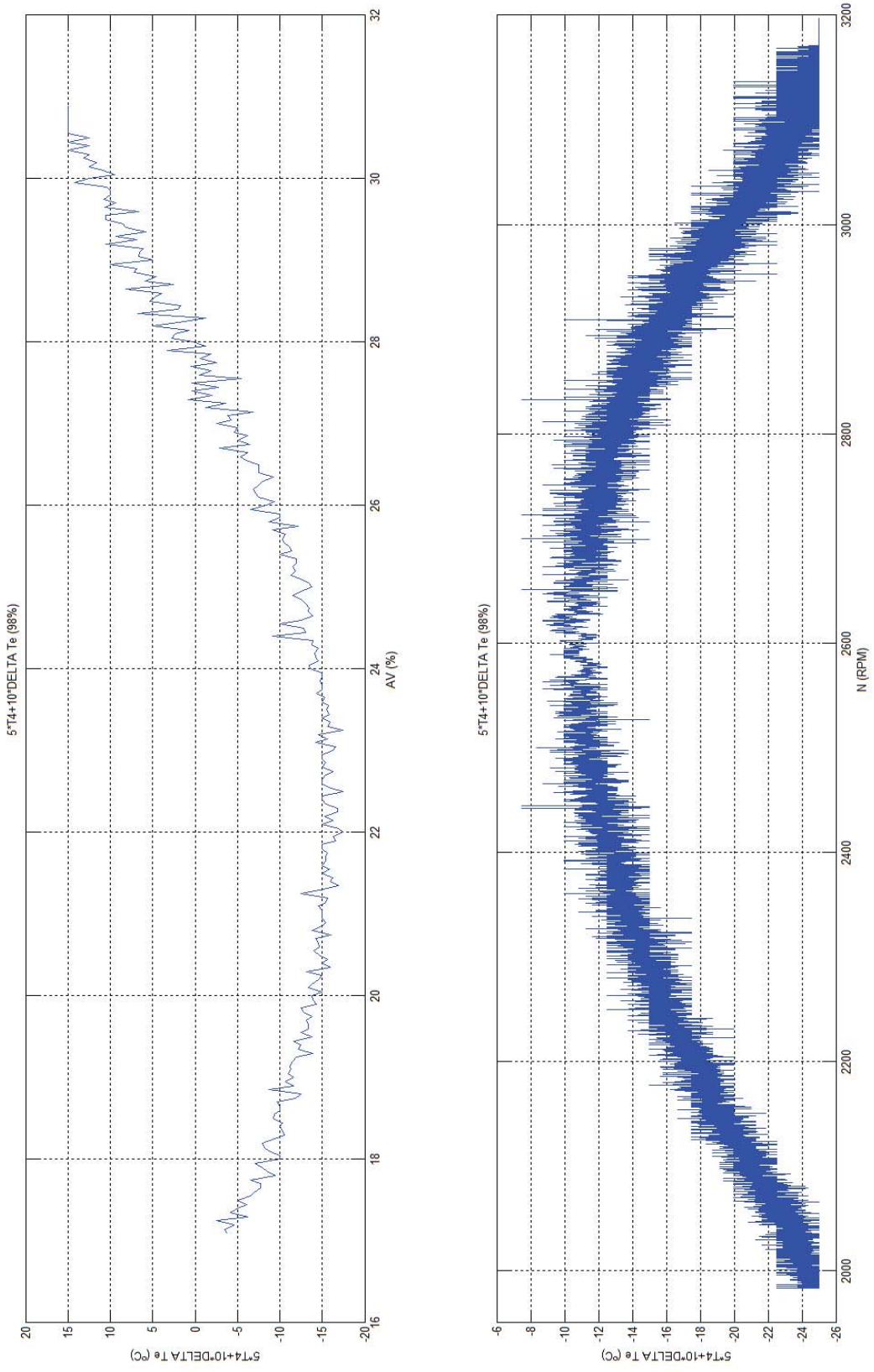


Figura B.54: $f_2(T_4, \Delta T_e) = 5T_4 + 10\Delta T_e$ *(98%)*

Apéndice C

C. CÓDIGO MATLAB

C.1. RUTINA PARA EL ANÁLISIS GLOBAL

```
% ------------------------------------------------------------------------
%
%           Programa para la obtención de los puntos de trabajo
%          factibles para una familia de ciclos de refrigeración
%
% ------------------------------------------------------------------------

% Inicializa el entorno
tic;clc;clear;

% ---------------------- PARÁMETROS PARA EL CÁLCULO -------------------

% Indica si debe generar los datos u obtenerlos del fichero
BoolPtosOp_Generar=false;

% Indica si debe ajustar los datos al grid u obtenerlos del fichero
BoolPtosOp_AjustarGrid=false;

% Indica si debe realizar la selección de los puntos de trabajo
BoolPtosOp_Seleccion=true;

% Fija el nombre de los ficheros de datos
Fichero_Matlab_Ptos='../Datos/5PTOS/Datos_5Ptos.mat';
Fichero_Matlab_Grid='../Datos/5PTOS/Datos_5Ptos_Grid.mat';
Fichero_Excel_Ptos='../Datos/5PTOS/PtosOp_5Ptos.xlsx';


% PARÁMETROS PARA LA OBTENCIÓN DE LOS PUNTOS DE TRABAJO
% -----------------------------------------------------

% Fluido y referencia termodinámica a utilizar
Parametros.Fluido='R134A';
Parametros.Referencia='DEF';

% Parametros del compresor
Parametros.Param_Compresor.Tipo=1;
Parametros.Param_Compresor.Deltak=3.042*10^-5;

% Parametros de la válvula
Parametros.Param_Valvula.Tipo=1;

% Barrido a realizar en RPM
Parametros.Barrido_RPM.Min=0; Parametros.Barrido_RPM.Max=10000;
Parametros.Barrido_RPM.Paso_Porc=0;
Parametros.Barrido_RPM.Paso_Abs=0.01;

% Barrido a realizar en Av
Parametros.Barrido_Av.Min=0;Parametros.Barrido_Av.Max=100;
Parametros.Barrido_Av.Paso_Porc=0;
Parametros.Barrido_Av.Paso_Abs=0.05;


% Restricciones para el caudal
Parametros.Restricciones_Caudal.Min=0;
Parametros.Restricciones_Caudal.Tolerancia=10^-6;

% Punto de trabajo  nominal
Parametros.PuntoTrabajo.TL0=-5;
Parametros.PuntoTrabajo.TH0=20;
Parametros.PuntoTrabajo.Delta_Te0=5;
Parametros.PuntoTrabajo.Delta_Tc0=10;
Parametros.PuntoTrabajo.Delta_Tsh0=4;
Parametros.PuntoTrabajo.Delta_Tsc0=5;

% Rango para el barrido de las temperaturas Delta_Te0, Delta_Tc0, Delta_Tsc0 y Delta_Tsh0
```

```
Parametros.Barrido_DeltaTemp.Rango=5;
Parametros.Barrido_DeltaTemp.NPuntos=5;

% Rango para el barrido de TH
Parametros.Barrido_TH.Rango=0;
Parametros.Barrido_TH.NPuntos=1;


% PARÁMETROS PARA EL AJUSTE DE LOS PUNTOS DE TRABAJO AL GRID
% ----------------------------------------------------------

% Paso para el GRID
Parametros_Grid.Paso.RPM_Porc=0;  % Paso en % Sobre el rango de barrido
Parametros_Grid.Paso.RPM_Abs=0.2; % Paso en valor absoluto
Parametros_Grid.Paso.Av_Porc=0;   % Paso en % sobre el rango de barrido
Parametros_Grid.Paso.Av_Abs=0.2;  % Paso en valor absoluto

% Indica el ciclo que debe ajustarse al grid (0 indica todos)
Parametros_Grid.ID_Ciclo=0;

% Fija las variables que se quieren discretizar junto con umbral del
% porcentaje de error
Parametros_Grid.Variables.m=true;Parametros_Grid.Variables.m_PorcentajeMax=6.66;
Parametros_Grid.Variables.P1=true;Parametros_Grid.Variables.P1_PorcentajeMax=66.67;
Parametros_Grid.Variables.P2=true;Parametros_Grid.Variables.P2_PorcentajeMax=66.68;
Parametros_Grid.Variables.Qe=true;Parametros_Grid.Variables.Qe_PorcentajeMax=9.14;
Parametros_Grid.Variables.W=true;Parametros_Grid.Variables.W_PorcentajeMax=5.39;
Parametros_Grid.Variables.nCarnot=true;Parametros_Grid.Variables.nCarnot_PorcentajeMax=44.31;
Parametros_Grid.Variables.COP=true;Parametros_Grid.Variables.COP_PorcentajeMax=45.62;
Parametros_Grid.Variables.COPc=true;Parametros_Grid.Variables.COPc_PorcentajeMax=54.04;
Parametros_Grid.Variables.T1=true;Parametros_Grid.Variables.T1_PorcentajeMax=54.17;
Parametros_Grid.Variables.T2=true;Parametros_Grid.Variables.T2_PorcentajeMax=44.18;
Parametros_Grid.Variables.T3=true;Parametros_Grid.Variables.T3_PorcentajeMax=54.18;
Parametros_Grid.Variables.T4=true;Parametros_Grid.Variables.T4_PorcentajeMax=66.68;
Parametros_Grid.Variables.DeltaTe=true;Parametros_Grid.Variables.DeltaTe_PorcentajeMax=66.68;
Parametros_Grid.Variables.DeltaTc=true;Parametros_Grid.Variables.DeltaTc_PorcentajeMax=66.68;
Parametros_Grid.Variables.DeltaTsh=true;Parametros_Grid.Variables.DeltaTsh_PorcentajeMax=66.68;
Parametros_Grid.Variables.DeltaTsc=true;Parametros_Grid.Variables.DeltaTsc_PorcentajeMax=66.68;
Parametros_Grid.Variables.TH=true;Parametros_Grid.Variables.TH_PorcentajeMax=5;


%                 PARÁMETROS PARA LA SELECCIÓN DE LOS PUNTOS DE TRABAJO
% ------------------------------------------------------------------------------------------

% Fija los puntos de operación (% QEmax) a buscar y los porcentajes de variación
% máximos sobre los mismos
PtosEstudio(1).Porcentaje=50; PtosEstudio(1).Variacion=0.04; PtosEstudio(1).NPuntosMax=5;
PtosEstudio(2).Porcentaje=90; PtosEstudio(2).Variacion=0.04; PtosEstudio(2).NPuntosMax=5;
PtosEstudio(3).Porcentaje=98; PtosEstudio(3).Variacion=0.04; PtosEstudio(3).NPuntosMax=5;
Bool_Modo_Ascendente=false;

% Fija que gráfica se mostrará de Qe y sus parámetros
Graficas.Qe.Surf=true;
Graficas.Qe.Contour=false;
Graficas.Qe.PtoOp=0;
Graficas.Qe.Rango_RPM=[0 0];
Graficas.Qe.Rango_Av=[0 0];
Graficas.Qe.RangoVariable=[0 1537];
Graficas.Qe.AzEl=[-47.5 74];%Graficas.Qe.AzEl=[-219 70];
Graficas.Qe.Anotacion.YMin=0.4;%0.65;
Graficas.Qe.Anotacion.YDelta=0.2; % 0.06

% Fija que gráfica se mostrará de We y sus parámetros
Graficas.We.Surf=false;
Graficas.We.Contour=false;
Graficas.We.Rango_RPM=[0 0];
Graficas.We.Rango_Av=[0 0];
Graficas.We.RangoVariable=[0 260];
Graficas.We.AzEl=[-47.5 74];


% Fija que gráfica se mostrará de nCarnot y sus parámetros
Graficas.nCarnot.Surf=false;
Graficas.nCarnot.Contour=false;
Graficas.nCarnot.PtoOp=0;
Graficas.nCarnot.Rango_RPM=[0 0];
Graficas.nCarnot.Rango_Av=[0 0];
Graficas.nCarnot.RangoVariable=[0.8191 0.9015];
Graficas.nCarnot.AzEl=[-47.5 74];
Graficas.nCarnot.Anotacion.YMin=0.3;
Graficas.nCarnot.Anotacion.YDelta=0.06;
```

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

% Inicialización
addpath('C:\Archivos de Programa\MATLAB\CoolProp');
addpath('..\Termodinamica');
addpath('..\Modelos');
addpath('..\Varios');


% Si hay que buscar los puntos de operación válidos
if BoolPtosOp_Generar

   % Obtiene los puntros de trabajo para los ciclos indicados
   [Ciclos,Tiempo_CPU]= Obt_PtosOp_Entorno_Ciclo (Parametros,Fichero_Matlab_Ptos,...
                                                  Fichero_Excel_Ptos);

else

   % Carga el fichero de datos Matlab con los puntos de operación
   if ~isempty(Fichero_Matlab_Ptos)
      load (Fichero_Matlab_Ptos);
   end

end


% Si hay que ajustar los puntos al grid
if BoolPtosOp_AjustarGrid

   % Ejecuta la rutina de discretización
   [Vector_RPM,Vector_Av,Datos_Grid]=Grid_Ajustar_PtosOp_Ciclos_RPMAV (Fichero_Matlab_Grid,...
                                                                       Parametros,...
                                                                       Parametros_Grid,...
                                                                       Ciclos);
else

   % Carga el fichero de datos Matlab con los puntos ajustados al grid
   if ~isempty(Fichero_Matlab_Grid)
      load (Fichero_Matlab_Grid);
   end

end


% Si hay que seleccionar los datos
if BoolPtosOp_Seleccion
   Sel_PtosOP_Ciclos (Ciclos,Datos_Grid,Vector_RPM,Vector_Av,Graficas,PtosEstudio,...
                      Bool_Modo_Ascendente)
end
```

C.2. RUTINA PARA EL ANÁLISIS DE LOS PUNTOS DE TRABAJO

```
% ------------------------------------------------------------------------
%
%           Programa para el análisis de puntos de trabajo
%
% ------------------------------------------------------------------------

% Inicializa el entorno
clc;clear;

% ---------------------- PARÁMETROS PARA EL CÁLCULO --------------------

% Indica si debe generar los datos u obtenerlos del fichero
BoolPtosOp_Generar=false;

% Indica si debe ajustar los datos al grid u obtenerlos del fichero
BoolPtosOp_AjustarGrid=true;

% Indica si debe realizar la selección de los puntos de trabajo
BoolPtosOp_ProcesarGrid=true;


%                  PARÁMETROS PARA LA OBTENCIÓN DE LOS PUNTOS DE TRABAJO
% ------------------------------------------------------------------------------------------------

% Fija el nombre de los ficheros de datos
%P50
Parametros_General(1).Fichero_Matlab_RPM0='../Datos/P50/Datos_P50_RPM0.mat';
Parametros_General(1).Fichero_Matlab_AV0='../Datos/P50/Datos_P50_AV0.mat';
Parametros_General(1).Fichero_Matlab_TH='../Datos//P50/Datos_P50_TH.mat';
Parametros_General(1).Fichero_Excel_RPM0='../Datos//P50/Datos_P50_RPM0.xlsx';
Parametros_General(1).Fichero_Excel_AV0='../Datos/P50/Datos_P50_AV0.xlsx';
Parametros_General(1).Fichero_Excel_TH='../Datos/P50/Datos_P50_TH.xlsx';

%P90
Parametros_General(2).Fichero_Matlab_RPM0='../Datos/P90/Datos_P90_RPM0.mat';
Parametros_General(2).Fichero_Matlab_AV0='../Datos/P90/Datos_P90_AV0.mat';
Parametros_General(2).Fichero_Matlab_TH='../Datos//P90/Datos_P90_TH.mat';
Parametros_General(2).Fichero_Excel_RPM0='../Datos//P90/Datos_P90_RPM0.xlsx';
Parametros_General(2).Fichero_Excel_AV0='../Datos/P90/Datos_P90_AV0.xlsx';
Parametros_General(2).Fichero_Excel_TH='../Datos/P90/Datos_P90_TH.xlsx';

%P98
Parametros_General(3).Fichero_Matlab_RPM0='../Datos/P98/Datos_P98_RPM0.mat';
Parametros_General(3).Fichero_Matlab_AV0='../Datos/P98/Datos_P98_AV0.mat';
Parametros_General(3).Fichero_Matlab_TH='../Datos//P98/Datos_P98_TH.mat';
Parametros_General(3).Fichero_Excel_RPM0='../Datos//P98/Datos_P98_RPM0.xlsx';
Parametros_General(3).Fichero_Excel_AV0='../Datos/P98/Datos_P98_AV0.xlsx';
Parametros_General(3).Fichero_Excel_TH='../Datos/P98/Datos_P98_TH.xlsx';


% Fluido y referencia termodinámica a utilizar
%P50
Parametros_General(1).Fluido='R134A';
Parametros_General(1).Referencia='DEF';

%P90
Parametros_General(2).Fluido='R134A';
Parametros_General(2).Referencia='DEF';

%P98
Parametros_General(3).Fluido='R134A';
Parametros_General(3).Referencia='DEF';

% Parametros del compresor
%P50
Parametros_General(1).Param_Compresor.Tipo=1;              % Modelo a emplear para el compresor
Parametros_General(1).Param_Compresor.Deltak=3.042*10^-5;  % Volumen del compresor

%P90
Parametros_General(2).Param_Compresor.Tipo=1;              % Modelo a emplear para el compresor
Parametros_General(2).Param_Compresor.Deltak=3.042*10^-5;  % Volumen del compresor

%P98
Parametros_General(3).Param_Compresor.Tipo=1;              % Modelo a emplear para el compresor
Parametros_General(3).Param_Compresor.Deltak=3.042*10^-5;  % Volumen del compresor

% Parametros de la válvula
%P50
```

```
Parametros_General(1).Param_Valvula.Tipo=1;                 % Modelo a emplear para la válvula

%P90
Parametros_General(2).Param_Valvula.Tipo=1;                 % Modelo a emplear para la válvula

%P98
Parametros_General(3).Param_Valvula.Tipo=1;                 % Modelo a emplear para la válvula


% Barrido a realizar en RPM y Av para la obtención de los puntos de trabajo válidos.
% Para el paso, utilizar uno de ellos y mantener a cero el otro.
%P50
Parametros_General(1).Barrido_RPM.RPM0=3696.33;             % Punto de trabajo nominal para RPM
Parametros_General(1).Barrido_RPM.Rango_Porc=30;            % ±(%) sobre RPM0
Parametros_General(1).Limites_RPM.Inf=0;                    % Límite inferior máximo para RPM
Parametros_General(1).Limites_RPM.Sup=4200;                 % Límite superior máximo para RPM
Parametros_General(1).Barrido_RPM.Paso_Porc=0;              % Paso en % Sobre el rango indicado
Parametros_General(1).Barrido_RPM.Paso_Abs=0.05;            % Paso en valor absoluto

Parametros_General(1).Barrido_Av.Av0=34.70;                 % Punto de trabajo nominal para Av
Parametros_General(1).Barrido_Av.Rango_Porc=30;             % ±(%) sobre AvM0
Parametros_General(1).Limites_Av.Inf=0;                     % Límite inferior máximo
Parametros_General(1).Limites_Av.Sup=100;                   % Límite superior máximo
Parametros_General(1).Barrido_Av.Paso_Porc=0;               % Paso en % Sobre el rango indicado
Parametros_General(1).Barrido_Av.Paso_Abs=0.05;             % Paso en valor absoluto

%P90
Parametros_General(2).Barrido_RPM.RPM0=2987.29;             % Punto de trabajo nominal para RPM
Parametros_General(2).Barrido_RPM.Rango_Porc=30;            % ±(%) sobre RPM0
Parametros_General(2).Limites_RPM.Inf=0;                    % Límite inferior máximo para RPM
Parametros_General(2).Limites_RPM.Sup=4200;                 % Límite superior máximo para RPM
Parametros_General(2).Barrido_RPM.Paso_Porc=0;              % Paso en % Sobre el rango indicado
Parametros_General(2).Barrido_RPM.Paso_Abs=0.05;            % Paso en valor absoluto

Parametros_General(2).Barrido_Av.Av0=28.05;                 % Punto de trabajo nominal para Av
Parametros_General(2).Barrido_Av.Rango_Porc=30;             % ±(%) sobre AvM0
Parametros_General(2).Limites_Av.Inf=0;                     % Límite inferior máximo
Parametros_General(2).Limites_Av.Sup=100;                   % Límite superior máximo
Parametros_General(2).Barrido_Av.Paso_Porc=0;               % Paso en % Sobre el rango indicado
Parametros_General(2).Barrido_Av.Paso_Abs=0.05;             % Paso en valor absoluto


%P98
Parametros_General(3).Barrido_RPM.RPM0=2832.75;             % Punto de trabajo nominal para RPM
Parametros_General(3).Barrido_RPM.Rango_Porc=30;            % ±(%) sobre RPM0
Parametros_General(3).Limites_RPM.Inf=0;                    % Límite inferior máximo para RPM
Parametros_General(3).Limites_RPM.Sup=4200;                 % Límite superior máximo para RPM
Parametros_General(3).Barrido_RPM.Paso_Porc=0;              % Paso en % Sobre el rango indicado
Parametros_General(3).Barrido_RPM.Paso_Abs=0.05;            % Paso en valor absoluto

Parametros_General(3).Barrido_Av.Av0=24.45;                 % Punto de trabajo nominal para Av
Parametros_General(3).Barrido_Av.Rango_Porc=30;             % ±(%) sobre AvM0
Parametros_General(3).Limites_Av.Inf=0;                     % Límite inferior máximo
Parametros_General(3).Limites_Av.Sup=100;                   % Límite superior máximo
Parametros_General(3).Barrido_Av.Paso_Porc=0;               % Paso en % Sobre el rango indicado
Parametros_General(3).Barrido_Av.Paso_Abs=0.05;             % Paso en valor absoluto


% Punto de trabajo  nominal
%P50
Parametros_General(1).PuntoTrabajo.TL0=-5;                  % Temperatura del recinto
Parametros_General(1).PuntoTrabajo.TH0=20;                  % Temperatura ambiente
Parametros_General(1).PuntoTrabajo.Delta_Te0=2.5;           % Salto térmico en el evaporador
Parametros_General(1).PuntoTrabajo.Delta_Tc0=7.5;           % Salto térmico en el condensador
Parametros_General(1).PuntoTrabajo.Delta_Tsc0=7.5;          % Sobreenfriamiento
Parametros_General(1).PuntoTrabajo.Delta_Tsh0=4;            % Sobrecalentamiento

%P90
Parametros_General(2).PuntoTrabajo.TL0=-5;                  % Temperatura del recinto
Parametros_General(2).PuntoTrabajo.TH0=20;                  % Temperatura ambiente
Parametros_General(2).PuntoTrabajo.Delta_Te0=3.75;          % Salto térmico en el evaporador
Parametros_General(2).PuntoTrabajo.Delta_Tc0=7.5;           % Salto térmico en el condensador
Parametros_General(2).PuntoTrabajo.Delta_Tsc0=7.5;          % Sobreenfriamiento
Parametros_General(2).PuntoTrabajo.Delta_Tsh0=6.5;          % Sobrecalentamiento

%P98
Parametros_General(3).PuntoTrabajo.TL0=-5;                  % Temperatura del recinto
Parametros_General(3).PuntoTrabajo.TH0=20;                  % Temperatura ambiente
Parametros_General(3).PuntoTrabajo.Delta_Te0=2.5;           % Salto térmico en el evaporador
Parametros_General(3).PuntoTrabajo.Delta_Tc0=7.5;           % Salto térmico en el condensador
Parametros_General(3).PuntoTrabajo.Delta_Tsc0=7.5;          % Sobreenfriamiento
Parametros_General(3).PuntoTrabajo.Delta_Tsh0=1.5;          % Sobrecalentamiento
```

```
% Rango para el barrido de las temperaturas Delta_Te0, Delta_Tc0, Delta_Tsc0, Delta_Tsh0 y TH
%P50
Parametros_General(1).Barrido_DeltaTemp.Rango=8;
Parametros_General(1).Barrido_DeltaTemp.NPuntos=17;%17;
Parametros_General(1).Barrido_TH.Rango=6;
Parametros_General(1).Barrido_TH.NPuntos=9;

%P90
Parametros_General(2).Barrido_DeltaTemp.Rango=8;
Parametros_General(2).Barrido_DeltaTemp.NPuntos=17;%17;
Parametros_General(2).Barrido_TH.Rango=6;
Parametros_General(2).Barrido_TH.NPuntos=9;

%P98
Parametros_General(3).Barrido_DeltaTemp.Rango=8;
Parametros_General(3).Barrido_DeltaTemp.NPuntos=17;%17;
Parametros_General(3).Barrido_TH.Rango=6;
Parametros_General(3).Barrido_TH.NPuntos=9;

% Restricciones para el caudal
%P50
Parametros_General(1).Restricciones_Caudal.Min=10^-4;       % Caudal mínimo a considerar
Parametros_General(1).Restricciones_Caudal.Tolerancia=10^-4;% Diferencia (g/s) máxima caudales
                                                            % de la válvula y el del compresor
%P90
Parametros_General(2).Restricciones_Caudal.Min=10^-4;       % Caudal mínimo a considerar
Parametros_General(2).Restricciones_Caudal.Tolerancia=10^-4;% Diferencia (g/s) máxima caudales
                                                            % de la válvula y el del compresor
%P98
Parametros_General(3).Restricciones_Caudal.Min=10^-4;       % Caudal mínimo a considerar
Parametros_General(3).Restricciones_Caudal.Tolerancia=10^-4;% Diferencia (g/s) máxima caudales
                                                            % de la válvula y el del compresor



%                 PARÁMETROS PARA EL AJUSTE DE LOS PUNTOS DE TRABAJO AL GRID
% ------------------------------------------------------------------------------------------

% Fija el nombre de los ficheros de datos
%P50
Parametros_General(1).Fichero_Matlab_RPM0_Grid='../Datos/P50/Datos_P50_RPM0_Grid.mat';
Parametros_General(1).Fichero_Matlab_AV0_Grid='../Datos/P50/Datos_P50_AV0_Grid.mat';
Parametros_General(1).Fichero_Matlab_TH_Grid='../Datos/P50/Datos_P50_TH_Grid.mat';

%P90
Parametros_General(2).Fichero_Matlab_RPM0_Grid='../Datos/P90/Datos_P90_RPM0_Grid.mat';
Parametros_General(2).Fichero_Matlab_AV0_Grid='../Datos/P90/Datos_P90_AV0_Grid.mat';
Parametros_General(2).Fichero_Matlab_TH_Grid='../Datos/P90/Datos_P90_TH_Grid.mat';

%P98
Parametros_General(3).Fichero_Matlab_RPM0_Grid='../Datos/P98/Datos_P98_RPM0_Grid.mat';
Parametros_General(3).Fichero_Matlab_AV0_Grid='../Datos/P98/Datos_P98_AV0_Grid.mat';
Parametros_General(3).Fichero_Matlab_TH_Grid='../Datos/P98/Datos_P98_TH_Grid.mat';


% Paso para RPM y Av en el GRID. Fijar uno de ellos y mantener a cero el otro
Parametros_Grid.Paso.RPM_Porc=0;                        % Paso en % Sobre el rango de barrido
Parametros_Grid.Paso.RPM_Abs=0.05;                      % Paso en valor absoluto (0.05)
Parametros_Grid.Paso.Av_Porc=0;                         % Paso en % sobre el rango de barrido
Parametros_Grid.Paso.Av_Abs=0.05;                       % Paso en valor absoluto (0.05)

% Paso para TH en el GRID. Fijar uno de ellos y mantener a cero el otro
Parametros_Grid.Paso.TH_Porc=0;                         % Paso en % sobre el rango de barrido
Parametros_Grid.Paso.TH_Abs=0.75;                       % Paso en valor absoluto (0.75)

% Indica el ciclo que debe ajustarse al grid (0 indica todos)
Parametros_Grid.ID_Ciclo=0;

% Fija las variables que se quieren discretizar junto con umbral del
% porcentaje de error
Parametros_Grid.Variables.m=true;Parametros_Grid.Variables.m_PorcentajeMax=100;
Parametros_Grid.Variables.P1=true;Parametros_Grid.Variables.P1_PorcentajeMax=100;
Parametros_Grid.Variables.P2=true;Parametros_Grid.Variables.P2_PorcentajeMax=100;
Parametros_Grid.Variables.Qe=true;Parametros_Grid.Variables.Qe_PorcentajeMax=100;
Parametros_Grid.Variables.W=true;Parametros_Grid.Variables.W_PorcentajeMax=100;
Parametros_Grid.Variables.nCarnot=true;Parametros_Grid.Variables.nCarnot_PorcentajeMax=100;
Parametros_Grid.Variables.COP=true;Parametros_Grid.Variables.COP_PorcentajeMax=100;
Parametros_Grid.Variables.COPc=true;Parametros_Grid.Variables.COPc_PorcentajeMax=100;
Parametros_Grid.Variables.T1=true;Parametros_Grid.Variables.T1_PorcentajeMax=100;
Parametros_Grid.Variables.T2=true;Parametros_Grid.Variables.T2_PorcentajeMax=100;
Parametros_Grid.Variables.T3=true;Parametros_Grid.Variables.T3_PorcentajeMax=100;
Parametros_Grid.Variables.T4=true;Parametros_Grid.Variables.T4_PorcentajeMax=100;
```

```
Parametros_Grid.Variables.DeltaTe=true;Parametros_Grid.Variables.DeltaTe_PorcentajeMax=100;
Parametros_Grid.Variables.DeltaTc=true;Parametros_Grid.Variables.DeltaTc_PorcentajeMax=100;
Parametros_Grid.Variables.DeltaTsh=true;Parametros_Grid.Variables.DeltaTsh_PorcentajeMax=100;
Parametros_Grid.Variables.DeltaTsc=true;Parametros_Grid.Variables.DeltaTsc_PorcentajeMax=100;
Parametros_Grid.Variables.TH=true;Parametros_Grid.Variables.TH_PorcentajeMax=100;



%                 PARÁMETROS PARA EL ANÁLISIS DE LOS PUNTOS DE TRABAJO DEL GRID
% ------------------------------------------------------------------------------------------

% Porcentaje para el cálculo de la matriz de ganancias
Parametros_Analisis.PorcentajeK=5;                    % ±(%) sobre RPM0 y Av0 para el cálculo
                                                      % de la matriz de ganancias

Parametros_Analisis.Variables_Calculadas.Aux1.BoolActivar=true;
Parametros_Analisis.Variables_Calculadas.Aux1.Coef=[5 10];% 10 5
Parametros_Analisis.Variables_Calculadas.Aux1.Nombre={'T4','DeltaTe'};

Parametros_Analisis.Variables_Calculadas.Aux2.BoolActivar=true;
Parametros_Analisis.Variables_Calculadas.Aux2.Coef=[5 10];
Parametros_Analisis.Variables_Calculadas.Aux2.Nombre={'T3','DeltaTc'};

Parametros_Analisis.Variables_Calculadas.Aux3.BoolActivar=false;
Parametros_Analisis.Variables_Calculadas.Aux3.Coef=[1 1];
Parametros_Analisis.Variables_Calculadas.Aux3.Nombre={'P1','P2'};


% Fichero en donde se guardarán los datos procesados
Parametros_Analisis.Fichero_Matlab_Procesado='../Datos/Procesado/Datos_Procesados.mat';


% Inicialización
addpath('C:\Archivos de Programa\MATLAB\CoolProp');
addpath('..\Termodinamica');
addpath('..\Modelos');
addpath('..\Varios');


% Realiza el barrido para cada uno de los puntos de trabajo seleccionados
for i=1:length(Parametros_General)

   % Si hay que buscar los puntos de operación válidos
   if BoolPtosOp_Generar

      % ------------ BARRIDO DE AV MANTENIENDO RPM EN EL PUNTO DE TRABAJO ------------
      % ------------------------------------------------------------------------------

      % Obtiene los parámetros
      Param=Parametros_General(i);

      % Obtiene el barrido a realizar en Av
      Min=(1-0.01*Param.Barrido_Av.Rango_Porc)*Param.Barrido_Av.Av0;
      Max=(1+0.01*Param.Barrido_Av.Rango_Porc)*Param.Barrido_Av.Av0;
      if Min<Param.Limites_Av.Inf
         Min=Param.Limites_Av.Inf;
      end

      if Max>Param.Limites_Av.Sup
         Max=Param.Limites_Av.Sup;
      end

      Param.Barrido_Av.Min=Min; Param.Barrido_Av.Max=Max;

      % Obtiene el barrido a realizar en RPM
      RPM0=Param.Barrido_RPM.RPM0;
      Param.Barrido_RPM.Min=RPM0; Param.Barrido_RPM.Max=RPM0;
      Param.Barrido_RPM.Paso_Porc=0;
      Param.Barrido_RPM.Paso_Abs=1;

      % Fija TH a TH0
      Param.Barrido_TH.Rango=0;Param.Barrido_TH.NPuntos=1;

      fprintf ('OBTENIENDO PUNTOS PARA RPM=RPM0...\n\n');

      % Obtiene los puntros de trabajo manteniendo RPM=RPM0 y variando Av
      [Ciclos_Brutos(i).RPM0,Ciclos_Brutos(i).RPM0.Tiempo_CPU]= ...
                              Obt_PtosOp_Entorno_Ciclo (Param,Param.Fichero_Matlab_RPM0,...
                                                        Param.Fichero_Excel_RPM0);


      % ------------ BARRIDO DE RPM MANTENIENDO AV EN EL PUNTO DE TRABAJO ------------
```

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

    % Obtiene nuevamente los parámetros
    Param=Parametros_General(i);

    % Obtiene el barrido a realizar en RPM
    Min=(1-0.01*Param.Barrido_RPM.Rango_Porc)*Param.Barrido_RPM.RPM0;
    Max=(1+0.01*Param.Barrido_RPM.Rango_Porc)*Param.Barrido_RPM.RPM0;

    if Min<Param.Limites_RPM.Inf
       Min=Param.Limites_RPM.Inf;
    end

    if Max>Param.Limites_RPM.Sup
       Max=Param.Limites_RPM.Sup;
    end

    Param.Barrido_RPM.Min=Min; Param.Barrido_RPM.Max=Max;

    % Obtiene el barrido a realizar en Av
    AV0=Param.Barrido_Av.Av0;
    Param.Barrido_Av.Min=AV0; Param.Barrido_Av.Max=AV0;
    Param.Barrido_Av.Paso_Porc=0;
    Param.Barrido_Av.Paso_Abs=1;

    % Fija TH a TH0
    Param.Barrido_TH.Rango=0;Param.Barrido_TH.NPuntos=1;

    fprintf ('\n\nOBTENIENDO PUNTOS PARA AV=AV0...\n\n');

    % Obtiene los puntros de trabajo manteniendo AV=AV0 y variando RPM
    [Ciclos_Brutos(i).AV0,Ciclos_Brutos(i).AV0.Tiempo_CPU]= ...
                              Obt_PtosOp_Entorno_Ciclo (Param,Param.Fichero_Matlab_AV0,...
                                                        Param.Fichero_Excel_AV0);



    % --------- BARRIDO DE TH MANTENIENDO RPM Y AV EN EL PUNTO DE TRABAJO ----------
    % ------------------------------------------------------------------------------

    % Obtiene nuevamente los parámetros
    Param=Parametros_General(i);

    % Barrido a realizar en RPM
    RPM0=Param.Barrido_RPM.RPM0;
    Param.Barrido_RPM.Min=RPM0; Param.Barrido_RPM.Max=RPM0;
    Param.Barrido_RPM.Paso_Porc=0;
    Param.Barrido_RPM.Paso_Abs=1;

    % Barrido a realizar en Av
    AV0=Param.Barrido_Av.Av0;
    Param.Barrido_Av.Min=AV0;Param.Barrido_Av.Max=AV0;
    Param.Barrido_Av.Paso_Porc=0;
    Param.Barrido_Av.Paso_Abs=1;

    fprintf ('\n\nOBTENIENDO PUNTOS PARA TH...\n\n');

    % Obtiene los puntros de trabajo para los ciclos indicados
    [Ciclos_Brutos(i).TH,Ciclos_Brutos(i).TH.Tiempo_CPU]= ...
                    Obt_PtosOp_Entorno_Ciclo (Param,Param.Fichero_Matlab_TH,...
                                              Param.Fichero_Excel_TH);


else

    % Carga el fichero de datos Matlab con los puntos de operación para RPM=RPM0
    if ~isempty(Parametros_General(i).Fichero_Matlab_RPM0)

       try

          load (Parametros_General(i).Fichero_Matlab_RPM0);
          Ciclos_Brutos(i).RPM0=Ciclos; Ciclos_Brutos(i).RPM0.Tiempo_CPU=Tiempo_CPU;

       catch

          Ciclos_Brutos(i).RPM0.PtosOp_Q=[];

       end

    end

    % Carga el fichero de datos Matlab con los puntos de operación para RPM=AV0
    if ~isempty(Parametros_General(i).Fichero_Matlab_AV0)
```

```
         try

            load (Parametros_General(i).Fichero_Matlab_AV0);
            Ciclos_Brutos(i).AV0=Ciclos; Ciclos_Brutos(i).AV0.Tiempo_CPU=Tiempo_CPU;

         catch

            Ciclos_Brutos(i).AV0.PtosOp_Q=[];

         end

      end

      % Carga el fichero de datos Matlab con los puntos de operación para TH
      if ~isempty(Parametros_General(i).Fichero_Matlab_TH)

         try

            load (Parametros_General(i).Fichero_Matlab_TH);
            Ciclos_Brutos(i).TH=Ciclos; Ciclos_Brutos(i).TH.Tiempo_CPU=Tiempo_CPU;

         catch

             Ciclos_Brutos(i).TH.PtosOp_Q=[];

         end

      end

      clear ('Ciclos','Tiempo_CPU');

   end

end


% Ajusta el grid para cada uno de los puntos de trabajo seleccionados
for i=1:length(Parametros_General)

   % Si hay que ajustar los puntos al grid
   if BoolPtosOp_AjustarGrid

      fprintf ('\n\nAJUSTE DE VALORES AL GRID...\n\n');

      % Ejecuta la rutina de discretización para RPM0
      if length(Ciclos_Brutos(i).RPM0.PtosOp_Q)>0

         fprintf ('Ajustando valores al GRID para RPM0...\n\n');
         [Ciclos_Grid(i).RPM0.Vector_RPM,Ciclos_Grid(i).RPM0.Vector_Av,...
          Ciclos_Grid(i).RPM0.Datos]=...
          Grid_Ajustar_PtosOp_Ciclos_RPMAV (Parametros_General(i).Fichero_Matlab_RPM0_Grid, ...
                                            Parametros_General(i),Parametros_Grid,...
                                            Ciclos_Brutos(i).RPM0);

      end

      % Ejecuta la rutina de discretización para AV0
      if length(Ciclos_Brutos(i).AV0.PtosOp_Q)>0

         fprintf ('\nAjustando valores al GRID para AV0...\n\n');
         [Ciclos_Grid(i).AV0.Vector_RPM,Ciclos_Grid(i).AV0.Vector_Av,...
          Ciclos_Grid(i).AV0.Datos]=...
          Grid_Ajustar_PtosOp_Ciclos_RPMAV (Parametros_General(i).Fichero_Matlab_AV0_Grid, ...
                                            Parametros_General(i),Parametros_Grid,...
                                            Ciclos_Brutos(i).AV0);

      end

      % Ejecuta la rutina de discretización para TH (RPM0-AV0)
      if length(Ciclos_Brutos(i).TH.PtosOp_Q)>0

         fprintf ('\nAjustando valores al GRID para TH...\n\n');
         [Ciclos_Grid(i).TH.Vector, Ciclos_Grid(i).TH.Datos]=...
            Vector_Ajustar_PtosOp_Ciclos_TH (Parametros_General(i).Fichero_Matlab_TH_Grid, ...
                                             Parametros_Grid,...
                                             Ciclos_Brutos(i).TH);

      end

      fprintf ('\nPROCESO DE AJUSTE DE VALORES AL GRID FINALIZADO CORRECTAMENTE.\n\n');

   else
```

```
      % Carga el fichero de datos Matlab con los puntos de operación para RPM=RPM0
      if ~isempty(Parametros_General(i).Fichero_Matlab_RPM0_Grid)

         try

            load (Parametros_General(i).Fichero_Matlab_RPM0_Grid);
            Ciclos_Grid(i).RPM0.Vector_RPM=Vector_RPM;
            Ciclos_Grid(i).RPM0.Vector_Av=Vector_Av;
            Ciclos_Grid(i).RPM0.Datos=Datos_Grid;

         catch

            Ciclos_Grid(i).RPM0.Vector_RPM=[];
            Ciclos_Grid(i).RPM0.Vector_Av=[];
            Ciclos_Grid(i).RPM0.Datos=[];

         end

      end

      % Carga el fichero de datos Matlab con los puntos de operación para RPM=AV0
      if ~isempty(Parametros_General(i).Fichero_Matlab_AV0_Grid)

         try

            load (Parametros_General(i).Fichero_Matlab_AV0_Grid);
            Ciclos_Grid(i).AV0.Vector_RPM=Vector_RPM;
            Ciclos_Grid(i).AV0.Vector_Av=Vector_Av;
            Ciclos_Grid(i).AV0.Datos=Datos_Grid;

         catch

            Ciclos_Grid(i).AV0.Vector_RPM=[];
            Ciclos_Grid(i).AV0.Vector_Av=[];
            Ciclos_Grid(i).AV0.Datos=[];

         end

      end

      % Carga el fichero de datos Matlab con los puntos de operación para TH
      if ~isempty(Parametros_General(i).Fichero_Matlab_TH_Grid)

         try

            load (Parametros_General(i).Fichero_Matlab_TH_Grid);
            Ciclos_Grid(i).TH.Vector=Vector_TH;
            Ciclos_Grid(i).TH.Datos=Datos_Vector;

         catch

            Ciclos_Grid(i).TH.Vector=[];
            Ciclos_Grid(i).TH.Datos=[];

         end

      end

      clear ('Ciclos','Datos_Grid','Datos_Vector','Vector_RPM','Vector_Av','Vector_TH')

    end

end


% Si hay que procesar el grid
if BoolPtosOp_ProcesarGrid

   Ciclos_Procesados=[];


   % Procesa los datos del grid para cada uno de los puntos de trabajo seleccionados
   for i=1:length(Parametros_General)

      fprintf ('\n\nPROCESANDO GRID...\n\n');

      % Procesa los datos para el barrido de Av sobre RPM0
      if length(Ciclos_Brutos(i).RPM0.PtosOp_Q)>0

         fprintf ('Procesando GRID para RPM0...\n');
         [Ciclos_Procesados(i).RPM0]=...
                 Procesar_Variables_Ciclos_RPMAV(Ciclos_Grid(i).RPM0,Parametros_General(i),...
```

```
                                                Parametros_Grid,Parametros_Analisis,true,...
                                                Ciclos_Brutos(i).RPM0.Tiempo_CPU);
      end



      % Procesa los datos para el barrido de RPM sobre Av
      if length(Ciclos_Brutos(i).AV0.PtosOp_Q)>0

         fprintf ('Procesando GRID para AV0...\n');
         [Ciclos_Procesados(i).AV0]=...
                 Procesar_Variables_Ciclos_RPMAV(Ciclos_Grid(i).AV0,Parametros_General(i),...
                                                 Parametros_Grid,Parametros_Analisis,false,...
                                                 Ciclos_Brutos(i).AV0.Tiempo_CPU);


      end


      % Procesa los datos para el barrido de TH con RPM0,AV0
      if length(Ciclos_Brutos(i).TH.PtosOp_Q)>0

         fprintf ('Procesando GRID para TH...\n');


         [Ciclos_Procesados(i).TH]=...
                   Procesar_Variables_Ciclos_TH(Ciclos_Grid(i).TH,Parametros_General(i),...
                                                Parametros_Grid,Parametros_Analisis,...
                                                Ciclos_Brutos(i).TH);

      end


      % Calcula SSGM y GRGA
      if (length(Ciclos_Brutos(i).RPM0.PtosOp_Q)>0) && ...
         (length(Ciclos_Brutos(i).AV0.PtosOp_Q)>0)

         fprintf ('Procesando SSGM y GRGA...\n\n');
         [Ciclos_Procesados(i).Controladas,Ciclos_Procesados(i).Manipulables,...
          Ciclos_Procesados(i).SSGM,Ciclos_Procesados(i).GRGA]=...
                         Procesar_SSGM_GRGA(Ciclos_Procesados(i),Parametros_Grid.Variables,...
                                            Parametros_Analisis.Variables_Calculadas);

         fprintf ('Variables controladas\n\n');
         disp(Ciclos_Procesados(i).Controladas');
         fprintf('Variables manipulables\n\n');
         disp(Ciclos_Procesados(i).Manipulables');
         fprintf('SSGM\n\n');
         disp(Ciclos_Procesados(i).SSGM)
         fprintf('GRGA\n\n');
         disp(Ciclos_Procesados(i).GRGA)

      end

      fprintf ('\nPROCESO DEL GRID FINALIZADO CORRECTAMENTE.\n');

   end

   % Guarda los resultados
   if ~isempty(Parametros_Analisis.Fichero_Matlab_Procesado)
      delete (Parametros_Analisis.Fichero_Matlab_Procesado);
      save (Parametros_Analisis.Fichero_Matlab_Procesado,'Ciclos_Procesados');
   end

else

   % Carga el fichero de datos Matlab con los datos procesados
   if ~isempty(Parametros_Analisis.Fichero_Matlab_Procesado)
      load (Parametros_Analisis.Fichero_Matlab_Procesado);
   end
end
```

C.3. MODELOS DE COMPRESOR Y VÁLVULA

```
% ------------------------------------------------------------------------
%        Función para obtener el caudal en kg/s que atraviesa el
%        compresor utilizando el modelo especificado
%
%       VARIABLES DE ENTRADA
%
%       Fluido:             Alguno de los soportados por CoolProp
%       Param_Compresor:    Estructura con sus parámetros:
%           Tipo:           Tipo de modelo. El único utilizado es el Tipo=1 (Neera-Jain)
%           Delta_Tsh:      Sobrecalentamiento (°C)
%           Deltak:         Volumen del compresor (m3)
%       RPM:                Vueltas por minuto a las que esta operando.
%       P1:                 Presión más baja (bares) del ciclo (evaporador)
%       P2:                 Presión más alta (bares) del ciclo (condensador)
%
%       VARIABLES DE SALIDA
%
%       m:                  Caudal en kg/s
% ------------------------------------------------------------------------

function m=Obt_Q_Compresor (Fluido,Param_Compresor,RPM,P1,P2)

   % Modelo de Jain
   if Param_Compresor.Tipo==1
      m=Obt_Q_Compresor_Jain (Fluido,Param_Compresor.Delta_Tsh,Param_Compresor.Deltak,...
                              RPM,P1,P2);
   end

end
```

```
% ------------------------------------------------------------------------------------------
%     Función para obtener el caudal en kg/s que atraviesa la válvula
%              de expansión utilizando el modelo especificado
%
%       VARIABLES DE ENTRADA
%
%       Fluido:             Uno de los soportados por el CoolProp
%       Param_Valvula:      Estructura con sus parámetros:
%           Tipo:           Tipo de modelo. El único utilizado es el Tipo=1 (Neera-Jain)
%           Delta_Tsc:      Sobreenfriamiento (°C)
%       Posicion:           Porcentaje apertura (0-100%)
%       P1:                 Presión más baja (bares) del ciclo (evaporador)
%       P2:                 Presión más alta (bares) del ciclo (condensador)
%
%       VARIABLES DE SALIDA
%
%       m:                  Caudal en kg/s
% ------------------------------------------------------------------------------------------

function m=Obt_Q_Valvula (Fluido,Param_Valvula,Posicion,P1,P2)

   % Modelo de Jain
   if Param_Valvula.Tipo==1
      m=Obt_Q_Valvula_Jain (Fluido,Param_Valvula.Delta_Tsc,Posicion,P1,P2);
   end

end
```

```
% ------------------------------------------------------------------------
%        Función para obtener el caudal en kg/s que atraviesa el
%        compresor utilizando el modelo propuesto por Neera Jain
%
%       VARIABLES DE ENTRADA
%
%       Fluido:         Alguno de los soportados por CoolProp
%       Delta_Tsh:      Sobrecalentamiento (°C)
%       Deltak:         Volumen del compresor (m3)
%       RPM:            Vueltas por minuto a las que esta operando.
%       P1:             Presión más baja (bares) del ciclo (evaporador)
%       P2:             Presión más alta (bares) del ciclo (condensador)
%
%       VARIABLES DE SALIDA
%
%       m:              Caudal en kg/s
% ------------------------------------------------------------------------

function m=Obt_Q_Compresor_Jain (Fluido,Delta_Tsh,Deltak,RPM,P1,P2)

  % Obtiene las presiones (bares) a la entrada y salida del compresor
   Pin=P1;
   Pout=P2;

   % Obtiene los parámetros termodinámicos de saturación a P=Pin (bares)
   Estado_Saturacion=Obt_Param_Zona_Saturacion (Fluido,'P',Pin);

   % Obtiene los parámetros termodinámicos a la entrada del compresor
   Tin=Estado_Saturacion.T+Delta_Tsh;
   Estado_in=Obt_Param_Zona_No_Bifasica(Fluido,Pin,Tin);

   % Obtiene la densidad (kg/m3)a la entrada del compresor
   din=1/Estado_in.v;

   % Obtiene las presiones (KPa) a la entrada y salida del compresor
   Pin=100*P1;
   Pout=100*P2;

   % Calcula n según el modelo empírico de Neera Jain
   k1=0.65127; k2=0.00027681; k3=-0.031338;
   k4=3.0221*10^-5; k5=-1.1905*10^-7; k6=-0.0081256;
   n=k1+k2.*RPM+k3*Pout/Pin+k4.*RPM.*Pout/Pin+k5.*RPM.^2+k6*(Pout/Pin)^2;

   % Calcula el caudal másico (kg/s) que atraviesa el compresor kg/s
   m=n*din*Deltak.*RPM./60;

end



% ------------------------------------------------------------------------
%     Función para obtener el caudal en kg/s que atraviesa la válvula
%        de expansión utilizando el modelo propuesto por Neera Jain
%
%       VARIABLES DE ENTRADA
%
%       Fluido:         Uno de los soportados por el CoolProp
%       Delta_Tsc:      Subenfriamiento (°C)
%       Posicion:       Porcentaje apertura (0-100%)
%       P1:             Presión más baja (bares) del ciclo (evaporador)
%       P2:             Presión más alta (bares) del ciclo (condensador)
%
%       VARIABLES DE SALIDA
%
%       m:              Caudal en kg/s
% ------------------------------------------------------------------------

function m=Obt_Q_Valvula_Jain (Fluido,Delta_Tsc,Posicion,P1,P2)

   % Obtiene las presiones (bares) a la entrada y salida de la válvula
   Pin=P2;
   Pout=P1;

   % Obtiene los parámetros termodinámicos de saturación a P=Pin (bares)
   Estado_Saturacion=Obt_Param_Zona_Saturacion (Fluido,'P',Pin);

   % Obtiene los parámetros termodinámicos a la entrada de la válvula
   Tin=Estado_Saturacion.T-Delta_Tsc;
```

```
    Estado_in=Obt_Param_Zona_No_Bifasica(Fluido,Pin,Tin);

    % Obtiene la densidad (kg/m3)a la entrada de la válvula
    din=1/Estado_in.v;

    % Obtiene las presiones (KPa) a la entrada y salida de la válvula
    Pin=100*P2;
    Pout=100*P1;

    % Calcula Kv (m2) según el modelo empírico de Neera Jain
    k1=-9.5984*10^-6; k2=2.0481*10^-6; k3=5.4106*10^-9;
    k4=-7.4909*10^-10; k5=-3.7775*10^-8;
    Kv=k1+k2.*Posicion+k3*(Pin-Pout)+k4.*Posicion.*(Pin-Pout)+k5.*Posicion.^2;

    % Calcula el caudal másico (kg/s) a la salida de la válvula en kg/s
    m=Kv.*sqrt(din*(Pin-Pout));

end
```

C.4. FUNCIONES TERMODINÁMICAS

```
% ------------------------------------------------------------------------------
% Función para fijar la referencia de los parámetros termodinámicos
%
%     VARIABLES DE ENTRADA
%
%     Fluido:               Alguno de los soportados por CoolProp
%     Tipo:                 ASHRAE: h=0, s=0 @-40°C (Líquido saturado)
%                           IIR: h=200 kJ/kg, s=1 kJ/kgK @0°C (Líquido saturado)
%                           NBP: h=0, s=0 @P=1 bar (Líquido saturado)
%                           DEF: Valor por defecto
% ------------------------------------------------------------------------------

function FijarReferencia(Fluido,Tipo)

   switch Tipo

      case 'ASHRAE' % h = 0, s = 0 a -40°C (líquido saturado)
         CoolProp.set_reference_stateS(Fluido,'DEF');
         Densidad_Molar=CoolProp.PropsSI('DMOLAR', 'T', 233.15,'Q', 0, Fluido);
         CoolProp.set_reference_stateD(Fluido,233.15,Densidad_Molar,0,0);

      case 'IIR'    % h = 200 kJ/kg, s=1 kJ/kg/K a 0°C (líquido saturado)
          CoolProp.set_reference_stateS(Fluido,'DEF');
          CoolProp.set_reference_stateS(Fluido,'IIR');

      case 'NBP'    % h = 0, s = 0 a P=1 bar (líquido saturado)
          CoolProp.set_reference_stateS(Fluido,'DEF');
          CoolProp.set_reference_stateS(Fluido,'NBP');

      case 'DEF'
          CoolProp.set_reference_stateS(Fluido,'DEF');

      otherwise
          CoolProp.set_reference_stateS(Fluido,'DEF');

end
```

```
% ------------------------------------------------------------------------------------------
%                Función para obtener los parámetros termodinámicos
%                    de un ciclo de refrigeración de 7 estados
%
%       VARIABLES DE ENTRADA
%
%       Fluido:                     Alguno de los soportados por CoolProp
%       Referencia:                 Tipo de referencia: ASHRAE,IIR,NBP o DEF
%       TH:                         Temperatura ambiente (°C)
%       TL:                         Temperatura recinto a refrigerar (°C)
%       Delta_Te0:                  Salto térmico en el evaporador (°C)
%       Delta_Tc0:                  Salto térmico en el condensador (°C)
%       Delta_Tsh0:                 Sobrecalentamiento (°C)
%       Delta_Tsc0:                 Subenfriamiento (°C)
%       nCompresor:                 Rendimiento del compresor (%)
%       Caudal_Fluido               Caudal del fluido (kg/s)
%
%       VARIABLES DE SALIDA
%
%       Qe:                         Potencia frigorífica (kW)
%       We:                         Potencia requerida (kW)
%       COP:                        Índice de rendimiento del ciclo
%       COP_Carnot:                 Índice de rendimiento para un ciclo Carnot
%       nCarnot:                    Índice de rendimiento referenciado a Carnot
%
%       Una estructura por cada uno de los estados del ciclo (Estado1s,Estado1,
%       Estado2,Estado2s,Estado3s,Estado3,Estado4) con los siguientes campos:
%
%       P:                          Presión (bares)
%       T:                          Temperatura (°C)
%       vf:                         Volumen específico (m3/kg) de líquido saturado a T
%       vg:                         Volumen específico (m3/kg) de vapor saturado a T
%       uf:                         Energía interna (kJ/kg) de líquido saturado a T
%       ug:                         Energía interna (kJ/kg) de vapor saturado a T
%       hf:                         Entalpía (kJ/kg) de líquido saturado a T
%       hg:                         Entalpía (kJ/kg) de vapor saturado a T
%       sf:                         Entropía (kJ/kgK) de líquido saturado a T
%       sg:                         Entropía (kJ/kgK) de vapor saturado a T
%       v:                          Volumen específico (m3/kg) @T,Q
%       u:                          Energía interna (kJ/kg) @T,Q
%       h:                          Entalpía (kJ/kg) @T,Q
%       s:                          Entropía (kJ/kgK) @T,Q
%       x:                          Fracción de vapor (0-1)
%
% ------------------------------------------------------------------------------------------

function [Estado1s,Estado1,Estado2,Estado2s,Estado3s,Estado3,Estado4,Qe,We,...
          COP,COP_Carnot,nCarnot,BoolOK]=Obt_CicloRefrigeracion7 (Fluido,Referencia,TH,TL,...
                                                                 Delta_Te,Delta_Tc,...
                                                                 Delta_Tsh,Delta_Tsc,...
                                                                 nCompresor,Caudal_Fluido)

   try

      % Fija el fluido y la referencia a emplear en los cálculos termodinámicos
      Fijar_Referencia(Fluido,Referencia);

      % Obtiene las temperaturas en el evaporador y condensador
      Te=TL-Delta_Te;
      Tc=TH+Delta_Tc;

      % Obtiene las temperaturas (°C) a la entrada del compresor (T1) y
      % a la entrada de la válvula de expansión (T3)
      T1=Te+Delta_Tsh;
      T3=Tc-Delta_Tsc;

      % Estado 1s
      Estado1s=Obt_Param_Zona_Bifasica(Fluido,Te,1);

      % Estado 1
      Estado1=Obt_Param_Zona_No_Bifasica(Fluido,Estado1s.P,T1);

      % Estado 2s
      Estado2s=Obt_Param_Zona_Bifasica(Fluido,Tc,1);

      % Estado 2
      T2=Obt_T_P_s(Fluido,Estado2s.P,Estado1.s);
      Estado2=Obt_Param_Zona_No_Bifasica(Fluido,Estado2s.P,T2);

      if (nCompresor~=100)
         h2_real=Estado1.h+(Estado2.h-Estado1.h)*100/nCompresor;
         t2_real=Obt_T_P_h(Fluido,Estado2s.P,h2_real);
         Estado2=Obt_Param_Zona_No_Bifasica(Fluido,Estado2s.P,t2_real);
```

```
        end

        % Estado 3s
        Estado3s=Obt_Param_Zona_Bifasica(Fluido,Tc,0);

        % Estado 3
        Estado3=Obt_Param_Zona_No_Bifasica(Fluido,Estado3s.P,T3);

        % Estado 4
        x=Obt_x_P_h(Fluido,Estado1s.P,Estado3.h);
        Estado4=Obt_Param_Zona_Bifasica(Fluido,Te,x);

        % Calcula los parámetros de salida
        Qe=Caudal_Fluido*(Estado1.h-Estado4.h);
        We=Caudal_Fluido*(Estado2.h-Estado1.h);
        COP=Qe/We;
        COP_Carnot=(Estado1s.T+273.15)/(Estado2s.T-Estado1s.T);
        nCarnot=COP/COP_Carnot;

        BoolOK=true;

    catch

        Estado1s=false; Estado1=false; Estado2=false; Estado2s=false; Estado3s=false;
        Estado3=false; Estado4=false; Qe=0; We=0; COP=0; COP_Carnot=0; nCarnot=0;
        BoolOK=false;

    end

end
```

```
% ------------------------------------------------------------------------------------
% Función para obtener los parámetros termodinámicos en la zona BIFÁSICA
%
%       VARIABLES DE ENTRADA
%
%       Fluido:                         Alguno de los soportados por CoolProp
%       T:                              Temperatura (°C)
%       x:                              Fracción de vapor (0<=x<=1)
%
%       VARIABLES DE SALIDA
%
%       Una estructura Estado con los siguientes campos:
%
%       P:                             Presión (bares)
%       T:                              Temperatura (°C)
%       vf:                             Volumen específico (m3/kg) de líquido saturado a T
%       vg:                             Volumen específico (m3/kg) de vapor saturado a T
%       uf:                             Energía interna (kJ/kg) de líquido saturado a T
%       ug:                             Energía interna (kJ/kg) de vapor saturado a T
%       hf:                             Entapía (kJ/kg) de líquido saturado a T
%       hg:                             Entalpía (kJ/kg) de vapor saturado a T
%       sf:                             Entropía (kJ/kgK) de líquido saturado a T
%       sg:                             Entropía (kJ/kgK) de vapor saturado a T
%       v:                              Volumen específico (m3/kg) @T,Q
%       u:                              Energía interna (kJ/kg) @T,Q
%       h:                              Entalpía (kJ/kg) @T,Q
%       s:                              Entropía (kJ/kgK) @T,Q
%       x:                              Fracción de vapor (0-1)
%
% ------------------------------------------------------------------------------------------
function Estado=Obt_Param_Zona_Bifasica(Fluido,T,x)

   % Obtiene los parámetros de saturación
   Estado=Obt_Param_Zona_Saturacion (Fluido,'T',T);

   % Guarda la Calidad
   Estado.x=x;

   % Convierte la Temperatura (°K)
   T1=T+273.15;

   % Calcula el volumen específico (m3/kg) a T=T1 (°K),x
   Estado.v=CoolProp.PropsSI('D', 'T', T1, 'Q', x, Fluido)^-1;

   % Energía interna (kJ/kg) a T=T1 (°K),Q
   Estado.u=CoolProp.PropsSI('U', 'T', T1, 'Q', x, Fluido)/1000;

   % Entalpía (kJ/kg) a T=T1(°K),Q
   Estado.h=CoolProp.PropsSI('H', 'T',T1,'Q',x,Fluido)/1000;

   % Entropía (kJ/kgK) a T=T1 (°K),Q
   Estado.s=CoolProp.PropsSI('S', 'T', T1,'Q', x, Fluido)/1000;

end
```

```
% ------------------------------------------------------------------------------------
% Función para obtener los parámetros termodinámicos en la zona NO BIFÁSICA
%
%       VARIABLES DE ENTRADA
%
%       Fluido:                         Alguno de los soportados por CoolProp
%       P:                              Presión (Bares)
%       T:                              Temperatura (°C)
%
%       VARIABLES DE SALIDA
%
%       Una estructura Estado con los siguientes campos:
%
%       P:                              Presión (bares)
%       T:                              Temperatura (°C)
%       vf:                             Volumen específico (m3/kg) de líquido saturado a T
%       vg:                             Volumen específico (m3/kg) de vapor saturado a T
%       uf:                             Energía interna (kJ/kg) de líquido saturado a T
%       ug:                             Energía interna (kJ/kg) de vapor saturado a T
%       hf:                             Entapía (kJ/kg) de líquido saturado a T
%       hg:                             Entalpía (kJ/kg) de vapor saturado a T
%       sf:                             Entropía (kJ/kgK) de líquido saturado a T
%       sg:                             Entropía (kJ/kgK) de vapor saturado a T
%       v:                              Volumen específico (m3/kg) @T,Q
%       u:                              Energía interna (kJ/kg) @T,Q
%       h:                              Entalpía (kJ/kg) @T,Q
%       s:                              Entropía (kJ/kgK) @T,Q
%       x:                              Fracción de vapor (0-1)
%
% ------------------------------------------------------------------------------------

function Estado=Obt_Param_Zona_No_Bifasica(Fluido,P,T)

   % Obtiene los parámetros de saturación
   Estado=Obt_Param_Zona_Saturacion (Fluido,'T',T);

   % Guarda la Presión (Bares), la Temperatura (°C) y la Calidad
   Estado.P=P; Estado.T=T; Estado.Q=-1;

   % Convierte la Presión (Pa) y la Temperatura (°K)
   P1=P*10^5; T1=T+273.15;

   % Calcula el volumen específico (m3/kg) a P=P1 (Pa) y T=T1(°K)
   Estado.v=CoolProp.PropsSI('D', 'P', P1, 'T', T1, Fluido)^-1;

   % Energía interna (kJ/kg) a P=P1 (Pa) y T=T1 (°K)
   Estado.u=CoolProp.PropsSI('U', 'P', P1, 'T', T1, Fluido)/1000;

   % Entalpía (kJ/kg) a P=P1 (Pa) y T=T1(°K)
   Estado.h=CoolProp.PropsSI('H', 'P',P1,'T',T1,Fluido)/1000;

   % Entropía (kJ/kgK) a P=P1 (Pa) y T=T1 (°K),Q
   Estado.s=CoolProp.PropsSI('S', 'P', P1,'T', T1, Fluido)/1000;

end
```

```
% ------------------------------------------------------------------------------------
% Función para obtener los parámetros termodinámicos en la zona de SATURACIÓN
%
%       VARIABLES DE ENTRADA
%
%       Fluido:                         Alguno de los soportados por CoolProp
%       Param_S:                        'T' si Param_V es T(°C) o P si es presión (bares)
%       Param_V:                        Valor de la temperatura o la presión
%
%       VARIABLES DE SALIDA
%
%       Una estructura Estado con los siguientes campos:
%
%       P:                              Presión (bares)
%       T:                              Temperatura (°C)
%       vf:                             Volumen específico (m3/kg) de líquido saturado a T
%       vg:                             Volumen específico (m3/kg) de vapor saturado a T
%       uf:                             Energía interna (kJ/kg) de líquido saturado a T
%       ug:                             Energía interna (kJ/kg) de vapor saturado a T
%       hf:                             Entapía (kJ/kg) de líquido saturado a T
%       hg:                             Entalpía (kJ/kg) de vapor saturado a T
%       sf:                             Entropía (kJ/kgK) de líquido saturado a T
%       sg:                             Entropía (kJ/kgK) de vapor saturado a T
%       v:                              Volumen específico (m3/kg) @T,x
%       u:                              Energía interna (kJ/kg) @T,x
%       h:                              Entalpía (kJ/kg) @T,x
%       s:                              Entropía (kJ/kgK) @T,x
%
% ------------------------------------------------------------------------------------

function Estado=Obt_Param_Zona_Saturacion (Fluido,Param_S,Param_V)

   switch upper(Param_S)

      case 'P'

         % Guarda la Presión (bares)
         Estado.P=Param_V;

         % Convierte la Presión (Pa)
         P1=Param_V*10^5;

         % Calcula la temperatura T (°C) a P=P1 (Pascales)
         Estado.T=CoolProp.PropsSI('T', 'P',P1,'Q',0,Fluido)-273.15;

      case 'T'

         % Guarda la temperatura (°C)
         Estado.T=Param_V;

         % Convierte la temperatura (°K)
         T1=Param_V+273.15;

         % Calcula la presión P (bares)
         P1=CoolProp.PropsSI('P', 'T',T1,'Q',0,Fluido);
         Estado.P=P1*10^-5;

       otherwise

          Estado.P=-1;Estado.T=-1;
          Estado.vf=-1;Estado.vg=-1;
          Estado.uf=-1;Estado.ug=-1
          Estado.hf=-1;Estado.hg=-1;
          Estado.sf=-1;Estado.sg=-1;

   end

   % Calcula el volumen específico (m3/kg) de líquido saturado a P=P1 (Pa)
   Estado.vf=CoolProp.PropsSI('D', 'P', P1, 'Q', 0, Fluido)^-1;

   % Calcula el volumen específico (m3/kg) de vapor saturado a P=P1 (Pa)
   Estado.vg=CoolProp.PropsSI('D', 'P', P1, 'Q', 1, Fluido)^-1;

   % Energía interna (kJ/kg) de líquido saturado a P=P1 (Pa)
   Estado.uf=CoolProp.PropsSI('U', 'P', P1, 'Q', 0, Fluido)/1000;

   % Energía interna (kJ/kg) de vapor saturado a P=P1 (Pa)
   Estado.ug=CoolProp.PropsSI('U', 'P', P1, 'Q', 1, Fluido)/1000;

   % Entalpía de líquido saturado (kJ/kg) a P=P1 (Pa)
   Estado.hf=CoolProp.PropsSI('H', 'P',P1,'Q',0,Fluido)/1000;
```

```
   % Entalpía de vapor saturado (kJ/kg) a P=P1 (Pa)
   Estado.hg=CoolProp.PropsSI('H', 'P',P1,'Q',1,Fluido)/1000;

   % Entropia de líquido saturado (kJ/kgK) a P=P1 (Pa)
   Estado.sf=CoolProp.PropsSI('S', 'P',P1,'Q',0,Fluido)/1000;

   % Entropía de vapor saturado (kJ/kgK) a P=P1 (Pa)
   Estado.sg=CoolProp.PropsSI('S', 'P',P1,'Q',1,Fluido)/1000;

end


% -------------------------------------------------------------------------------------
%                   Función para obtener la temperatura a la que se encuentra
%                   el fluido en función de la presión y la entalpía
%
%       VARIABLES DE ENTRADA
%
%       Fluido:                         Alguno de los soportados por CoolProp
%       P:                              Presión (bares)
%       h:                              Entalpía (kJ/kg)
%
%       VARIABLES DE SALIDA
%
%       T:                              Temperatura (°C)
%
% -------------------------------------------------------------------------------------

function T=Obt_T_P_h(Fluido,P,h)

   T=CoolProp.PropsSI('T', 'P', P*10^5,'H', h*10^3, Fluido)-273.15;

end


% -------------------------------------------------------------------------------------
%                   Función para obtener la temperatura a la que se encuentra
%                   el fluido en función de la presión y la entropía
%
%       VARIABLES DE ENTRADA
%
%       Fluido:                         Alguno de los soportados por CoolProp
%       P:                              Presión (bares)
%       s:                              Entropía (kJ/kgK)
%
%       VARIABLES DE SALIDA
%
%       T:                              Temperatura (°C)
%
% -------------------------------------------------------------------------------------

function T=Obt_T_P_s(Fluido,P,s)

   T=CoolProp.PropsSI('T', 'P', P*10^5,'S', s*10^3, Fluido)-273.15;

end


% -------------------------------------------------------------------------------------
% Función para obtener la fracción de vapor (x) en la zona bifásica
%
%       VARIABLES DE ENTRADA
%
%       Fluido:                         Alguno de los soportados por CoolProp
%       P:                              Presión (bares)
%       h:                              Entalpía (kJ/kg)
%
%       VARIABLES DE SALIDA
%
%       x:                              Fracción de vapor (0-1)
%
% -------------------------------------------------------------------------------------


function x=Obt_x_P_h(Fluido,P,h)

   Estado=Obt_Param_Zona_Saturacion (Fluido,'P',P);
   x=(h-Estado.hf)/(Estado.hg-Estado.hf);

end
```

C.5. FUNCIONES DE APOYO

```
% ------------------------------------------------------------------------------------
%               Función para ajustar los datos al grid para RPM/Av
%
%       VARIABLES DE ENTRADA
%
%       Datos:                      Matriz con los datos brutos de los puntos de
%                                   operación:
%                                       Col1: Índice para Vector_RPM_O (RPM)
%                                       Col2: Índice para Vector Av_O (Av%)
%                                       Col3: Variable a ajustar al grid (m, P1, P2, etc)
%
%       Vector_RPM_O:               Vector conteniendo los valores RPM correspondientes
%                                   a los datos brutos.
%       Vector_Av_O:                Vector conteniendo los valores Av(%) correspondientes
%                                   a los datos brutos.
%
%       Vector_RPM_D:               Vector conteniendo los valores RPM correspondientes
%                                   al GRID
%       Vector_Av_D:                Vector conteniendo los valores Av (%)correspondientes
%                                   al GRID
%
%       PorcLimite:                 Límite máximo admitido entre el valor máximo/mínimo
%                                   y el medio a partir del cual se incluye la coordenada
%                                   en [RPM_Sup Av_Sup]
%
%       VARIABLES DE SALIDA
%
%       Grid_Datos:                 Estructura conteniendo los siguientes campos:
%           Valores:                Matriz con los datos ajustados al Grid. Consta de 4
%                                   columnas:
%                                       Col1: Valor medio
%                                       Col2: Valor mínimo
%                                       Col3: Valor máximo
%                                       Col4: Número de datos brutos ajustados al punto
%           Min:                    Valor mínimo encontrado en los datos brutos
%           Max:                    Valor máximo encontrado en los datos brutos
%           PtosSuperan_Porc        Índices con los puntos que han superado PorcLimite
%
% ------------------------------------------------------------------------------------

function Grid_Datos= Grid_Ajustar_Datos(Datos,Vector_RPM_O,Vector_Av_O,...
                                        Vector_RPM_D,Vector_Av_D,PorcLimite)

   % Obtiene el paso para el Grid de RPM y para el de Av
   PasoGrid_RPM=Vector_RPM_D(2)-Vector_RPM_D(1);
   PasoGrid_Av=Vector_Av_D(2)-Vector_Av_D(1);

   % Obtiene los valores mínimos del grid para RPM y Av
   RPM_D_Min=Vector_RPM_D(1);
   Av_D_Min=Vector_Av_D(1);


   % Inicializa la matriz que contendrá los datos discretizados:
   % (1)-Dato (2)-Min (3)-Max (4)-N°Datos detectados
   Grid_Datos.Valores=zeros(length(Vector_RPM_D),length(Vector_Av_D),4);

   % Posiciona cada uno de los puntos de operación obtenidos sobre el punto
   % mas cercano del grid guardando los valores máximos, mínimos y medios
   % para cada uno de ellos.
   for i=1:size(Datos,1)

      % Obtiene el punto de operación a analizar
      RPM0=Vector_RPM_O(Datos(i,1));
      Av0=Vector_Av_O(Datos(i,2));
      Valor=Datos(i,3);

      % Obtiene las coordenadas que le corresponden en el grid
      [i_RPM_D,i_Av_D]=Grid_Obt_Indices(RPM0,Av0,RPM_D_Min,Av_D_Min,PasoGrid_RPM,PasoGrid_Av);

      % Si es la primera vez que se analiza ese punto de operación
      if Grid_Datos.Valores(i_RPM_D,i_Av_D,4)==0

         Grid_Datos.Valores(i_RPM_D,i_Av_D,1)=Valor;    % Guarda el valor medio
         Grid_Datos.Valores(i_RPM_D,i_Av_D,2)=Valor;    % Guarda el valor mínimo
         Grid_Datos.Valores(i_RPM_D,i_Av_D,3)=Valor;    % Guarda el valor máximo
         Grid_Datos.Valores(i_RPM_D,i_Av_D,4)=1;        % Guarda el número de datos

      else
```

```
            MediaActual=Grid_Datos.Valores(i_RPM_D,i_Av_D,1);
            MinimoActual=Grid_Datos.Valores(i_RPM_D,i_Av_D,2);
            MaximoActual=Grid_Datos.Valores(i_RPM_D,i_Av_D,3);
            NDatosActual=Grid_Datos.Valores(i_RPM_D,i_Av_D,4);

            % Guarda el nuevo valor medio
            Grid_Datos.Valores(i_RPM_D,i_Av_D,1)=(MediaActual*NDatosActual+Valor)/(NDatosActual+1);

            % Si hay que actualizar el mínimo
            if Valor<MinimoActual
               Grid_Datos.Valores(i_RPM_D,i_Av_D,2)=Valor;
            end

            % Si hay que actualizar el máximo
            if Valor>MaximoActual
               Grid_Datos.Valores(i_RPM_D,i_Av_D,3)=Valor;
            end

             % Guarda el nuevo número de datos
            Grid_Datos.Valores(i_RPM_D,i_Av_D,4)=NDatosActual+1;

         end

      end

      % Calcula la variación máxima permitida sobre la media a partir de la
      % cual se muestra el punto por pantalla
      Grid_Datos.Min=min(Datos(:,3));
      Grid_Datos.Max=max(Datos(:,3));
      Variacion_Max=(Grid_Datos.Max-Grid_Datos.Min)*0.01*PorcLimite;

      % Calcula la diferencia en valor absoluto entre el valor medio y el
      % valor minimo. Si el resultado es mayor que la variación máxima
      % permitida activa el flag de error Error_Min.
      Error_Min=abs((Grid_Datos.Valores(:,:,1)-Grid_Datos.Valores(:,:,2)))-Variacion_Max>0;

      % Calcula la diferencia en valor absoluto entre el valor medio y el
      % valor maximo. Si el resultado es mayor que la variación máxima
      % permitida activa el flag de error Error_Max.
      Error_Max=abs((Grid_Datos.Valores(:,:,1)-Grid_Datos.Valores(:,:,3)))-Variacion_Max>0;

      % Obtiene los índices de los elementos del grid que han incumplido el límite establecido
      [i_RPM_Sup,i_Av_Sup]=find(Error_Min+Error_Max>0);
      Grid_Datos.Ptos_Superan_Porc=sortrows([i_RPM_Sup,i_Av_Sup]);

end
```

```
% ------------------------------------------------------------------------------------
%                Función para la discretización de los puntos de trabajo
%                factibles para un conjunto de ciclos de refrigeración
%
%       VARIABLES DE ENTRADA
%
%       Fichero_Matlab_Out          Fichero donde se guardarán los datos obtenidos:
%                                   (Ciclos,Datos_Grid,Vector_RPM,Vector_Av)
%
%       Parametros_General:         Estructura parámetros de configuración generales
%                                   de los que se usan los siguientes:
%
%           Barrido_RPM:            Barrido a realizar en RPM
%               Min:                Valor mínimo
%               Paso_Porc           Paso en % sobre el rango efectivo de RPM
%               Paso_Abs            Paso en valor absoluto para RM
%               Max:                Valor máximo
%           Barrido_Av:             Barrido a realizar en Av
%               Min:                Valor mínimo
%               Paso_Porc:          Paso en % sobre el rango efectivo de Av
%               Paso_Abs            Paso en valor absoluto para Av
%               Max:                Valor máximo
%
%        Parametros_Grid:           Estructura con los parámetros de configuración
%                                   para el ajuste al GRID. Se utilizan los
%                                   siguientes campos:
%
%               Paso.RPM_Porc:      Paso en % sobre el rango de barrido para RPM
%               Paso.RPM_Abs:       Paso en valor absoluto para RPM. Utilizar uno u otro
%                                   y colocar el otro a cero.
%
%               Paso.Av_Porc:       Paso en % sobre el rango de barrido para Av
%               Paso.Av_Abs:        Paso en valor absoluto para Av. Utilizar uno u otro
%                                   y colocar el otro a cero.
%               ID_Ciclo:           Si es distinto de cero indica que debe ajustarse un
%                                   único ciclo al grid que viene dado por este ID
%               Variables:          Estructura con dos campos por cada variable a tratar.
%                                   El primero indica si debe ajustarse al grid y el
%                                   segundo el porcentaje máximo de desviación antes
%                                   de incluirlo en los puntos a comprobar
%                   m,m_PorcentajeMax
%                   P1,P1_PorcentajeMax, P2,P2_PorcentajeMax
%                   Qe,Qe_PorcentajeMax, W,W_PorcentajeMax
%                   nCarnot.nCarnot_PorcentajeMax,COP,COP_PorcentajeMax
%                   COPc,COPc_PorcentajeMax
%                   T1,T1_PorcentajeMax,T2,T2_PorcentajeMax
%                   T3,T3_PorcentajeMax,T4,T4_PorcentajeMax
%                   DeltaTe,DeltaTe_PorcentajeMax,DeltaTc,DeltaTc_PorcentajeMax
%                   DeltaTsh,DeltaTsh_PorcentajeMax,DeltaTsc,DeltaTsc_PorcentajeMax
%                   TH,TH_PorcentajeMax
%
%       Ciclos:                     Estructura con los puntos de operación válidos.
%                                   Contiene los siguientes campos:
%
%               RPM:                Vector con los valores de RPM de los datos brutos
%               Av:                 Vector con los valores de apertura (%) datos brutos
%               PtosOP_Q:           Vector de tres componentes con los puntos operación
%                                   aptos encontrados. Contiene tres columnas:
%                                       Col1: Índice para Temp_Test
%                                       Col2: Índice para RPM
%                                       Col3: Índice para Av
%                                       Col4: Caudal (kg/s)
%
%               P1:                 Vector de n filas con la presión (bares) evaporador
%                                   para cada una de las filas de Temp_Test
%               P2:                 Vector de n filas con la presión (bares) condensador
%                                   para cada una de las filas de Temp_Test
%               T1:                 Vector de n filas con la temperatura (ºC) entrada
%                                   del compresor para cada una de las filas de Temp_Test
%               T2:                 Vector de n filas con la temperatura (ºC) a la salida
%                                   del compresor para cada una de las filas de Temp_Test
%               T3:                 Vector de n filas con la temperatura (ºC) entrada
%                                   de la válvula de expansión para cada una de las filas
%                                   de Temp_Test
%               T4:                 Vector de n filas con la temperatura (ºC) salida de
%                                   la válvula de expansión para cada una de las filas
%                                   de Temp_Test
%               H14:                Vector de n filas con la diferencia de entalpía entre
%                                   los estados 1 y 4 en kJ/kg para cada una de las filas
%                                   de Temp_Test
%               H21:                Vector de n filas con la diferencia de entalpía entre
%                                   los estados 2 y 1 en kJ/kg para cada una de las filas
%                                   de Temp_Test
```

```
%               COPc:               Vector de nfilas con el COP Carnot para cada una de
%                                   las filas de Temp_Test
%
%               Temp_Test:          Matriz con n filas conteniendo las temp. empleadas:
%                                      Col1: Delta_Te (ºC)
%                                      Col2: Delta_Tc (ºC)
%                                      COl3: Delta_Tsh (ºC)
%                                      Col4: Delta_Tsc (ºC)
%                                      COl5: TH (ºC)
%
%       VARIABLES DE SALIDA
%
%          Vector_RPM:              Vector con los valores de RPM del grid
%          Vector_Av:               Vector con los valores de Av del grid
%
%          Datos_Grid:              Estructura con un campo por cada una de las variables
%                                   ajustadas al grid:
%                                   (m, P1, P2, Qe, W, nCarnot, COP,COPc, T1, T2, T3, T4,
%                                   DeltaTe, DeltaTc, DeltaTsh, DeltaTsc y TH).
%                                   Cada uno de ellos es, a su vez, otra estructura con
%                                   estos campos:
%
%               Valores:            Matriz con los datos ajustados al Grid. Consta de 4
%                                   columnas:
%                                       Col1: Valor medio
%                                       Col2: Valor mínimo
%                                       Col3: Valor máximo
%                                       Col4: Número de datos brutos ajustados al punto
%               Min:                Valor mínimo encontrado en los datos brutos
%               Max:                Valor máximo encontrado en los datos brutos
%               PtosSuperan_Porc    Índices con los puntos que han superado PorcLimite
%
% ------------------------------------------------------------------------------------

function [Vector_RPM,Vector_Av,Datos_Grid]=...
         Grid_Ajustar_PtosOp_Ciclos_RPMAV (Fichero_Matlab_Out,Parametros_General,...
                                           Parametros_Grid,Ciclos)


   % Obtiene los parámetros
   PasoGrid=Parametros_Grid.Paso;
   Variables=Parametros_Grid.Variables;


   % Si el paso de RPM para el grid se ha indicado en % sobre el rango
   if PasoGrid.RPM_Porc>0 && PasoGrid.RPM_Abs==0

      % Obtiene el paso en valor absoluto a utilizar en el barrido de RPM
      PasoGrid.RPM=0.01*PasoGrid.RPM_Porc*(Parametros_General.Barrido_RPM.Max-...
                                           Parametros_General.Barrido_RPM.Min);

   else

      PasoGrid.RPM=PasoGrid.RPM_Abs;

   end


   % Si el paso de Av para el grid se ha indicado en % sobre el rango
   if PasoGrid.Av_Porc>0 && PasoGrid.Av_Abs==0

      % Obtiene el paso en valor absoluto a utilizar en el barrido de RPM
      PasoGrid.Av=0.01*PasoGrid.Av_Porc*(Parametros_General.Barrido_Av.Max-...
                                         Parametros_General.Barrido_Av.Min);

   else

      PasoGrid.Av=PasoGrid.Av_Abs;

   end


   % Inicializa las variables de salida
   Datos_Grid.m=false; Datos_Grid.P1=false; Datos_Grid.P2=false;
   Datos_Grid.Qe=false;Datos_Grid.W=false;Datos_Grid.nCarnot=false;
   Datos_Grid.COP=false;Datos_Grid.COPc=false;Datos_Grid.T1=false;
   Datos_Grid.T2=false;Datos_Grid.T3=false;Datos_Grid.T4=false;
   Datos_Grid.DeltaTe=false;Datos_Grid.DeltaTc=false;Datos_Grid.DeltaTsh=false;
   Datos_Grid.DeltaTsc=false;Datos_Grid.TH=false;
```

```
% Obtiene el rango efectivo obtenido para RPM y Av
RPM_Min=fix(min(Ciclos.RPM(Ciclos.PtosOp_Q(:,2))));
RPM_Max=fix(max(Ciclos.RPM(Ciclos.PtosOp_Q(:,2)))+1);
Av_Min=fix(min(Ciclos.Av(Ciclos.PtosOp_Q(:,3))));
Av_Max=fix(max(Ciclos.Av(Ciclos.PtosOp_Q(:,3)))+1);

% Muestra el rango a analizar
fprintf ('RANGO DE ESTUDIO:\n');
fprintf('RPM (min):%5d\t\tRPM(max): %5d \tAv(min):%3d \tAv(max):%3d\n',...
        RPM_Min,RPM_Max,Av_Min,Av_Max);

% Genera el grid a utilizar en el estudio
[Vector_RPM,Vector_Av]=Grid_Generar(RPM_Min,RPM_Max,Av_Min,Av_Max,PasoGrid.RPM,PasoGrid.Av);

 % Si ID_Ciclo es distinto cero indica el único ciclo a tener en cuenta.
% Este valor puede obtenerse del fichero Excel
if Parametros_Grid.ID_Ciclo~=0
    Pos_Min_Ciclo=find (Ciclos.PtosOp_Q(:,1)==Parametros_Grid.ID_Ciclo,1,'first');
    Pos_Max_Ciclo=find (Ciclos.PtosOp_Q(:,1)==Parametros_Grid.ID_Ciclo,1,'last');
    Ciclos.PtosOp_Q=Ciclos.PtosOp_Q(Pos_Min_Ciclo:Pos_Max_Ciclo,:);
end


% Guarda en Datos_Temp los índices de RPM(1) y Av(2)
Datos_Temp(:,1:2)=Ciclos.PtosOp_Q(:,2:3);

% Si hay que tratar el caudal (m)
if (Variables.m)

   % Almacena el valor del caudal en g/s en Datos_Temp
   Datos_Temp(:,3)= Ciclos.PtosOp_Q(:,4)*1000;

   % Ajusta los datos del caudal (g/s) al grid
   Datos_Grid.m=Grid_Ajustar_Datos(Datos_Temp,Ciclos.RPM,Ciclos.Av,...
                                   Vector_RPM,Vector_Av,Variables.m_PorcentajeMax);

   % Muestra los datos que incumplen el porcentaje establecido
   Grid_Mostrar_Info_Variable('m (g/s)',Datos_Grid.m,Vector_RPM,Vector_Av);

end


% Ajusta la presión P1 (bares) con los parámetros establecidos previamente
if (Variables.P1)

   % Almacena el valor de P1 (bares) en Datos_Temp
   Datos_Temp(:,3)= Ciclos.P1(Ciclos.PtosOp_Q(:,1));

   % Ajusta los datos de P1 (bares) al grid
   Datos_Grid.P1=Grid_Ajustar_Datos(Datos_Temp,Ciclos.RPM,Ciclos.Av,...
                                    Vector_RPM,Vector_Av,Variables.P1_PorcentajeMax);

   % Muestra los datos que incumplen el porcentaje establecido
   Grid_Mostrar_Info_Variable('P1 (bares)',Datos_Grid.P1,Vector_RPM,Vector_Av);

end


% Ajusta la presión P2 (bares) con los parámetros establecidos previamente
if (Variables.P2)

   % Almacena el valor de P2 (bares) en Datos_Temp
   Datos_Temp(:,3)= Ciclos.P2(Ciclos.PtosOp_Q(:,1));

   % Ajusta los datos de P2 (bares) al grid
   Datos_Grid.P2=Grid_Ajustar_Datos(Datos_Temp,Ciclos.RPM,Ciclos.Av,...
                                    Vector_RPM,Vector_Av,Variables.P2_PorcentajeMax);

   % Muestra los datos que incumplen el porcentaje establecido
   Grid_Mostrar_Info_Variable('P2 (bares)',Datos_Grid.P2,Vector_RPM,Vector_Av);

end


% Ajusta la potencia frigorífica Qe(W) con los parámetros establecidos previamente
if (Variables.Qe)

   % Almacena el valor de Qe (W) en Datos_Temp
   Datos_Temp(:,3)=Ciclos.H14(Ciclos.PtosOp_Q(:,1)).*...
                   Ciclos.PtosOp_Q(:,4)*1000;


   % Ajusta los datos de Qe (W) al grid
```

```
    Datos_Grid.Qe=Grid_Ajustar_Datos(Datos_Temp,Ciclos.RPM,Ciclos.Av,...
                                     Vector_RPM,Vector_Av,Variables.Qe_PorcentajeMax);

    % Muestra los datos que incumplen el porcentaje establecido
    Grid_Mostrar_Info_Variable('Qe(W)',Datos_Grid.Qe,Vector_RPM,Vector_Av);

end


% Ajusta el trabajo (W) con los parámetros establecidos previamente
if (Variables.W)

    % Almacena el valor de W (W) en Datos_Temp
    Datos_Temp(:,3)=Ciclos.H21(Ciclos.PtosOp_Q(:,1)).*...
                    Ciclos.PtosOp_Q(:,4)*1000;

    % Ajusta los datos de Qe (W) al grid
    Datos_Grid.W=Grid_Ajustar_Datos(Datos_Temp,Ciclos.RPM,Ciclos.Av,...
                                    Vector_RPM,Vector_Av,Variables.W_PorcentajeMax);

    % Muestra los datos que incumplen el porcentaje establecido
    Grid_Mostrar_Info_Variable('W(W)',Datos_Grid.W,Vector_RPM,Vector_Av);

end


% Ajusta el rendimiento de CARNOT con los parámetros establecidos previamente
if (Variables.nCarnot)

    % Almacena el valor de nCarnot en Datos_Temp
    Datos_Temp(:,3)= (Ciclos.H14(Ciclos.PtosOp_Q(:,1))./...
                      Ciclos.H21(Ciclos.PtosOp_Q(:,1)))./...
                      Ciclos.COPc(Ciclos.PtosOp_Q(:,1));

    % Ajusta los datos de nCarnot al grid
    Datos_Grid.nCarnot=Grid_Ajustar_Datos(Datos_Temp,Ciclos.RPM,Ciclos.Av,Vector_RPM,...
                                          Vector_Av,Variables.nCarnot_PorcentajeMax);

    % Muestra los datos que incumplen el porcentaje establecido
    Grid_Mostrar_Info_Variable('nCarnot',Datos_Grid.nCarnot,Vector_RPM,Vector_Av);

end


% Ajusta el COP con los parámetros establecidos previamente
if (Variables.COP)

    % Almacena el valor del COP en Datos_Temp
    Datos_Temp(:,3)=Ciclos.H14(Ciclos.PtosOp_Q(:,1))./...
                    Ciclos.H21(Ciclos.PtosOp_Q(:,1));

    % Ajusta los datos del COP al grid
    Datos_Grid.COP=Grid_Ajustar_Datos(Datos_Temp,Ciclos.RPM,Ciclos.Av,Vector_RPM,...
                                      Vector_Av,Variables.COP_PorcentajeMax);

    % Muestra los datos que incumplen el porcentaje establecido
    Grid_Mostrar_Info_Variable('COP',Datos_Grid.COP,Vector_RPM,Vector_Av);

end


% Ajusta el COPc con los parámetros establecidos previamente
if (Variables.COPc)

    % Almacena el valor del COPc en Datos_Temp
    Datos_Temp(:,3)=Ciclos.COPc(Ciclos.PtosOp_Q(:,1));

    % Ajusta los datos del COPc al grid
    Datos_Grid.COPc=Grid_Ajustar_Datos(Datos_Temp,Ciclos.RPM,Ciclos.Av,Vector_RPM,...
                                       Vector_Av,Variables.COPc_PorcentajeMax);

    % Muestra los datos que incumplen el porcentaje establecido
    Grid_Mostrar_Info_Variable('COPc',Datos_Grid.COPc,Vector_RPM,Vector_Av);

end


% Ajusta la temperatura T1 (°C) con los parámetros establecidos previamente
if (Variables.T1)
```

```
    % Almacena el valor de T1 (°C) en Datos_Temp
    Datos_Temp(:,3)= Ciclos.T1(Ciclos.PtosOp_Q(:,1));

    % Ajusta los datos de T1 (°C) al grid
    Datos_Grid.T1=Grid_Ajustar_Datos(Datos_Temp,Ciclos.RPM,Ciclos.Av,...
                                     Vector_RPM,Vector_Av,Variables.T1_PorcentajeMax);

    % Muestra los datos que incumplen el porcentaje establecido
    Grid_Mostrar_Info_Variable('T1 (°C)',Datos_Grid.T1,Vector_RPM,Vector_Av);

end

% Ajusta la temperatura T2 (°C) con los parámetros establecidos previamente
if (Variables.T2)

    % Almacena el valor de T2 (°C) en Datos_Temp
    Datos_Temp(:,3)= Ciclos.T2(Ciclos.PtosOp_Q(:,1));

    % Ajusta los datos de T2 (°C) al grid
    Datos_Grid.T2=Grid_Ajustar_Datos(Datos_Temp,Ciclos.RPM,Ciclos.Av,...
                                     Vector_RPM,Vector_Av,Variables.T2_PorcentajeMax);

    % Muestra los datos que incumplen el porcentaje establecido
    Grid_Mostrar_Info_Variable('T2 (°C)',Datos_Grid.T2,Vector_RPM,Vector_Av);

end

% Ajusta la temperatura T3 (°C) con los parámetros establecidos previamente
if (Variables.T3)

    % Almacena el valor de T3 (°C) en Datos_Temp
    Datos_Temp(:,3)= Ciclos.T3(Ciclos.PtosOp_Q(:,1));

       % Ajusta los datos de T3 (°C) al grid
       Datos_Grid.T3=Grid_Ajustar_Datos(Datos_Temp,Ciclos.RPM,Ciclos.Av,...
                                        Vector_RPM,Vector_Av,Variables.T3_PorcentajeMax);

       % Muestra los datos que incumplen el porcentaje establecido
       Grid_Mostrar_Info_Variable('T3 (°C)',Datos_Grid.T3,Vector_RPM,Vector_Av);

end

% Ajusta la temperatura T4 (°C) con los parámetros establecidos previamente
if (Variables.T4)

    % Almacena el valor de T4 (°C) en Datos_Temp
    Datos_Temp(:,3)= Ciclos.T4(Ciclos.PtosOp_Q(:,1));

    % Ajusta los datos de T4 (°C) al grid
    Datos_Grid.T4=Grid_Ajustar_Datos(Datos_Temp,Ciclos.RPM,Ciclos.Av,...
                                     Vector_RPM,Vector_Av,Variables.T4_PorcentajeMax);

    % Muestra los datos que incumplen el porcentaje establecido
    Grid_Mostrar_Info_Variable('T4 (°C)',Datos_Grid.T4,Vector_RPM,Vector_Av);

end


% Ajusta la temperatura Delta_Te (°C) con los parámetros establecidos previamente
if (Variables.DeltaTe)

    % Almacena el valor de DeltaTe (°C) en Datos_Temp
    Datos_Temp(:,3)= Ciclos.Temp_Test(Ciclos.PtosOp_Q(:,1),1);

    % Ajusta los datos de DeltaTe (°C) al grid
    Datos_Grid.DeltaTe=Grid_Ajustar_Datos(Datos_Temp,Ciclos.RPM,Ciclos.Av,...
                                          Vector_RPM,Vector_Av,...
                                          Variables.DeltaTe_PorcentajeMax);

    % Muestra los datos que incumplen el porcentaje establecido
    Grid_Mostrar_Info_Variable('Delta Te (°C)',Datos_Grid.DeltaTe,Vector_RPM,Vector_Av);

end


% Ajusta la temperatura Delta_Tc (°C) con los parámetros establecidos previamente
if (Variables.DeltaTc)

    % Almacena el valor de DeltaTc (°C) en Datos_Temp
    Datos_Temp(:,3)= Ciclos.Temp_Test(Ciclos.PtosOp_Q(:,1),2);

    % Ajusta los datos de DeltaTc (°C) al grid
    Datos_Grid.DeltaTc=Grid_Ajustar_Datos(Datos_Temp,Ciclos.RPM,Ciclos.Av,...
```

```
                                        Vector_RPM,Vector_Av,...
                                        Variables.DeltaTc_PorcentajeMax);

      % Muestra los datos que incumplen el porcentaje establecido
      Grid_Mostrar_Info_Variable('Delta Tc (°C)',Datos_Grid.DeltaTc,Vector_RPM,Vector_Av);

   end


   % Ajusta la temperatura Delta_Tsh (°C) con los parámetros establecidos previamente
   if (Variables.DeltaTsh)

      % Almacena el valor de DeltaTsh (°C) en Datos_Temp
      Datos_Temp(:,3)= Ciclos.Temp_Test(Ciclos.PtosOp_Q(:,1),3);

      % Ajusta los datos de DeltaTsh (°C) al grid
      Datos_Grid.DeltaTsh=Grid_Ajustar_Datos(Datos_Temp,Ciclos.RPM,Ciclos.Av,...
                                             Vector_RPM,Vector_Av,...
                                             Variables.DeltaTsh_PorcentajeMax);

      % Muestra los datos que incumplen el porcentaje establecido
      Grid_Mostrar_Info_Variable('Delta Tsh (°C)',Datos_Grid.DeltaTsh,Vector_RPM,Vector_Av);

   end


   % Ajusta la temperatura Delta_Tsc (°C) con los parámetros establecidos previamente
   if (Variables.DeltaTsc)

      % Almacena el valor de DeltaTsc (°C) en Datos_Temp
      Datos_Temp(:,3)= Ciclos.Temp_Test(Ciclos.PtosOp_Q(:,1),4);

      % Ajusta los datos de DeltaTsc (°C) al grid
      Datos_Grid.DeltaTsc=Grid_Ajustar_Datos(Datos_Temp,Ciclos.RPM,Ciclos.Av,...
                                             Vector_RPM,Vector_Av,...
                                             Variables.DeltaTsc_PorcentajeMax);

      % Muestra los datos que incumplen el porcentaje establecido
      Grid_Mostrar_Info_Variable('Delta Tsc (°C)',Datos_Grid.DeltaTsc,Vector_RPM,Vector_Av);

   end


   % Ajusta la temperatura TH (°C) con los parámetros establecidos previamente
   if (Variables.TH)

      % Almacena el valor de TH (°C) en Datos_Temp
       Datos_Temp(:,3)= Ciclos.Temp_Test(Ciclos.PtosOp_Q(:,1),5);

      % Ajusta los datos de TH (°C) al grid
      Datos_Grid.TH=Grid_Ajustar_Datos(Datos_Temp,Ciclos.RPM,Ciclos.Av,...
                                       Vector_RPM,Vector_Av,Variables.TH_PorcentajeMax);

      % Muestra los datos que incumplen el porcentaje establecido
      Grid_Mostrar_Info_Variable('TH (°C)',Datos_Grid.TH,Vector_RPM,Vector_Av);

   end


   % Guarda todas las variables para un posterior análisis en el fichero indicado
   if ~isempty(Fichero_Matlab_Out)
      delete (Fichero_Matlab_Out);
      fprintf ('\nGuardando variables en fichero Matlab.....\n');
      save (Fichero_Matlab_Out,'Ciclos','Datos_Grid','Vector_RPM','Vector_Av','-v7.3');
   end

   fprintf ('\nProceso finalizado correctamente.\n');

end
```

```
% ------------------------------------------------------------------------------------
%                            Función para obtener los vectores necesarios
%                            para la discretización de los datos
%
%       VARIABLES DE ENTRADA
%
%       RPM_Min:                    Valor mínimo para el grid en RPM
%       RPM_Max_Ini:                Valor máximo inicial para el grid en RPM
%       Av_Min:                     Valor mínimo para el grid en Av (%)
%       Av_Max_Ini:                 Valor máximo inicial para el grid en Av (%)
%       PasoGrid_RPM:               Paso del grid para RPM
%       PasoGrid_Av:                Paso del grid para Av
%
%       VARIABLES DE SALIDA
%
%       Vector_RPM:                 Vector conteniendo los puntos del grid para RPM
%       Vector_Av:                  Vector conteniendo los puntos del grid para Av
%
% ------------------------------------------------------------------------------------

function [Vector_RPM, Vector_Av]= ...
         Grid_Generar(RPM_Min,RPM_Max_Ini,Av_Min,Av_Max_Ini,PasoGrid_RPM,PasoGrid_Av)

   % Obtiene el vector para RPM
   Vector_RPM=[RPM_Min:PasoGrid_RPM:RPM_Max_Ini];

   if Vector_RPM(length(Vector_RPM))<RPM_Max_Ini
      Vector_RPM(length(Vector_RPM)+1)=Vector_RPM(length(Vector_RPM))+PasoGrid_RPM;
      RPM_Max=Vector_RPM(length(Vector_RPM));
   else
      RPM_Max=RPM_Max_Ini;
   end

   % Obtiene el vector para Av
   Vector_Av=[Av_Min:PasoGrid_Av:Av_Max_Ini];
   if Vector_Av(length(Vector_Av))<Av_Max_Ini
      Vector_Av(length(Vector_Av)+1)=Vector_Av(length(Vector_Av))+PasoGrid_Av;
      Av_Max=Vector_Av(length(Vector_Av));
   else
      Av_Max=Av_Max_Ini;
   end

end
```

```
% ------------------------------------------------------------------------------------
%                Función para mostrar información sobre los puntos ajustados al grid
%
%       VARIABLES DE ENTRADA
%
%       Variable_Desc:              String descriptivo de la variable.
%       Grid_Datos:                 Estructura con los siguientes campos:
%
%               Valores:            Matriz con los datos ajustados al Grid. Consta de 4
%                                   columnas:
%                                       Col1: Valor medio
%                                       Col2: Valor mínimo
%                                       Col3: Valor máximo
%                                       Col4: Número de datos brutos ajustados al punto
%               Min:                Valor mínimo encontrado en los datos brutos
%               Max:                Valor máximo encontrado en los datos brutos
%               PtosSuperan_Porc    Índices con los puntos que han superado PorcLimite
%
%       Vector_RPM:                 Vector conteniendo los valores a utilizar en el grid
%                                   para RPM.
%       Vector_Av:                  Vector conteniendo los valores a utilizar en el grid
%                                   para Av.
% ------------------------------------------------------------------------------------


function Grid_Mostrar_Info_Variable(Variable_Desc,Grid_Datos,Vector_RPM,Vector_Av)

   % Muestra por pantalla el parámetro y su valor mínimo y máximo en el grid
   fprintf('\nPARÁMETRO: %s \t VALOR MÍNIMO:%7.4f \t VALOR MÁXIMO:%7.4f \n', ...
           Variable_Desc,Grid_Datos.Min,Grid_Datos.Max);

   % Si hay datos que han incumplido el límite establecido
   if length(Grid_Datos.Ptos_Superan_Porc)>0

      % Muestra el mensaje informativo
      fprintf ('INCUMPLEN el límite establecido:\n\n');

      i_RPM_Sup=Grid_Datos.Ptos_Superan_Porc(:,1);
      i_Av_Sup=Grid_Datos.Ptos_Superan_Porc(:,2);

      % Recorre los puntos del grid que han incumplido el límite
      for i=1:length(i_RPM_Sup)

         Dato_Med=Grid_Datos.Valores(i_RPM_Sup(i),i_Av_Sup(i),1); % Valor medio en el punto
         Dato_Min=Grid_Datos.Valores(i_RPM_Sup(i),i_Av_Sup(i),2); % Valor mínimo en el punto
         Dato_Max=Grid_Datos.Valores(i_RPM_Sup(i),i_Av_Sup(i),3); % Valor máximo en el punto

         % Obtiene los porcentajes de error para el dato mínimo y máximo
         Porc_Min=100*abs((Dato_Med-Dato_Min)/(Grid_Datos.Max-Grid_Datos.Min));
         Porc_Max=100*abs((Dato_Med-Dato_Max)/(Grid_Datos.Max-Grid_Datos.Min));

         % Muestra la información por pantalla
         fprintf('Indice RPM: %5d \tIndice Av:%5d \t\t',i_RPM_Sup(i),i_Av_Sup(i));
         fprintf('RPM:% 8.2f \tAv:% 6.2f\t',Vector_RPM(i_RPM_Sup(i)), Vector_Av(i_Av_Sup(i)));
         fprintf('Min:% 7.3f(% 6.2f%%) \tMed:% 7.3f \t',Dato_Min,Porc_Min,Dato_Med);
         fprintf('Max:% 7.3f(% 6.2f%%) \n',Dato_Max,Porc_Max);

      end

   else
       fprintf('TODOS los puntos CUMPLEN el límite establecido.\n');
   end

end
```

```
% ------------------------------------------------------------------------------------
%  Función para obtener los índices del grid al que pertenece el punto operación RPM0-Av0
%
%       VARIABLES DE ENTRADA
%
%       RPM0:                       Revoluciones por minuto del compresor
%       Av:                         Porcentaje apertura válvula
%       RPM_Min:                    Valor mínimo del grid para RPM
%       Av_Min:                     Valor mínimo del grid para Av
%       PasoGrid_RPM:               Paso del grid para RPM
%       PasoGrid_Av:                Paso del grid para Av
%
%
%       VARIABLES DE SALIDA
%
%       i_RPMP:                     Índice en el grid RPM0
%       i_Av:                       Índice en el grid para Av0
%
% ------------------------------------------------------------------------------------

function [i_RPM,i_Av]=Grid_Obt_Indices(RPM0,Av0,RPM_Min,Av_Min,PasoGrid_RPM,PasoGrid_Av)

   % Obtiene el índice de RPM
   i_RPM=fix((RPM0-RPM_Min)/PasoGrid_RPM)+1;
   Valor_Base=RPM_Min+PasoGrid_RPM*(i_RPM-1);
   if RPM0>Valor_Base
       Valor_Medio=Valor_Base+PasoGrid_RPM/2;
       if RPM0>=Valor_Medio
           i_RPM=i_RPM+1;
       end
   end


   % Obtiene el índice de Av
   i_Av=fix((Av0-Av_Min)/PasoGrid_Av)+1;
   Valor_Base=Av_Min+PasoGrid_Av*(i_Av-1);
   if Av0>Valor_Base
       Valor_Medio=Valor_Base+PasoGrid_Av/2;
       if Av0>=Valor_Medio
           i_Av=i_Av+1;
       end
   end

   % Acota el límite inferior del índice de RPM
   if i_RPM<1
       i_RPM=1;
   end

   % Acota el límite inferior del índice de Av
   if i_Av<1
       i_Av=1;
   end

end
```

```
% ------------------------------------------------------------------------------------
%                Función para la ordenación de los indices correspondientes a unos datos
%                de la manera indicada en Modo
%
%       VARIABLES DE ENTRADA
%
%       Indice_RPM:                Indices de RPM del compresor de los datos a ordenar
%       Indice_Av:                 Índices de Av (%) de la válvula datos a ordenar
%       Datos:                     Matriz (RPM-Av) con los datos a ordenar
%       Modo:                      'ascend' o 'descend'
%
%       VARIABLES DE SALIDA
%
%       Indice_RPM_Sort:           Indice_RPM ordenados en función de los datos y modo
%       Indice_Av_Sort:            Indice_Av ordenados en función de los datos y modo
%
% ------------------------------------------------------------------------------------

function [Indice_RPM_Sort, Indice_Av_Sort]=Grid_Ordenar_Indices(Indice_RPM,Indice_Av,Datos,Modo)

   % Inicializa los índices ordenados así como el vector con los datos a ordenar
   Vector_Datos=zeros(length(Indice_RPM),1);
   Indice_RPM_Sort=zeros(length(Indice_RPM),1);
   Indice_Av_Sort=zeros(length(Indice_RPM),1);

   % Carga el vector con los datos a ordenar
   for i=1:length(Indice_RPM)
      Vector_Datos(i)=Datos(Indice_RPM(i),Indice_Av(i));
   end

   % Obtiene los índices del vector ordenados
   [Vector_Datos_Sort Indice_Vector_Sort]=sort(Vector_Datos,Modo);

   % Actualiza los índices de salida ordenados
   for i=1:length(Indice_RPM)
      Indice_RPM_Sort(i)=Indice_RPM(Indice_Vector_Sort(i));
      Indice_Av_Sort(i)=Indice_Av(Indice_Vector_Sort(i));
   end

end
```

```
% ------------------------------------------------------------------------------------
%               Función para obtener los Puntos de operación permitidos (RPM-Av) de
%               un ciclo utilizando los modelos de válvula y compresor seleccionados.
%
%
%       VARIABLES DE ENTRADA
%
%       RPM:                        Vector con los valores de RPM para el barrido
%       Av:                         Vector con los valores de apertura (%) para barrido
%
%       Parametros_General:         Estructura parámetros de configuración generales
%                                   de los que se usan los siguientes:
%
%           Fluido:                 Alguno de los soportados por CoolProp
%           Referencia:             Tipo de referencia: ASHRAE,IIR,NBP o DEF
%           PuntoTrabajo:           Estructura con los siguientes campos:
%
%               TL0:                Temperatura (°C) en el recinto a refrigerar
%               TH0:                Temperatura ambiente (°C)
%               Delta_Te0:          Salto térmico (°C) en el evaporador
%               Delta_Tc0:          Salto térmico (°C) en el condensador
%               Delta_Tsh0:         Sobrecalentamiento (°C)
%               Delta_Tsc0:         Subenfriamiento (°C)
%
%           Restricciones_Caudal:
%               Min:                Caudal mínimo (g/s) a seleccionar
%               Tolerancia:         Diferencia máxima (g/s) permitida entre los caudales
%                                   del compresor y la válvula
%
%           Param_Compresor:        Parámetros del compresor
%           Param_Valvula:          Parámetros para la válvula
%
%       Temp_Test:                  Matriz con n filas conteniendo las temp. empleadas:
%                                      Col1: Delta_Te (°C)
%                                      Col2: Delta_Tc (°C)
%                                      COl3: Delta_Tsh (°C)
%                                      Col4: Delta_Tsc (°C)
%                                      COl5: TH (°C)
%
%
%       VARIABLES DE SALIDA
%
%       PtosOP:                     Matriz de tres componentes con los ptos de operación
%                                   aptos encontrados:
%                                       Col1: Índice para RPM
%                                       Col2: Índice para Av
%                                       Col3: Caudal (kg/s)
%
%       P1:                         Presión (bares) en el evaporador.
%       P2:                         Presión (bares) en el condensador.
%       H14:                        Diferencia entalpía entre los estados 1 y 4 en kJ/kg
%       H21:                        Diferencia entalpía entre los estados 2 y 1 en kJ/kg
%       COPc:                       COP de Carnot
%       T1:                         Temperatura (°C) a la entrada del compresor
%       T2:                         Temperatura (°C) a la salida del compresor
%       T3:                         Temperatura (°C) a la entrada válvula de expansión
%       T4:                         Temperatura (°C) a la salida válvula de expansión
%
% ------------------------------------------------------------------------------------


function [PtosOp,P1,P2,H14,H21,...
          COPc,T1,T2,T3,T4]= Obt_PtosOP_Ciclo (RPM,Av,Parametros,Temp_Test)

   try

      % Obtiene los valores de las temperaturas
      Delta_Te0=Temp_Test(1); Delta_Tc0=Temp_Test(2);
      Delta_Tsh0=Temp_Test(3); Delta_Tsc0=Temp_Test(4);
      TH0=Temp_Test(5);

      [Estado1s,Estado1,Estado2,Estado2s,Estado3s,Estado3,Estado4,...
       Qe,We,COP,COP_Carnot,nCarnot,BoolOK]= ...
       Obt_CicloRefrigeracion7 (Parametros.Fluido,Parametros.Referencia,TH0,...
                                Parametros.PuntoTrabajo.TL0,Delta_Te0,Delta_Tc0, ...
                                Delta_Tsh0,Delta_Tsc0,100,1);

      if BoolOK

         P1=Estado1s.P;           % Presión (bares) en el evaporador
         P2=Estado2s.P;           % Presión (bares) en el condensador
         T1=Estado1.T;            % Temperatura T1 (°C) a la entrada del compresor
         T2=Estado2.T;            % Temperatura T2 (°C) a la salida del compresor
```

```
        T3=Estado3.T;            % Temperatura T3 (°C) a la entrada de la válvula
        T4=Estado4.T;            % Temperatura T4 (°C) a la salida de la válvula
        H14=Estado1.h-Estado4.h; % Diferencia de entalpía entre 1 y 4 en kJ/kg
        H21=Estado2.h-Estado1.h; % Diferencia de entalpía entre 2 y 1 en kJ/kg
        COPc=COP_Carnot;         % Valor del COP de Carnot

        % Obtiene el rango de valores posibles del caudal (g/s) en función de
        % la velocidad de giro del compresor
        Parametros.Param_Compresor.Delta_Tsh=Delta_Tsh0;
        mc=Obt_Q_Compresor (Parametros.Fluido,Parametros.Param_Compresor,RPM,P1,P2)*1000;

        % Obtiene el rango de valores posibles del caudal (g/s) en función de
        % la apertura de la válvula
        Parametros.Param_Valvula.Delta_Tsc=Delta_Tsc0;
        mv=Obt_Q_Valvula (Parametros.Fluido,Parametros.Param_Valvula,Av,P1,P2)*1000;

        % Recorre todos los posibles valores de flujo en función de la velocidad
        % de giro del compresor
        k=1; PtosOp(k,1)=0;PtosOp(k,2)=0;PtosOp(k,3)=0;
        for i=1:length(mc) % @RPM

           % Obtiene la diferencia entre todos los valores posibles de caudal
           % de la válvula y cada uno de los del compresor.
           mc_mv=abs(mv-mc(i));

           % Busca los valores de caudal de la válvula y el compresor cuya
           % diferencia sea menor o igual a la Tolerancia.
           Indices=find(mc_mv<=Parametros.Restricciones_Caudal.Tolerancia);

           % Guarda los valores de los índices de RPM (1) y de Av (2) que
           % cumplen la tolerancia especificada siempre y cuando el caudal
           % sea igual o superior a Qmin.
           for j=1:length(Indices)

              if mc(i)>=Parametros.Restricciones_Caudal.Min
                 PtosOp(k,1)=i;
                 PtosOp(k,2)=Indices(j);
                 PtosOp(k,3)=mc(i)/1000;
                 k=k+1;
              end

           end

        end

     else

        PtosOp(1,1)=0;PtosOp(1,2)=0;PtosOp(1,3)=0;
        P1=-1; P2=-1; H14=-1; H21=-1; COPc=-1;T1=-1;T2=-1;T3=-1;T4=-1;

     end

  catch

     PtosOp(1,1)=0;PtosOp(1,2)=0;PtosOp(1,3)=0;
     P1=-1; P2=-1; H14=-1; H21=-1; COPc=-1;T1=-1;T2=-1;T3=-1;T4=-1;

  end

end
```

```
% ------------------------------------------------------------------------------------------
%                Función para la obtención de los puntos de trabajo válidos
%                            en un entorno de un ciclo indicado
%
%       VARIABLES DE ENTRADA
%
%       Parametros_General:         Estructura con los parámetros de configuración generales
%                                   de los que se usan los siguientes:
%
%           Fluido:                 Alguno de los soportados por CoolProp
%           Referencia:             Tipo de referencia: ASHRAE,IIR,NBP o DEF
%           PuntoTrabajo:           Estructura con los siguientes campos:
%
%               TL0:                Temperatura (°C) en el recinto a refrigerar
%               TH0:                Temperatura ambiente (°C)
%               Delta_Te0:          Salto térmico (°C) en el evaporador
%               Delta_Tc0:          Salto térmico (°C) en el condensador
%               Delta_Tsh0:         Sobrecalentamiento (°C)
%               Delta_Tsc0:         Subenfriamiento (°C)
%
%           Barrido_RPM:            Barrido a realizar en RPM
%               Min:                Valor mínimo
%               Paso_Porc           Paso en % sobre el rango efectivo de RPM
%               Paso_Abs            Paso en valor absoluto para RPM
%               Max:                Valor máximo
%
%           Barrido_Av:             Barrido a realizar en Av
%               Min:                Valor mínimo
%               Paso_Porc:          Paso en % sobre el rango efectivo de Av
%               Paso_Abs            Paso en valor absoluto para Av
%               Max:                Valor máximo
%
%           Barrido_DeltaTemp:
%               Rango:              Rango (°C) para cada temperatura (Delta)
%               NPuntos:            N° puntos para cada temperatura (Delta)
%
%           Barrido_TH:
%               Rango:              Rango (°C) de la temperatura TH
%               NPuntos:            N° puntos para la temperatura TH
%
%           Restricciones_Caudal:
%               Min:                Caudal mínimo (g/s) a seleccionar
%               Tolerancia:         Diferencia máxima (g/s) permitida entre los caudales
%                                   del compresor y la válvula
%
%           Param_Compresor:        Parámetros del compresor
%           Param_Valvula:          Parámetros para la válvula
%
%       Fichero_Matlab_Out          Fichero donde se guardarán los datos obtenidos:
%                                   (Ciclos,Tiempo_CPU)
%
%       Fichero_Excel_Out:          Fichero Excel en donde se guardan los datos obtenidos
%
%
%       VARIABLES DE SALIDA
%
%       Ciclos:                     Estructura con los puntos de operación válidos.
%                                   Contiene los siguientes campos:
%
%               RPM:                Vector con los valores de RPM de los datos brutos
%               Av:                 Vector con los valores de apertura (%) de los datos brutos

%               PtosOP_Q:           Vector de tres componentes con los puntos de operación
%                                   aptos encontrados. Contiene tres columnas:
%                                       Col1: Índice para Temp_Test
%                                       Col2: Índice para RPM
%                                       Col3: Índice para Av
%                                       Col4: Caudal (kg/s)
%
%               P1:                 Vector de n filas con la presión (bares) en el evaporador
%                                   para cada una de las filas de Temp_Test
%               P2:                 Vector de n filas con la presión (bares) en el condensador
%                                   para cada una de las filas de Temp_Test
%               T1:                 Vector de n filas con la temperatura (°C) a la entrada
%                                   del compresor para cada una de las filas de Temp_Test
%               T2:                 Vector de n filas con la temperatura (°C) a la salida
%                                   del compresor para cada una de las filas de Temp_Test
%               T3:                 Vector de n filas con la temperatura (°C) a la entrada
%                                   de la válvula de expansión para cada una de las filas de
%                                   Temp_Test
%               T4:                 Vector de n filas con la temperatura (°C) a la salida de
%                                   la válvula de expansión para cada una de las filas
%                                   de Temp_Test
```

```
%               H14:                Vector de n filas con la diferencia de entalpía entre
%                                   los estados 1 y 4 en kJ/kg para cada una de las filas
%                                   de Temp_Test
%               H21:                Vector de n filas con la diferencia de entalpía entre
%                                   los estados 2 y 1 en kJ/kg para cada una de las filas
%                                   de Temp_Test
%               COPc:               Vector de nfilas con el COP de Carnot para cada una de
%                                   las filas de Temp_Test
%
%               Temp_Test:          Matriz con n filas conteniendo las temperaturas empleadas:
%                                      Col1: Delta_Te (ºC)
%                                      Col2: Delta_Tc (ºC)
%                                      Col3: Delta_Tsh (ºC)
%                                      Col4: Delta_Tsc (ºC)
%                                      Col5: TH (ºC)
%
%       Tiempo_CPU:                 Tiempo de CPU utilizado en su obtención
%
% ------------------------------------------------------------------------------------------

function [Ciclos,Tiempo_CPU]= Obt_PtosOp_Entorno_Ciclo (Parametros,Fichero_Matlab_Out,...
                                                        Fichero_Excel_Out)


   tic;

   % Borra el fichero de datos Matlab en caso de existir
   if ~isempty(Fichero_Matlab_Out)
      delete (Fichero_Matlab_Out);
   end

   % Borra el fichero Excel en caso de existir
   if ~isempty(Fichero_Excel_Out)
      delete (Fichero_Excel_Out);
   end

   fprintf ('Inicializando entorno....\n');

   % Si el paso de RPM se ha indicado en % sobre el rango
   if Parametros.Barrido_RPM.Paso_Porc>0 && Parametros.Barrido_RPM.Paso_Abs==0

      % Obtiene el paso en valor absoluto a utilizar en el barrido de RPM
      Parametros.Barrido_RPM.Paso=0.01*Parametros.Barrido_RPM.Paso_Porc*...
                                    (Parametros.Barrido_RPM.Max-Parametros.Barrido_RPM.Min);

      if Parametros.Barrido_RPM.Paso==0
         Parametros.Barrido_RPM.Paso=1;
      end

   else

      Parametros.Barrido_RPM.Paso=Parametros.Barrido_RPM.Paso_Abs;

   end


   % Si el paso de Av se ha indicado en % sobre el rango
   if Parametros.Barrido_Av.Paso_Porc>0 && Parametros.Barrido_Av.Paso_Abs==0

      % Obtiene el paso en valor absoluto a utilizar en el barrido de Av
      Parametros.Barrido_Av.Paso=0.01*Parametros.Barrido_Av.Paso_Porc*...
                                   (Parametros.Barrido_Av.Max-Parametros.Barrido_Av.Min);

      if Parametros.Barrido_Av.Paso==0
         Parametros.Barrido_Av.Paso=1;
      end

   else

      Parametros.Barrido_Av.Paso=Parametros.Barrido_Av.Paso_Abs;

   end

   % Genera los vectores de barrido
   Ciclos.RPM=(Parametros.Barrido_RPM.Min:Parametros.Barrido_RPM.Paso:...
               Parametros.Barrido_RPM.Max);
   Ciclos.Av=(Parametros.Barrido_Av.Min:Parametros.Barrido_Av.Paso:...
              Parametros.Barrido_Av.Max);

   % Genera la matriz con las temperaturas de prueba
```

```
Ciclos.Temp_Test= Obt_Temp_Test (Parametros);

% Inicializa las variables que contendrán los resultados
NCiclos=size(Ciclos.Temp_Test,1);
Ciclos.P1=zeros(NCiclos,1); Ciclos.P2=zeros(NCiclos,1);
Ciclos.H14=zeros(NCiclos,1); Ciclos.H21=zeros(NCiclos,1);
Ciclos.COPc=zeros(NCiclos,1);
Ciclos.T1=zeros(NCiclos,1); Ciclos.T2=zeros(NCiclos,1);
Ciclos.T3=zeros(NCiclos,1); Ciclos.T4=zeros(NCiclos,1);
Ciclos.PtosOp_Q=single.empty;

NPtosEncontrados=0;
fprintf ('N° Ciclos a analizar: %d\n',NCiclos);

% Realiza un recorrido por cada uno de los ciclos buscando los puntos válidos
for i=1:NCiclos

   % Temp_Test(1) -> Delta_Te       % Temp_Test(2) -> Delta_Tc
   % Temp_Test(3) -> Delta_Tsh      % Temp_Test(4) -> Delta_Tsc
   fprintf ('ID_Ciclo: %d/%d',i,NCiclos);

   % Calcula los puntos de trabajo válidos para cada ciclo
   [PtosOp0,Ciclos.P1(i),Ciclos.P2(i),Ciclos.H14(i),Ciclos.H21(i),Ciclos.COPc(i),...
    Ciclos.T1(i),Ciclos.T2(i),Ciclos.T3(i),Ciclos.T4(i)]= ...
   Obt_PtosOP_Ciclo (Ciclos.RPM,Ciclos.Av,Parametros,Ciclos.Temp_Test(i,:));

   % Comprueba el número válido de puntos de operación del ciclo actual
   NPuntosOp0=size(PtosOp0,1);

   % Añade los puntos válidos del ciclo actual a los obtenidos previamente
   if PtosOp0(1,1)>0
       NPtosEncontrados=NPtosEncontrados+NPuntosOp0;
       Ciclos.PtosOp_Q=[Ciclos.PtosOp_Q;[ones(NPuntosOp0,1)*i PtosOp0]];
   end

   fprintf (' (%d)\n',NPtosEncontrados);

end

Tiempo_CPU=toc;

% Guarda todas las variables para un posterior análisis en el fichero
% indicado
if ~isempty(Fichero_Matlab_Out)
   fprintf ('Guardando variables en fichero Matlab.....\n');
   save (Fichero_Matlab_Out,'Ciclos','Tiempo_CPU');
end

% Exporta los puntos de operación obtenidos a Excel
if ~isempty(Ciclos.PtosOp_Q)

   if ~isempty(Fichero_Excel_Out)

      fprintf ('Guardando los puntos de operación en el fichero Excel....\n');

      % Comprueba si solo se ha seleccionado una RPM
      if length(Ciclos.RPM)>1
         RPM_Aux=Ciclos.RPM(Ciclos.PtosOp_Q(:,2))';
      else
         RPM_Aux=Ciclos.RPM(Ciclos.PtosOp_Q(:,2));
      end

      % Comprueba si solo se ha seleccionado una Av
      if length(Ciclos.Av)>1
         Av_Aux=Ciclos.Av(Ciclos.PtosOp_Q(:,3))';
      else
         Av_Aux=Ciclos.Av(Ciclos.PtosOp_Q(:,3));
      end

      Tabla_PtosOp = ...
           table([Ciclos.PtosOp_Q(:,1)...
                  RPM_Aux Av_Aux Ciclos.PtosOp_Q(:,4)*1000 ...
                  Ciclos.P1(Ciclos.PtosOp_Q(:,1)) ...
                  Ciclos.P2(Ciclos.PtosOp_Q(:,1)) ...
                  Ciclos.H14(Ciclos.PtosOp_Q(:,1)).*Ciclos.PtosOp_Q(:,4)*1000 ...
                  Ciclos.H21(Ciclos.PtosOp_Q(:,1)).*Ciclos.PtosOp_Q(:,4)*1000 ...
                  Ciclos.H14(Ciclos.PtosOp_Q(:,1))./Ciclos.H21(Ciclos.PtosOp_Q(:,1)) ...
                  Ciclos.COPc(Ciclos.PtosOp_Q(:,1)) ...
                  (Ciclos.H14(Ciclos.PtosOp_Q(:,1))./Ciclos.H21(Ciclos.PtosOp_Q(:,1)))./...
                   Ciclos.COPc(Ciclos.PtosOp_Q(:,1)) ...
                  Ciclos.T1(Ciclos.PtosOp_Q(:,1)) Ciclos.T2(Ciclos.PtosOp_Q(:,1)) ...
                  Ciclos.T3(Ciclos.PtosOp_Q(:,1)) Ciclos.T4(Ciclos.PtosOp_Q(:,1)) ...
                  Ciclos.Temp_Test(Ciclos.PtosOp_Q(:,1),1) ...
```

```
                    Ciclos.Temp_Test(Ciclos.PtosOp_Q(:,1),2) ...
                    Ciclos.Temp_Test(Ciclos.PtosOp_Q(:,1),3) ...
                    Ciclos.Temp_Test(Ciclos.PtosOp_Q(:,1),4)...
                    Ciclos.Temp_Test(Ciclos.PtosOp_Q(:,1),5)]);

        writetable(Tabla_PtosOp,Fichero_Excel_Out,'Sheet',1,'Range','A1');

      end

   else
      fprintf ('No se han encontrado puntos de operación válidos.\n');
   end

   Tiempo_CPU_H=fix(Tiempo_CPU/3600);
   Tiempo_CPU_M=fix((Tiempo_CPU-Tiempo_CPU_H*3600)/60);
   Tiempo_CPU_S=fix(Tiempo_CPU-Tiempo_CPU_H*3600-Tiempo_CPU_M*60);
   fprintf('Tiempo CPU: %2dH:%2dM:%2dS\n',Tiempo_CPU_H,Tiempo_CPU_M,Tiempo_CPU_S);
   fprintf ('Proceso finalizado correctamente.\n');

end
```

```
% ------------------------------------------------------------------------------------------
%                       Función para la obtención de la matriz con las
%                       temperaturas a emplear en las simulaciones
%
%       VARIABLES DE ENTRADA
%
%       Parametros:                 Estructura con los parámetros de configuración generales
%                                   de los que se usan los siguientes:
%
%           PuntoTrabajo:           Estructura con los siguientes campos:
%
%               TL0:                Temperatura (°C) en el recinto a refrigerar
%               TH0:                Temperatura ambiente (°C)
%               Delta_Te0:          Salto térmico (°C) en el evaporador
%               Delta_Tc0:          Salto térmico (°C) en el condensador
%               Delta_Tsh0:         Sobrecalentamiento (°C)
%               Delta_Tsc0:         Subenfriamiento (°C)
%
%           Barrido_DeltaTemp:
%               Rango:              Rango (°C) para cada temperatura (Delta)
%               NPuntos:            N° puntos para cada temperatura (Delta)
%
%           Barrido_TH:
%               Rango:              Rango (°C) de la temperatura TH
%               NPuntos:            N° puntos para la temperatura TH
%
%       VARIABLES DE SALIDA
%
%       Temp_Test:                  Matriz con n filas conteniendo las temperaturas a utilizar
%                                      Col1: Delta_Te (°C)
%                                      Col2: Delta_Tc (°C)
%                                      COl3: Delta_Tsh (°C)
%                                      Col4: Delta_Tsc (°C)
%                                      COl5: TH (°C)
%
% ------------------------------------------------------------------------------------------

function [Temp_Test]= Obt_Temp_Test (Parametros)

   % Si el rango de TH es cero supone que solo se tratará el de operación (TH0)
   if Parametros.Barrido_TH.Rango==0
       Parametros.Barrido_TH.NPuntos=1;
   end

   % Si el rango de DeltaTemp es cero supone que solo se tratará el de operación
   if Parametros.Barrido_DeltaTemp.Rango==0
       Parametros.Barrido_DeltaTemp.NPuntos=1;
   end

   % Si hay que realizar un barrido de TH
   if Parametros.Barrido_TH.NPuntos>1

      % Obtiene el paso a utilizar para TB
      Salto_TH=Parametros.Barrido_TH.Rango./(Parametros.Barrido_TH.NPuntos-1);

      % Calcula el valor mínimo para TH
      TH_min=Parametros.PuntoTrabajo.TH0-Parametros.Barrido_TH.Rango/2;

      % Genera un vector con las temperaturas TH de prueba
      TH_Test=(TH_min:Salto_TH:TH_min+Parametros.Barrido_TH.Rango);

   else
      TH_Test=Parametros.PuntoTrabajo.TH0;
   end


   % Si hay que hacer un barrido para DeltaTemp
   if Parametros.Barrido_DeltaTemp.NPuntos>1

      % Obtiene el paso a utilizar en las temperaturas
      Salto_Temp=Parametros.Barrido_DeltaTemp.Rango./(Parametros.Barrido_DeltaTemp.NPuntos-1);

      % Calcula el valor mínimo para cada parámetro
      Delta_Te_min=Parametros.PuntoTrabajo.Delta_Te0-Parametros.Barrido_DeltaTemp.Rango/2;
      Delta_Tc_min=Parametros.PuntoTrabajo.Delta_Tc0-Parametros.Barrido_DeltaTemp.Rango/2;
      Delta_Tsh_min=Parametros.PuntoTrabajo.Delta_Tsh0-Parametros.Barrido_DeltaTemp.Rango/2;
      Delta_Tsc_min=Parametros.PuntoTrabajo.Delta_Tsc0-Parametros.Barrido_DeltaTemp.Rango/2;

      % Acota los valores mínimos
      if Delta_Te_min<10^-4 Delta_Te_min=10^-4;end;
      if Delta_Tc_min<10^-4 Delta_Tc_min=10^-4;end;
      if Delta_Tsh_min<10^-4 Delta_Tsh_min=10^-4;end;
      if Delta_Tsc_min<10^-4 Delta_Tsc_min=10^-4; end;
```

```
      % Genera un vector con cada una de las cuatro temperaturas
      Delta_Te=(Delta_Te_min:Salto_Temp:Delta_Te_min+Parametros.Barrido_DeltaTemp.Rango);
      Delta_Tc=(Delta_Tc_min:Salto_Temp:Delta_Tc_min+Parametros.Barrido_DeltaTemp.Rango);
      Delta_Tsh=(Delta_Tsh_min:Salto_Temp:Delta_Tsh_min+Parametros.Barrido_DeltaTemp.Rango);
      Delta_Tsc=(Delta_Tsc_min:Salto_Temp:Delta_Tsc_min+Parametros.Barrido_DeltaTemp.Rango);

      % Inicializa la matriz que contendrá las temperaturas de prueba
      Temp_Test=zeros(Parametros.Barrido_DeltaTemp.NPuntos^4*length(TH_Test),5);

      % Guarda las temperaturas de prueba en Temp_Test
      Indice=1;
      for i=1:Parametros.Barrido_DeltaTemp.NPuntos
         for j=1:Parametros.Barrido_DeltaTemp.NPuntos
            for k=1:Parametros.Barrido_DeltaTemp.NPuntos
               for l=1:Parametros.Barrido_DeltaTemp.NPuntos
                  for m=1:length(TH_Test)
                     Temp_Test(Indice,1)=Delta_Te(i);
                     Temp_Test(Indice,2)=Delta_Tc(j);
                     Temp_Test(Indice,3)=Delta_Tsh(k);
                     Temp_Test(Indice,4)=Delta_Tsc(l);
                     Temp_Test(Indice,5)=TH_Test(m);
                     Indice=Indice+1;
                  end
               end
            end
         end
      end

   else

      % Inicializa el vector que contendrá las temperaturas de prueba
      Temp_Test=zeros(length(TH_Test),5);

      % Genera la matriz de temperaturas de prueba
      for Indice=1:length(TH_Test)
         Temp_Test(Indice,1)=Parametros.PuntoTrabajo.Delta_Te0;
         Temp_Test(Indice,2)=Parametros.PuntoTrabajo.Delta_Tc0;
         Temp_Test(Indice,3)=Parametros.PuntoTrabajo.Delta_Tsh0;
         Temp_Test(Indice,4)=Parametros.PuntoTrabajo.Delta_Tsc0;
         Temp_Test(Indice,5)=TH_Test(Indice);
      end

   end

end
```

```
% ------------------------------------------------------------------------------------------
%                Función para la obtención de las matrices SSGM y GRGA
%
%       VARIABLES DE ENTRADA
%
%       Datos:                      Estructura con los ciclos procesados con los
%                                   siguientes campos: RPM0, AV0 y TH. Cada
%                                   uno de ellos es, a su vez, una
%                                   estructura con los siguientes campos:
%
%           Vector_Abciaas:         Vector con los valores de abcisas correspondientes a
%                                   los datos
%           TiempoCPU:              Tiempo empleado en la obtención de los datos
%           K:                      Ganancia de cada una de las variables con respecto a RPM
%                                   o Av.
%
%           m,P1,P2,Qe,T1,T2,T3,    Cada una de ellas contiene cuatro columnas: (Med,Min,Max,N)
%           T4,DeltaTe,DeltaTc,
%           DeltaTsh,DeltaTsc
%
%       Variables:                  Variables a incluir en la SSGM y GRGA
%       Variables_Calculadas:       Admite hasta tres variables calculadas con los
%                                   siguientes campos:
%
%           Aux1.BoolActivar:       Si hay que tenerla en cuenta
%           Aux1.Coef:              Vector con coeficientes con respecto al siguiente campo
%           Aux1.Nombre:            Vector con los nombres de las variables ligadas a
%                                   los coeficientes.
%           Aux2.BoolActivar:       Si hay que tenerla en cuenta
%           Aux2.Coef:              Vector con coeficientes con respecto al siguiente campo
%           Aux2.Nombre:            Vector con los nombres de las variables ligadas a
%                                   los coeficientes.
%
%           Aux3.BoolActivar:       Si hay que tenerla en cuenta
%           Aux3.Coef:              Vector con coeficientes con respecto al siguiente campo
%           Aux3.Nombre:            Vector con los nombres de las variables ligadas a
%                                   los coeficientes.
%
%       VARIABLES DE SALIDA
%
%       VControladas:               Vector con las variables candidatas a ser controladas
%       VManipulables:              Vector con las variables manipulables
%       SSGM:                       Matriz SSGM
%       GRGA:                       Matriz GRGA
%
% ------------------------------------------------------------------------------------------

function [VControladas,VManipulables,SSGM,GRGA]=Procesar_SSGM_GRGA (Datos,Variables,...
                                                                    Variables_Calculadas)

   % Fija el texto para las variables manipulables
   VMan={'RPM','Av'};
   VManipulables=[VMan(1);VMan(2)];

   % Inicializa las variables
   Indice_SSGM=1;
   VCtr={'m','Qe','W','P1','P2','nCarnot','COP','COPc','T1','T2','T3','T4',...
         'DeltaTe','DeltaTc','DeltaTsh','DeltaTsc','Aux1','Aux2','Aux3'};


   % Si hay que procesar m
   if Variables.m

      SSGM(Indice_SSGM,:)=[Datos.AV0.K.m;Datos.RPM0.K.m];
      VControladas(Indice_SSGM)=VCtr(1);
      Indice_SSGM=Indice_SSGM+1;

   end

   % Si hay que procesar Qe
   if Variables.Qe

      SSGM(Indice_SSGM,:)=[Datos.AV0.K.Qe;Datos.RPM0.K.Qe];
      VControladas(Indice_SSGM)=VCtr(2);
      Indice_SSGM=Indice_SSGM+1;

   end

   % Si hay que procesar W
   if Variables.W

      SSGM(Indice_SSGM,:)=[Datos.AV0.K.W;Datos.RPM0.K.W];
      VControladas(Indice_SSGM)=VCtr(3);
```

```
    Indice_SSGM=Indice_SSGM+1;

end

% Si hay que procesar P1
if Variables.P1

    SSGM(Indice_SSGM,:)=[Datos.AV0.K.P1;Datos.RPM0.K.P1];
    VControladas(Indice_SSGM)=VCtr(4);
    Indice_SSGM=Indice_SSGM+1;

end

% Si hay que procesar P2
if Variables.P2

    SSGM(Indice_SSGM,:)=[Datos.AV0.K.P2;Datos.RPM0.K.P2];
    VControladas(Indice_SSGM)=VCtr(5);
    Indice_SSGM=Indice_SSGM+1;

end

% Si hay que procesar nCarnot
if Variables.nCarnot

    SSGM(Indice_SSGM,:)=[Datos.AV0.K.nCarnot;Datos.RPM0.K.nCarnot];
    VControladas(Indice_SSGM)=VCtr(6);
    Indice_SSGM=Indice_SSGM+1;

end

% Si hay que procesar COP
if Variables.COP

    SSGM(Indice_SSGM,:)=[Datos.AV0.K.COP;Datos.RPM0.K.COP];
    VControladas(Indice_SSGM)=VCtr(7);
    Indice_SSGM=Indice_SSGM+1;

end

% Si hay que procesar COPc
if Variables.COPc

    SSGM(Indice_SSGM,:)=[Datos.AV0.K.COPc;Datos.RPM0.K.COPc];
    VControladas(Indice_SSGM)=VCtr(8);
    Indice_SSGM=Indice_SSGM+1;

end

% Si hay que procesar T1
if Variables.T1

    SSGM(Indice_SSGM,:)=[Datos.AV0.K.T1;Datos.RPM0.K.T1];
    VControladas(Indice_SSGM)=VCtr(9);
    Indice_SSGM=Indice_SSGM+1;

end

% Si hay que procesar T2
if Variables.T2

    SSGM(Indice_SSGM,:)=[Datos.AV0.K.T2;Datos.RPM0.K.T2];
    VControladas(Indice_SSGM)=VCtr(10);
    Indice_SSGM=Indice_SSGM+1;

end

% Si hay que procesar T3
if Variables.T3

    SSGM(Indice_SSGM,:)=[Datos.AV0.K.T3;Datos.RPM0.K.T3];
    VControladas(Indice_SSGM)=VCtr(11);
    Indice_SSGM=Indice_SSGM+1;

end

% Si hay que procesar T4
if Variables.T4

    SSGM(Indice_SSGM,:)=[Datos.AV0.K.T4;Datos.RPM0.K.T4];
```

```
      VControladas(Indice_SSGM)=VCtr(12);
      Indice_SSGM=Indice_SSGM+1;

   end

   % Si hay que procesar DeltaTe
   if Variables.DeltaTe

      SSGM(Indice_SSGM,:)=[Datos.AV0.K.DeltaTe;Datos.RPM0.K.DeltaTe];
      VControladas(Indice_SSGM)=VCtr(13);
      Indice_SSGM=Indice_SSGM+1;

   end

   % Si hay que procesar DeltaTc
   if Variables.DeltaTc

      SSGM(Indice_SSGM,:)=[Datos.AV0.K.DeltaTc;Datos.RPM0.K.DeltaTc];
      VControladas(Indice_SSGM)=VCtr(14);
      Indice_SSGM=Indice_SSGM+1;

   end

   % Si hay que procesar DeltaTsh
   if Variables.DeltaTsh

      SSGM(Indice_SSGM,:)=[Datos.AV0.K.DeltaTsh;Datos.RPM0.K.DeltaTsh];
      VControladas(Indice_SSGM)=VCtr(15);
      Indice_SSGM=Indice_SSGM+1;

   end

   % Si hay que procesar DeltaTsc
   if Variables.DeltaTsc

      SSGM(Indice_SSGM,:)=[Datos.AV0.K.DeltaTsc;Datos.RPM0.K.DeltaTsc];
      VControladas(Indice_SSGM)=VCtr(16);
      Indice_SSGM=Indice_SSGM+1;

   end

   % Si hay que procesar la variable Aux1
   if Variables_Calculadas.Aux1.BoolActivar && ...
      isnumeric(Datos.RPM0.K.Aux1) && isnumeric(Datos.AV0.K.Aux1)

      SSGM(Indice_SSGM,:)=[Datos.AV0.K.Aux1;Datos.RPM0.K.Aux1];
      VControladas(Indice_SSGM)=VCtr(17);
      Indice_SSGM=Indice_SSGM+1;

   end

   % Si hay que procesar la variable Aux2
   if Variables_Calculadas.Aux2.BoolActivar && ...
      isnumeric(Datos.RPM0.K.Aux2) && isnumeric(Datos.AV0.K.Aux2)

      SSGM(Indice_SSGM,:)=[Datos.AV0.K.Aux2;Datos.RPM0.K.Aux2];
      VControladas(Indice_SSGM)=VCtr(18);
      Indice_SSGM=Indice_SSGM+1;

   end

   % Si hay que procesar la variable Aux3
   if Variables_Calculadas.Aux3.BoolActivar && ...
      isnumeric(Datos.RPM0.K.Aux3) && isnumeric(Datos.AV0.K.Aux3)

      SSGM(Indice_SSGM,:)=[Datos.AV0.K.Aux3;Datos.RPM0.K.Aux3];
      VControladas(Indice_SSGM)=VCtr(19);

   end

   VControladas=VControladas';
   GRGA=SSGM.*(pinv(SSGM))';

end
```

```
% ------------------------------------------------------------------------------------
%           Programa para la eliminación de datos nulos y obtención de la matriz
%           de ganancias para el barrido de RPM/Av
%
%       VARIABLES DE ENTRADA
%
%          Ciclos_Grid:             Estructura con un campo por cada una de las variables
%                                   ajustadas al grid:
%                                   (m, P1, P2, Qe, W, nCarnot, COP,COPc, T1, T2, T3, T4,
%                                   DeltaTe, DeltaTc, DeltaTsh, DeltaTsc y TH).
%                                   Cada uno de ellos es, a su vez, otra estructura con
%                                   estos campos:
%
%               Valores:            Matriz con los datos ajustados al Grid. Consta de 4
%                                   columnas:
%                                       Col1: Valor medio
%                                       Col2: Valor mínimo
%                                       Col3: Valor máximo
%                                       Col4: Número de datos brutos ajustados al punto
%               Min:                Valor mínimo encontrado en los datos brutos
%               Max:                Valor máximo encontrado en los datos brutos
%               PtosSuperan_Porc    Índices con los puntos que han superado PorcLimite
%
%       Parametros_General:         Estructura con los parámetros configuración generales
%                                   de los que se usan los siguientes:
%
%           Barrido_RPM:            Barrido a realizar en RPM
%               Min:                Valor mínimo
%               Paso_Porc           Paso en % sobre el rango efectivo de RPM
%               Paso_Abs            Paso en valor absoluto para RM
%               Max:                Valor máximo
%           Barrido_Av:             Barrido a realizar en Av
%               Min:                Valor mínimo
%               Paso_Porc:          Paso en % sobre el rango efectivo de Av
%               Paso_Abs            Paso en valor absoluto para Av
%               Max:                Valor máximo
%
%        Parametros_Grid:           Estructura con los parámetros de configuración
%                                   para el ajuste al GRID. Se utilizan los
%                                   siguientes campos:
%
%               Paso.RPM_Porc:      Paso en % sobre el rango de barrido para RPM
%               Paso.RPM_Abs:       Paso en valor absoluto para RPM. Utilizar uno u otro
%                                   y colocar el otro a cero.
%
%               Paso.Av_Porc:       Paso en % sobre el rango de barrido para Av
%               Paso.Av_Abs:        Paso en valor absoluto para Av. Utilizar uno u otro
%                                   y colocar el otro a cero.
%
%               Variables:          Estructura con dos campos por cada variable a tratar.
%                                   El primero indica si debe ajustarse al grid y el
%                                   segundo el porcentaje máximo de desviación antes
%                                   de incluirlo en los puntos a comprobar
%                   m,m_PorcentajeMax
%                   P1,P1_PorcentajeMax, P2,P2_PorcentajeMax
%                   Qe,Qe_PorcentajeMax, W,W_PorcentajeMax
%                   nCarnot.nCarnot_PorcentajeMax,COP,COP_PorcentajeMax
%                   COPc,COPc_PorcentajeMax
%                   T1,T1_PorcentajeMax,T2,T2_PorcentajeMax
%                   T3,T3_PorcentajeMax,T4,T4_PorcentajeMax
%                   DeltaTe,DeltaTe_PorcentajeMax,DeltaTc,DeltaTc_PorcentajeMax
%                   DeltaTsh,DeltaTsh_PorcentajeMax,DeltaTsc,DeltaTsc_PorcentajeMax
%                   TH,TH_PorcentajeMax
%
%       Parametros_Analisis:        Estructura con los siguientes campos:
%
%            PorcentajeK:           Porcentaje alrededor del punto operación a utilizar
%                                   para el cálculo de las ganancias
%           Variables_Calculadas:   Admite hasta tres variables calculadas con los
%                                   siguientes campos:
```

```
%
%           Aux1.BoolActivar:       Si hay que tenerla en cuenta
%           Aux1.Coef:              Vector con coeficientes con respecto siguiente campo
%           Aux1.Nombre:            Vector con los nombres de las variables ligadas a
%                                   los coeficientes.
%           Aux2.BoolActivar:       Si hay que tenerla en cuenta
%           Aux2.Coef:              Vector con coeficientes con respecto siguiente campo
%           Aux2.Nombre:            Vector con los nombres de las variables ligadas a
%                                   los coeficientes.
%
%           Aux3.BoolActivar:       Si hay que tenerla en cuenta
%           Aux3.Coef:              Vector con coeficientes con respecto siguiente campo
%           Aux3.Nombre:            Vector con los nombres de las variables ligadas a
%                                   los coeficientes.
%       BoolRPM0:                   Indica si la variable que está fija es RPM
%       TiempoCPU:                  Tiempo utilizado para los cálculos
%
%       VARIABLES DE SALIDA
%
%
%       DatosN_Grid:                Estructura con los ciclos procesados con los
%                                   siguientes campos:
%
%           Vector_Abcisas:         Vector con los valores de abcisas correspondientes a
%                                   los datos
%           TiempoCPU:              Tiempo empleado en la obtención de los datos
%           K:                      Ganancia de cada una de las variables respecto a RPM
%                                   o Av.
%
%           m,P1,P2,Qe,T1,T2,T3,    Cada una de ellas contiene cuatro columnas:
(Med,Min,Max,N)
%           T4,DeltaTe,DeltaTc,
%           DeltaTsh,DeltaTsc
% ------------------------------------------------------------------------------------


function [DatosN_Grid]=Procesar_Variables_Ciclos_RPMAV
(Ciclos_Grid,Parametros_General,...
                                                    Parametros_Grid,...
                                                    Parametros_Analisis,BoolRPM0,...
                                                    TiempoCPU)


   % Obtiene los parámetros
   Vector_RPM=Ciclos_Grid.Vector_RPM;
   Vector_Av=Ciclos_Grid.Vector_Av;
   Datos_Grid=Ciclos_Grid.Datos;
   RPM0=Parametros_General.Barrido_RPM.RPM0;
   Av0=Parametros_General.Barrido_Av.Av0;
   PasoGrid=Parametros_Grid.Paso;
   Variables=Parametros_Grid.Variables;
   Variables_Calculadas=Parametros_Analisis.Variables_Calculadas;
   PorcentajeK=Parametros_Analisis.PorcentajeK;

   % Si el paso de RPM para el grid se ha indicado en % sobre el rango
   if PasoGrid.RPM_Porc>0 && PasoGrid.RPM_Abs==0

      % Obtiene el paso en valor absoluto a utilizar en el barrido de RPM
      PasoGrid.RPM=0.01*PasoGrid.RPM_Porc*(Parametros.Barrido_RPM.Max-...
                                           Parametros.Barrido_RPM.Min);

   else

      PasoGrid.RPM=PasoGrid.RPM_Abs;

   end


   % Si el paso de Av para el grid se ha indicado en % sobre el rango
   if PasoGrid.Av_Porc>0 && PasoGrid.Av_Abs==0

      % Obtiene el paso en valor absoluto a utilizar en el barrido de RPM
```

```
      PasoGrid.Av=0.01*PasoGrid.Av_Porc*(Parametros.Barrido_Av.Max-...
                                         Parametros.Barrido_Av.Min);

   else

      PasoGrid.Av=PasoGrid.Av_Abs;

   end


   Indice_RPM0=0; Indice_Av0=0;

   % RPM se mantiene fija y se está realizando un barrido sobre Av
   if BoolRPM0

      % Obtiene los valores de apertura mínimo y máximo para el cálculo de la matriz de
      % ganancias
      Av_MinK=(1-0.01*PorcentajeK)*Av0;
      if Av_MinK<Vector_Av(1)
          Av_MinK=Vector_Av(1);
      end

      Av_MaxK=(1+0.01*PorcentajeK)*Av0;
      if Av_MaxK>Vector_Av(length(Vector_Av))
          Av_MaxK=Vector_Av(length(Vector_Av));
      end

      % Obtiene el índice de RPM0 y lo acota
      [Indice_RPM0,Aux]=Grid_Obt_Indices(RPM0,Av0,Vector_RPM(1),Vector_Av(1),...
                                         PasoGrid.RPM,PasoGrid.Av);
      if Indice_RPM0>length(Vector_RPM)
         Indice_RPM0=length(Vector_RPM);
      end

      Indice_Av0=0;


   else % Av se mantiene fija y se realiza un barrido sobre RPM

      % Obtiene los valores de RPM mínimo y máximo para el cálculo de la matriz de
      % ganancias
      RPM_MinK=(1-0.01*PorcentajeK)*RPM0;
      if RPM_MinK<Vector_RPM(1)
          RPM_MinK=Vector_RPM(1);
      end

      RPM_MaxK=(1+0.01*PorcentajeK)*RPM0;
      if RPM_MaxK>Vector_RPM(length(Vector_RPM))
          RPM_MaxK=Vector_RPM(length(Vector_RPM));
      end


      % Obtiene el índice de AV0 y lo acota
      [Aux,Indice_Av0]=Grid_Obt_Indices(RPM0,Av0,Vector_RPM(1),Vector_Av(1),...
                                        PasoGrid.RPM,PasoGrid.Av);
      if Indice_Av0>length(Vector_Av)
         Indice_Av0=length(Vector_Av);
      end

      Indice_RPM0=0;

   end


   % Obtiene los índices válidos para las variables para el caso de RPM=RPM0 y Av
variable
   BoolEncontrado=false;
   if Indice_RPM0>0

      VectorAux=Vector_Av;
```

```
Indices_Validos_RPM=Indice_RPM0;

if Variables.m
   Indices_Validos_Av=find(Datos_Grid.m.Valores(Indice_RPM0,:,4));
   BoolEncontrado=true;
end

if Variables.Qe && ~BoolEncontrado
   Indices_Validos_Av=find(Datos_Grid.Qe.Valores(Indice_RPM0,:,4));
   BoolEncontrado=true;
end

if Variables.W && ~BoolEncontrado
   Indices_Validos_Av=find(Datos_Grid.W.Valores(Indice_RPM0,:,4));
   BoolEncontrado=true;
end

if Variables.P1 && ~BoolEncontrado
   Indices_Validos_Av=find(Datos_Grid.P1.Valores(Indice_RPM0,:,4));
   BoolEncontrado=true;
end

if Variables.P2 && ~BoolEncontrado
   Indices_Validos_Av=find(Datos_Grid.P2.Valores(Indice_RPM0,:,4));
   BoolEncontrado=true;
end

if Variables.nCarnot && ~BoolEncontrado
   Indices_Validos_Av=find(Datos_Grid.nCarnot.Valores(Indice_RPM0,:,4));
   BoolEncontrado=true;
end

if Variables.COP && ~BoolEncontrado
   Indices_Validos_Av=find(Datos_Grid.COP.Valores(Indice_RPM0,:,4));
   BoolEncontrado=true;
end

if Variables.COPc && ~BoolEncontrado
   Indices_Validos_Av=find(Datos_Grid.COPc.Valores(Indice_RPM0,:,4));
   BoolEncontrado=true;
end

if Variables.T1 && ~BoolEncontrado
   Indices_Validos_Av=find(Datos_Grid.T1.Valores(Indice_RPM0,:,4));
   BoolEncontrado=true;
end

if Variables.T2 && ~BoolEncontrado
   Indices_Validos_Av=find(Datos_Grid.T2.Valores(Indice_RPM0,:,4));
   BoolEncontrado=true;
end

if Variables.T3 && ~BoolEncontrado
   Indices_Validos_Av=find(Datos_Grid.T3.Valores(Indice_RPM0,:,4));
   BoolEncontrado=true;
end

if Variables.T4 && ~BoolEncontrado
   Indices_Validos_Av=find(Datos_Grid.T4.Valores(Indice_RPM0,:,4));
   BoolEncontrado=true;
end

if Variables.DeltaTe && ~BoolEncontrado
   Indices_Validos_Av=find(Datos_Grid.DeltaTe.Valores(Indice_RPM0,:,4));
   BoolEncontrado=true;
end

if Variables.DeltaTc && ~BoolEncontrado
   Indices_Validos_Av=find(Datos_Grid.DeltaTc.Valores(Indice_RPM0,:,4));
   BoolEncontrado=true;
end

if Variables.DeltaTsh && ~BoolEncontrado
```

```
            Indices_Validos_Av=find(Datos_Grid.DeltaTsh.Valores(Indice_RPM0,:,4));
            BoolEncontrado=true;
        end

        if Variables.DeltaTsc && ~BoolEncontrado
            Indices_Validos_Av=find(Datos_Grid.DeltaTsc.Valores(Indice_RPM0,:,4));
            BoolEncontrado=true;
        end

        if BoolEncontrado

            % Guarda el vector de abcisas
            DatosN_Grid.Vector_Abcisas=VectorAux(Indices_Validos_Av);

            % Guarda el tiempo en segundos empleado en la obtención de los puntos
            DatosN_Grid.TiempoCPU=TiempoCPU/3600;

            % Obtiene los indices a utilizar para el cálculo de la matriz de
            % ganancias (+-PorcentajeK sobre el punto nominal de operación de
            % Av)
            Indice_MinK=find(DatosN_Grid.Vector_Abcisas>=Av_MinK,1,'first');
            Indice_MaxK=find(DatosN_Grid.Vector_Abcisas<=Av_MaxK,1,'last');

            if Indice_MinK>1
                if abs(DatosN_Grid.Vector_Abcisas(Indice_MinK-1)-Av_MinK)<...
                   abs(DatosN_Grid.Vector_Abcisas(Indice_MinK)-Av_MinK)
                   Indice_MinK=Indice_MinK-1;
                end
            end

            if Indice_MaxK<length(DatosN_Grid.Vector_Abcisas)
                if abs(DatosN_Grid.Vector_Abcisas(Indice_MaxK+1)-Av_MaxK)<...
                   abs(DatosN_Grid.Vector_Abcisas(Indice_MaxK)-Av_MaxK)
                   Indice_MaxK=Indice_MaxK+1;
                end
            end

        end

    end


    % Obtiene los índices válidos para las variables para el caso de Av=Av0 y RPM
variable.
    BoolEncontrado=false;
    if Indice_Av0>0

        VectorAux=Vector_RPM;
        Indices_Validos_Av=Indice_Av0;

        if Variables.m
            Indices_Validos_RPM=find(Datos_Grid.m.Valores(:,Indice_Av0,4));
            BoolEncontrado=true;
        end

        if Variables.Qe && ~BoolEncontrado
            Indices_Validos_RPM=find(Datos_Grid.Qe.Valores(:,Indice_Av0,4));
            BoolEncontrado=true;
        end

        if Variables.W && ~BoolEncontrado
            Indices_Validos_RPM=find(Datos_Grid.W.Valores(:,Indice_Av0,4));
            BoolEncontrado=true;
        end

        if Variables.P1 && ~BoolEncontrado
            Indices_Validos_RPM=find(Datos_Grid.P1.Valores(:,Indice_Av0,4));
            BoolEncontrado=true;
        end
```

```
if Variables.P2 && ~BoolEncontrado
   Indices_Validos_RPM=find(Datos_Grid.P2.Valores(:,Indice_Av0,4));
   BoolEncontrado=true;
end

if Variables.nCarnot && ~BoolEncontrado
   Indices_Validos_RPM=find(Datos_Grid.nCarnot.Valores(:,Indice_Av0,4));
   BoolEncontrado=true;
end

if Variables.COP && ~BoolEncontrado
   Indices_Validos_RPM=find(Datos_Grid.COP.Valores(:,Indice_Av0,4));
   BoolEncontrado=true;
end

if Variables.COPc && ~BoolEncontrado
   Indices_Validos_RPM=find(Datos_Grid.COPc.Valores(:,Indice_Av0,4));
   BoolEncontrado=true;
end

if Variables.T1 && ~BoolEncontrado
   Indices_Validos_RPM=find(Datos_Grid.T1.Valores(:,Indice_Av0,4));
   BoolEncontrado=true;
end

if Variables.T2 && ~BoolEncontrado
   Indices_Validos_RPM=find(Datos_Grid.T2.Valores(:,Indice_Av0,4));
   BoolEncontrado=true;
end

if Variables.T3 && ~BoolEncontrado
   Indices_Validos_RPM=find(Datos_Grid.T3.Valores(:,Indice_Av0,4));
   BoolEncontrado=true;
end

if Variables.T4 && ~BoolEncontrado
   Indices_Validos_RPM=find(Datos_Grid.T4.Valores(:,Indice_Av0,4));
   BoolEncontrado=true;
end

if Variables.DeltaTe && ~BoolEncontrado
   Indices_Validos_RPM=find(Datos_Grid.DeltaTe.Valores(:,Indice_Av0,4));
   BoolEncontrado=true;
end

if Variables.DeltaTc && ~BoolEncontrado
   Indices_Validos_RPM=find(Datos_Grid.DeltaTc.Valores(:,Indice_Av0,4));
   BoolEncontrado=true;
end

if Variables.DeltaTsh && ~BoolEncontrado
   Indices_Validos_RPM=find(Datos_Grid.DeltaTsh.Valores(:,Indice_Av0,4));
   BoolEncontrado=true;
end

if Variables.DeltaTsc && ~BoolEncontrado
   Indices_Validos_RPM=find(Datos_Grid.DeltaTsc.Valores(:,Indice_Av0,4));
   BoolEncontrado=true;
end

if BoolEncontrado

   % Guarda el vector de abcisas
   DatosN_Grid.Vector_Abcisas=VectorAux(Indices_Validos_RPM);

   % Guarda el tiempo en segundos empleado en la obtención de los puntos
   DatosN_Grid.TiempoCPU=TiempoCPU/3600;

   % Obtiene los indices a utilizar para el cálculo de la matriz de
   % ganancias (+-PorcentajeK sobre el punto nominal de operación de
   % RPM)
   Indice_MinK=find(DatosN_Grid.Vector_Abcisas>=RPM_MinK,1,'first');
   Indice_MaxK=find(DatosN_Grid.Vector_Abcisas<=RPM_MaxK,1,'last');
```

```
        if Indice_MinK>1
            if abs(DatosN_Grid.Vector_Abcisas(Indice_MinK-1)-RPM_MinK)<...
               abs(DatosN_Grid.Vector_Abcisas(Indice_MinK)-RPM_MinK)
               Indice_MinK=Indice_MinK-1;
            end
        end

        if Indice_MaxK<length(DatosN_Grid.Vector_Abcisas)
            if abs(DatosN_Grid.Vector_Abcisas(Indice_MaxK+1)-RPM_MaxK)<...
               abs(DatosN_Grid.Vector_Abcisas(Indice_MaxK)-RPM_MaxK)
               Indice_MaxK=Indice_MaxK+1;
            end
        end

      end

   end


   % Si hay que procesar m
   if Variables.m

       % Guarda el valor de los datos de la variable
       DatosN_Grid.m(:,:)=Datos_Grid.m.Valores(Indices_Validos_RPM,Indices_Validos_Av,:);
       [NDatos,NColumnas]=size(DatosN_Grid.m);
       if NColumnas~=4
            DatosN_Grid.m= DatosN_Grid.m';
       end

       % Guarda el valor de K para la variable
       Coef=polyfit(DatosN_Grid.Vector_Abcisas(Indice_MinK:Indice_MaxK),...
                    DatosN_Grid.m(Indice_MinK:Indice_MaxK,1)',1);
       DatosN_Grid.K.m=Coef(1);

   end


   % Si hay que procesar P1
   if Variables.P1

       % Guarda el valor de los datos de la variable

DatosN_Grid.P1(:,:)=Datos_Grid.P1.Valores(Indices_Validos_RPM,Indices_Validos_Av,:);
       [NDatos,NColumnas]=size(DatosN_Grid.P1);
       if NColumnas~=4
            DatosN_Grid.P1= DatosN_Grid.P1';
       end

       % Guarda el valor de K para la variable
       Coef=polyfit(DatosN_Grid.Vector_Abcisas(Indice_MinK:Indice_MaxK),...
                    DatosN_Grid.P1(Indice_MinK:Indice_MaxK,1)',1);
       DatosN_Grid.K.P1=Coef(1);

   end


   % Si hay que procesar P2
   if Variables.P2

       % Guarda el valor de los datos de la variable

DatosN_Grid.P2(:,:)=Datos_Grid.P2.Valores(Indices_Validos_RPM,Indices_Validos_Av,:);
       [NDatos,NColumnas]=size(DatosN_Grid.P2);
       if NColumnas~=4
            DatosN_Grid.P2= DatosN_Grid.P2';
       end

       % Guarda el valor de K para la variable
       Coef=polyfit(DatosN_Grid.Vector_Abcisas(Indice_MinK:Indice_MaxK),...
```

```
                    DatosN_Grid.P2(Indice_MinK:Indice_MaxK,1)',1);
        DatosN_Grid.K.P2=Coef(1);

    end


    % Si hay que procesar Qe
    if Variables.Qe

        % Guarda el valor de los datos de la variable

DatosN_Grid.Qe(:,:)=Datos_Grid.Qe.Valores(Indices_Validos_RPM,Indices_Validos_Av,:);
        [NDatos,NColumnas]=size(DatosN_Grid.Qe);
        if NColumnas~=4
             DatosN_Grid.Qe= DatosN_Grid.Qe';
        end

        % Guarda el valor de K para la variable
        Coef=polyfit(DatosN_Grid.Vector_Abcisas(Indice_MinK:Indice_MaxK),...
                    DatosN_Grid.Qe(Indice_MinK:Indice_MaxK,1)',1);
        DatosN_Grid.K.Qe=Coef(1);

    end


    % Si hay que procesar W
    if Variables.W

        % Guarda el valor de los datos de la variable
        DatosN_Grid.W(:,:)=Datos_Grid.W.Valores(Indices_Validos_RPM,Indices_Validos_Av,:);
        [NDatos,NColumnas]=size(DatosN_Grid.W);
        if NColumnas~=4
             DatosN_Grid.W= DatosN_Grid.W';
        end

        % Guarda el valor de K para la variable
        Coef=polyfit(DatosN_Grid.Vector_Abcisas(Indice_MinK:Indice_MaxK),...
                    DatosN_Grid.W(Indice_MinK:Indice_MaxK,1)',1);
        DatosN_Grid.K.W=Coef(1);

    end


    % Si hay que procesar nCarnot
    if Variables.nCarnot

        % Guarda el valor de los datos de la variable

DatosN_Grid.nCarnot(:,:)=Datos_Grid.nCarnot.Valores(Indices_Validos_RPM,Indices_Validos_A
v,:);
        [NDatos,NColumnas]=size(DatosN_Grid.nCarnot);
        if NColumnas~=4
             DatosN_Grid.nCarnot= DatosN_Grid.nCarnot';
        end

        % Guarda el valor de K para la variable
        Coef=polyfit(DatosN_Grid.Vector_Abcisas(Indice_MinK:Indice_MaxK),...
                    DatosN_Grid.nCarnot(Indice_MinK:Indice_MaxK,1)',1);
        DatosN_Grid.K.nCarnot=Coef(1);

    end


    % Si hay que procesar COP
    if Variables.COP

        % Guarda el valor de los datos de la variable

DatosN_Grid.COP(:,:)=Datos_Grid.COP.Valores(Indices_Validos_RPM,Indices_Validos_Av,:);
        [NDatos,NColumnas]=size(DatosN_Grid.COP);
        if NColumnas~=4
             DatosN_Grid.COP= DatosN_Grid.COP';
        end
```

```
          % Guarda el valor de K para la variable
       Coef=polyfit(DatosN_Grid.Vector_Abcisas(Indice_MinK:Indice_MaxK),...
                    DatosN_Grid.COP(Indice_MinK:Indice_MaxK,1)',1);
       DatosN_Grid.K.COP=Coef(1);

   end


   % Si hay que procesar W
   if Variables.COPc
       % Guarda el valor de los datos de la variable

DatosN_Grid.COPc(:,:)=Datos_Grid.COPc.Valores(Indices_Validos_RPM,Indices_Validos_Av,:);
       [NDatos,NColumnas]=size(DatosN_Grid.COPc);
       if NColumnas~=4
            DatosN_Grid.COPc= DatosN_Grid.COPc';
       end

       % Guarda el valor de K para la variable
       Coef=polyfit(DatosN_Grid.Vector_Abcisas(Indice_MinK:Indice_MaxK),...
                    DatosN_Grid.COPc(Indice_MinK:Indice_MaxK,1)',1);
       DatosN_Grid.K.COPc=Coef(1);

   end


   % Si hay que procesar T1
   if Variables.T1

       % Guarda el valor de los datos de la variable

DatosN_Grid.T1(:,:)=Datos_Grid.T1.Valores(Indices_Validos_RPM,Indices_Validos_Av,:);
       [NDatos,NColumnas]=size(DatosN_Grid.T1);
       if NColumnas~=4
            DatosN_Grid.T1= DatosN_Grid.T1';
       end

       % Guarda el valor de K para la variable
       Coef=polyfit(DatosN_Grid.Vector_Abcisas(Indice_MinK:Indice_MaxK),...
                    DatosN_Grid.T1(Indice_MinK:Indice_MaxK,1)',1);
       DatosN_Grid.K.T1=Coef(1);

   end


   % Si hay que procesar T2
   if Variables.T2

       % Guarda el valor de los datos de la variable

DatosN_Grid.T2(:,:)=Datos_Grid.T2.Valores(Indices_Validos_RPM,Indices_Validos_Av,:);
       [NDatos,NColumnas]=size(DatosN_Grid.T2);
       if NColumnas~=4
            DatosN_Grid.T2= DatosN_Grid.T2';
       end

       % Guarda el valor de K para la variable
       Coef=polyfit(DatosN_Grid.Vector_Abcisas(Indice_MinK:Indice_MaxK),...
                    DatosN_Grid.T2(Indice_MinK:Indice_MaxK,1)',1);
       DatosN_Grid.K.T2=Coef(1);

   end


   % Si hay que procesar T3
   if Variables.T3

       % Guarda el valor de los datos de la variable
```

```
DatosN_Grid.T3(:,:)=Datos_Grid.T3.Valores(Indices_Validos_RPM,Indices_Validos_Av,:);
       [NDatos,NColumnas]=size(DatosN_Grid.T3);
       if NColumnas~=4
            DatosN_Grid.T3= DatosN_Grid.T3';
       end

       % Guarda el valor de K para la variable
       Coef=polyfit(DatosN_Grid.Vector_Abcisas(Indice_MinK:Indice_MaxK),...
                    DatosN_Grid.T3(Indice_MinK:Indice_MaxK,1)',1);
       DatosN_Grid.K.T3=Coef(1);

   end


   % Si hay que procesar T4
   if Variables.T4

       % Guarda el valor de los datos de la variable

DatosN_Grid.T4(:,:)=Datos_Grid.T4.Valores(Indices_Validos_RPM,Indices_Validos_Av,:);
       [NDatos,NColumnas]=size(DatosN_Grid.T4);
       if NColumnas~=4
            DatosN_Grid.T4= DatosN_Grid.T4';
       end

       % Guarda el valor de K para la variable
       Coef=polyfit(DatosN_Grid.Vector_Abcisas(Indice_MinK:Indice_MaxK),...
                    DatosN_Grid.T4(Indice_MinK:Indice_MaxK,1)',1);
       DatosN_Grid.K.T4=Coef(1);

   end


   % Si hay que procesar DeltaTe
   if Variables.DeltaTe

       % Guarda el valor de los datos de la variable

DatosN_Grid.DeltaTe(:,:)=Datos_Grid.DeltaTe.Valores(Indices_Validos_RPM,Indices_Validos_A
v,:);
       [NDatos,NColumnas]=size(DatosN_Grid.DeltaTe);
       if NColumnas~=4
            DatosN_Grid.DeltaTe= DatosN_Grid.DeltaTe';
       end

       % Guarda el valor de K para la variable
       Coef=polyfit(DatosN_Grid.Vector_Abcisas(Indice_MinK:Indice_MaxK),...
                    DatosN_Grid.DeltaTe(Indice_MinK:Indice_MaxK,1)',1);
       DatosN_Grid.K.DeltaTe=Coef(1);

   end


   % Si hay que procesar DeltaTc
   if Variables.DeltaTc

       % Guarda el valor de los datos de la variable

DatosN_Grid.DeltaTc(:,:)=Datos_Grid.DeltaTc.Valores(Indices_Validos_RPM,Indices_Validos_A
v,:);
       [NDatos,NColumnas]=size(DatosN_Grid.DeltaTc);
       if NColumnas~=4
            DatosN_Grid.DeltaTc= DatosN_Grid.DeltaTc';
       end

       % Guarda el valor de K para la variable
       Coef=polyfit(DatosN_Grid.Vector_Abcisas(Indice_MinK:Indice_MaxK),...
                    DatosN_Grid.DeltaTc(Indice_MinK:Indice_MaxK,1)',1);
       DatosN_Grid.K.DeltaTc=Coef(1);

   end
```

```
   % Si hay que procesar DeltaTsh
   if Variables.DeltaTsh

       % Guarda el valor de los datos de la variable

DatosN_Grid.DeltaTsh(:,:)=Datos_Grid.DeltaTsh.Valores(Indices_Validos_RPM,Indices_Validos
_Av,:);
       [NDatos,NColumnas]=size(DatosN_Grid.DeltaTsh);
       if NColumnas~=4
            DatosN_Grid.DeltaTsh= DatosN_Grid.DeltaTsh';
       end

       % Guarda el valor de K para la variable
       Coef=polyfit(DatosN_Grid.Vector_Abcisas(Indice_MinK:Indice_MaxK),...
                    DatosN_Grid.DeltaTsh(Indice_MinK:Indice_MaxK,1)',1);
       DatosN_Grid.K.DeltaTsh=Coef(1);

   end


   % Si hay que procesar DeltaTsc
   if Variables.DeltaTsc

       % Guarda el valor de los datos de la variable

DatosN_Grid.DeltaTsc(:,:)=Datos_Grid.DeltaTsc.Valores(Indices_Validos_RPM,Indices_Validos
_Av,:);
       [NDatos,NColumnas]=size(DatosN_Grid.DeltaTsc);
       if NColumnas~=4
            DatosN_Grid.DeltaTsc= DatosN_Grid.DeltaTsc';
       end

       % Guarda el valor de K para la variable
       Coef=polyfit(DatosN_Grid.Vector_Abcisas(Indice_MinK:Indice_MaxK),...
                    DatosN_Grid.DeltaTsc(Indice_MinK:Indice_MaxK,1)',1);
       DatosN_Grid.K.DeltaTsc=Coef(1);

   end

   % Si hay que procesar Aux1
   if Variables_Calculadas.Aux1.BoolActivar

      try

         % Genera el string para el campo calculado
         Aux1_Str='';
         for i=1:length(Variables_Calculadas.Aux1.Coef)

            if i~=1
               Aux1_Str=strcat(Aux1_Str,'+');
            end

            Aux1_Str=strcat(Aux1_Str,num2str(Variables_Calculadas.Aux1.Coef(i)),...

'*','DatosN_Grid.',Variables_Calculadas.Aux1.Nombre(i),'(:,:)');

         end


         % Guarda el valor de los datos de la variable
         DatosN_Grid.Aux1=eval(cell2mat(Aux1_Str));
         [NDatos,NColumnas]=size(DatosN_Grid.Aux1);
         if NColumnas~=4
            DatosN_Grid.Aux= DatosN_Grid.Aux';
         end

         % Guarda el valor de K para la variable
         Coef=polyfit(DatosN_Grid.Vector_Abcisas(Indice_MinK:Indice_MaxK),...
                      DatosN_Grid.Aux1(Indice_MinK:Indice_MaxK,1)',1);
```

```
         DatosN_Grid.K.Aux1=Coef(1);

      catch

          DatosN_Grid.K.Aux1='-';
      end

   end


   % Si hay que procesar Aux2
   if Variables_Calculadas.Aux2.BoolActivar

      try

         % Genera el string para el campo calculado
         Aux2_Str='';
         for i=1:length(Variables_Calculadas.Aux2.Coef)

            if i~=1
               Aux2_Str=strcat(Aux2_Str,'+');
            end

            Aux2_Str=strcat(Aux2_Str,num2str(Variables_Calculadas.Aux2.Coef(i)),...
'*','DatosN_Grid.',Variables_Calculadas.Aux2.Nombre(i),'(:,:)');

         end


         % Guarda el valor de los datos de la variable
         DatosN_Grid.Aux2=eval(cell2mat(Aux2_Str));
         [NDatos,NColumnas]=size(DatosN_Grid.Aux2);
         if NColumnas~=4
            DatosN_Grid.Aux2= DatosN_Grid.Aux2';
         end

         % Guarda el valor de K para la variable
         Coef=polyfit(DatosN_Grid.Vector_Abcisas(Indice_MinK:Indice_MaxK),...
                      DatosN_Grid.Aux2(Indice_MinK:Indice_MaxK,1)',1);
         DatosN_Grid.K.Aux2=Coef(1);

      catch

          DatosN_Grid.K.Aux2='-';

      end

   end


   % Si hay que procesar Aux3
   if Variables_Calculadas.Aux3.BoolActivar

      try

         % Genera el string para el campo calculado
         Aux3_Str='';
         for i=1:length(Variables_Calculadas.Aux3.Coef)

            if i~=1
               Aux3_Str=strcat(Aux3_Str,'+');
            end

            Aux3_Str=strcat(Aux3_Str,num2str(Variables_Calculadas.Aux3.Coef(i)),...
'*','DatosN_Grid.',Variables_Calculadas.Aux3.Nombre(i),'(:,:)');

         end


         % Guarda el valor de los datos de la variable
         DatosN_Grid.Aux3=eval(cell2mat(Aux3_Str));
```

```
        [NDatos,NColumnas]=size(DatosN_Grid.Aux3);
        if NColumnas~=4
           DatosN_Grid.Aux3= DatosN_Grid.Aux3';
        end

        % Guarda el valor de K para la variable
        Coef=polyfit(DatosN_Grid.Vector_Abcisas(Indice_MinK:Indice_MaxK),...
                     DatosN_Grid.Aux3(Indice_MinK:Indice_MaxK,1)',1);
        DatosN_Grid.K.Aux3=Coef(1);


      catch

         DatosN_Grid.K.Aux3='-';

      end

   end

end
```

```
% ------------------------------------------------------------------------------------------
%           Programa para la eliminación de datos nulos y obtención de la matriz
%           de ganancias para el barrido de TH
%
%       VARIABLES DE ENTRADA
%
%          Ciclos_Grid:             Estructura con un campo por cada una de las variables
%                                   ajustadas al grid:
%                                   (m, P1, P2, Qe, W, nCarnot, COP,COPc, T1, T2, T3, T4,
%                                   DeltaTe, DeltaTc, DeltaTsh, DeltaTsc y TH).
%                                   Cada uno de ellos es, a su vez, otra estructura con
%                                   estos campos:
%
%               Valores:            Matriz con los datos ajustados al Grid. Consta de 4
%                                   columnas:
%                                       Col1: Valor medio
%                                       Col2: Valor mínimo
%                                       Col3: Valor máximo
%                                       Col4: Número de datos brutos ajustados al punto
%               Min:                Valor mínimo encontrado en los datos brutos
%               Max:                Valor máximo encontrado en los datos brutos
%               PtosSuperan_Porc    Índices con los puntos que han superado PorcLimite
%
%       Parametros_General:         Estructura con los parámetros de configuración generales
%                                   de los que se usan los siguientes:
%
%           Barrido_RPM:            Barrido a realizar en RPM
%               Min:                Valor mínimo
%               Paso_Porc           Paso en % sobre el rango efectivo de RPM
%               Paso_Abs            Paso en valor absoluto para RM
%               Max:                Valor máximo
%           Barrido_Av:             Barrido a realizar en Av
%               Min:                Valor mínimo
%               Paso_Porc:          Paso en % sobre el rango efectivo de Av
%               Paso_Abs            Paso en valor absoluto para Av
%               Max:                Valor máximo
%
%        Parametros_Grid:           Estructura con los parámetros de configuración
%                                   para el ajuste al GRID. Se utilizan los
%                                   siguientes campos:
%
%               Paso.RPM_Porc:      Paso en % sobre el rango de barrido para RPM
%               Paso.RPM_Abs:       Paso en valor absoluto para RPM. Utilizar uno u otro
%                                   y colocar el otro a cero.
%
%               Paso.Av_Porc:       Paso en % sobre el rango de barrido para Av
%               Paso.Av_Abs:        Paso en valor absoluto para Av. Utilizar uno u otro
%                                   y colocar el otro a cero.
%
%               Variables:          Estructura con dos campos por cada variable a tratar.
%                                   El primero indica si debe ajustarse al grid y el
%                                   segundo el porcentaje máximo de desviación antes
%                                   de incluirlo en los puntos a comprobar
%                   m,m_PorcentajeMax
%                   P1,P1_PorcentajeMax, P2,P2_PorcentajeMax
%                   Qe,Qe_PorcentajeMax, W,W_PorcentajeMax
%                   nCarnot.nCarnot_PorcentajeMax,COP,COP_PorcentajeMax
%                   COPc,COPc_PorcentajeMax
%                   T1,T1_PorcentajeMax,T2,T2_PorcentajeMax
%                   T3,T3_PorcentajeMax,T4,T4_PorcentajeMax
%                   DeltaTe,DeltaTe_PorcentajeMax,DeltaTc,DeltaTc_PorcentajeMax
%                   DeltaTsh,DeltaTsh_PorcentajeMax,DeltaTsc,DeltaTsc_PorcentajeMax
%                   TH,TH_PorcentajeMax
%
%       Parametros_Analisis:        Estructura con los siguientes campos:
%
%            PorcentajeK:           Porcentaje alrededor del punto de operación a utilizar
%                                   para el cálculo de las ganancias
%
%       Ciclos_Brutos:              Esttructura con los puntos de operación válidos.
%                                   Contiene los siguientes campos:
%
%               RPM:                Vector con los valores de RPM de los datos brutos
%               Av:                 Vector con los valores de apertura (%) de los datos brutos
%               PtosOP_Q:           Vector de tres componentes con los puntos de operación
%                                   aptos encontrados. Contiene tres columnas:
%                                       Col1: Índice para Temp_Test
%                                       Col2: Índice para RPM
%                                       Col3: Índice para Av
%                                       Col4: Caudal (kg/s)
%
%               P1:                 Vector de n filas con la presión (bares) en el evaporador
%                                   para cada una de las filas de Temp_Test
```

```
%               P2:                 Vector de n filas con la presión (bares) en el condensador
%                                   para cada una de las filas de Temp_Test
%               T1:                 Vector de n filas con la temperatura (°C) a la entrada
%                                   del compresor para cada una de las filas de Temp_Test
%               T2:                 Vector de n filas con la temperatura (°C) a la salida
%                                   del compresor para cada una de las filas de Temp_Test
%               T3:                 Vector de n filas con la temperatura (°C) a la entrada
%                                   de la válvula de expansión para cada una de las filas de
%                                   Temp_Test
%               T4:                 Vector de n filas con la temperatura (°C) a la salida de
%                                   la válvula de expansión para cada una de las filas
%                                   de Temp_Test
%               H14:                Vector de n filas con la diferencia de entalpía entre
%                                   los estados 1 y 4 en kJ/kg para cada una de las filas
%                                   de Temp_Test
%               H21:                Vector de n filas con la diferencia de entalpía entre
%                                   los estados 2 y 1 en kJ/kg para cada una de las filas
%                                   de Temp_Test
%               COPc:               Vector de nfilas con el COP de Carnot para cada una de
%                                   las filas de Temp_Test
%
%               Temp_Test:          Matriz con n filas conteniendo las temperaturas empleadas:
%                                      Col1: Delta_Te (°C)
%                                      Col2: Delta_Tc (°C)
%                                      COl3: Delta_Tsh (°C)
%                                      Col4: Delta_Tsc (°C)
%                                      COl5: TH (°C)
%               Tiempo_CPU
%
%       VARIABLES DE SALIDA
%
%
%       DatosN_Grid:                Estructura con los ciclos procesados con los
%                                   siguientes campos:
%
%           Vector_Abcisas:         Vector con los valores de abcisas correspondientes a
%                                   los datos
%           TiempoCPU:              Tiempo empleado en la obtención de los datos
%           K:                      Ganancia de cada una de las variables con respecto a RPM
%                                   o Av.
%
%           m,P1,P2,Qe,T1,T2,T3,    Cada una de ellas contiene cuatro columnas: (Med,Min,Max,N)
%           T4,DeltaTe,DeltaTc,
%           DeltaTsh,DeltaTsc
% ------------------------------------------------------------------------------------------

function [DatosN_Vector]=Procesar_Variables_Ciclos_TH (Ciclos_Grid,Parametros_General,...
                                                       Parametros_Grid,Parametros_Analisis,...
                                                       Ciclos_Brutos)

   % Obtiene los parámetros
   Vector_TH=Ciclos_Grid.Vector;
   Datos_Vector=Ciclos_Grid.Datos;
   PasoGrid=Parametros_Grid.Paso;
   Variables=Parametros_Grid.Variables;
   PorcentajeK=Parametros_Analisis.PorcentajeK;
   TH0=Parametros_General.PuntoTrabajo.TH0;
   TiempoCPU=Ciclos_Brutos.Tiempo_CPU;
   Variables_Calculadas=Parametros_Analisis.Variables_Calculadas;

   % Si el paso de TH para el grid se ha indicado en % sobre el rango
   if PasoGrid.TH_Porc>0 && PasoGrid.TH_Abs==0

      % Obtiene el paso en valor absoluto a utilizar en el barrido de RPM
      TH_Min=fix(min(Ciclos_Brutos.Temp_Test(:,5)));
      TH_Max=fix(max(Ciclos_Brutos.Temp_Test(:,5)));
      PasoGrid.TH=0.01*PasoGrid.TH_Porc*(TH_Max-TH_Min);

   else

      PasoGrid.TH=PasoGrid.TH_Abs;

   end

   % Obtiene los índices válidos para las variables para el caso de RPM=RPM0 y Av=Av0
   BoolEncontrado=false;

   if Variables.m
      Indices_Validos_TH=find(Datos_Vector.m.Valores(:,4));
      BoolEncontrado=true;
```

```
end

if Variables.Qe && ~BoolEncontrado
   Indices_Validos_TH=find(Datos_Vector.Qe.Valores(:,4));
   BoolEncontrado=true;
end

if Variables.W && ~BoolEncontrado
   Indices_Validos_TH=find(Datos_Vector.W.Valores(:,4));
   BoolEncontrado=true;
end

if Variables.P1 && ~BoolEncontrado
   Indices_Validos_TH=find(Datos_Vector.P1.Valores(:,4));
   BoolEncontrado=true;
end

if Variables.P2 && ~BoolEncontrado
   Indices_Validos_TH=find(Datos_Vector.P2.Valores(:,4));
   BoolEncontrado=true;
end

if Variables.nCarnot && ~BoolEncontrado
   Indices_Validos_TH=find(Datos_Vector.nCarnot.Valores(:,4));
   BoolEncontrado=true;
end

if Variables.COP && ~BoolEncontrado
   Indices_Validos_TH=find(Datos_Vector.COP.Valores(:,4));
   BoolEncontrado=true;
end

if Variables.COPc && ~BoolEncontrado
   Indices_Validos_TH=find(Datos_Vector.COPc.Valores(:,4));
   BoolEncontrado=true;
end

if Variables.T1 && ~BoolEncontrado
   Indices_Validos_TH=find(Datos_Vector.T1.Valores(:,4));
   BoolEncontrado=true;
end

if Variables.T2 && ~BoolEncontrado
   Indices_Validos_TH=find(Datos_Vector.T2.Valores(:,4));
   BoolEncontrado=true;
end

if Variables.T3 && ~BoolEncontrado
   Indices_Validos_TH=find(Datos_Vector.T3.Valores(:,4));
   BoolEncontrado=true;
end

if Variables.T4 && ~BoolEncontrado
   Indices_Validos_TH=find(Datos_Vector.T4.Valores(:,4));
   BoolEncontrado=true;
end

if Variables.DeltaTe && ~BoolEncontrado
   Indices_Validos_TH=find(Datos_Vector.DeltaTe.Valores(:,4));
   BoolEncontrado=true;
end

if Variables.DeltaTc && ~BoolEncontrado
   Indices_Validos_TH=find(Datos_Vector.DeltaTc.Valores(:,4));
   BoolEncontrado=true;
end

if Variables.DeltaTsh && ~BoolEncontrado
   Indices_Validos_TH=find(Datos_Vector.DeltaTsh.Valores(:,4));
   BoolEncontrado=true;
end

if Variables.DeltaTsc && ~BoolEncontrado
   Indices_Validos_TH=find(Datos_Vector.DeltaTsc.Valores(:,4));
   BoolEncontrado=true;
end

if BoolEncontrado

   % Guarda el vector de abcisas
   DatosN_Vector.Vector_Abcisas=Vector_TH(Indices_Validos_TH);

   % Guarda el tiempo en segundos empleado en la obtención de los puntos
```

```
    DatosN_Vector.TiempoCPU=TiempoCPU/3600;

    % Obtiene los valores de apertura mínimo y máximo para el cálculo de la matriz de
    % ganancias
    TH_MinK=(1-0.01*PorcentajeK)*TH0;
    if TH_MinK<Vector_TH(1)
        TH_MinK=Vector_TH(1);
    end

    TH_MaxK=(1+0.01*PorcentajeK)*TH0;
    if TH_MaxK>Vector_TH(length(Vector_TH))
        TH_MaxK=Vector_TH(length(Vector_TH));
    end

    % Obtiene el índice de TH0 y lo acota
    [Indice_TH0,Aux]=Grid_Obt_Indices(TH0,1,Vector_TH(1),1,PasoGrid.TH,1);

    if Indice_TH0>length(Vector_TH)
       Indice_TH0=length(Vector_TH);
    end

    % Obtiene los indices a utilizar para el cálculo de la matriz de
    % ganancias (+-PorcentajeK sobre el punto nominal de operación de
    % THv)
    Indice_MinK=find(DatosN_Vector.Vector_Abcisas>=TH_MinK,1,'first');
    Indice_MaxK=find(DatosN_Vector.Vector_Abcisas<=TH_MaxK,1,'last');

    if Indice_MinK>1
        if abs(DatosN_Vector.Vector_Abcisas(Indice_MinK-1)-TH_MinK)<...
           abs(DatosN_Vector.Vector_Abcisas(Indice_MinK)-TH_MinK)
           Indice_MinK=Indice_MinK-1;
        end
    end

    if Indice_MaxK<length(DatosN_Vector.Vector_Abcisas)
        if abs(DatosN_Vector.Vector_Abcisas(Indice_MaxK+1)-TH_MaxK)<...
           abs(DatosN_Vector.Vector_Abcisas(Indice_MaxK)-TH_MaxK)
           Indice_MaxK=Indice_MaxK+1;
        end
    end

end


% Si hay que procesar m
if Variables.m

    % Guarda el valor de los datos de la variable
    DatosN_Vector.m(:,:)=Datos_Vector.m.Valores(Indices_Validos_TH,:);
    [NDatos,NColumnas]=size(DatosN_Vector.m);
    if NColumnas~=4
         DatosN_Vector.m= DatosN_Vector.m';
    end


    % Guarda el valor de K para la variable
    Coef=polyfit(DatosN_Vector.Vector_Abcisas(Indice_MinK:Indice_MaxK),...
                 DatosN_Vector.m(Indice_MinK:Indice_MaxK,1)',1);
    DatosN_Vector.K.m=Coef(1);

end


% Si hay que procesar P1
if Variables.P1

    % Guarda el valor de los datos de la variable
    DatosN_Vector.P1(:,:)=Datos_Vector.P1.Valores(Indices_Validos_TH,:);
    [NDatos,NColumnas]=size(DatosN_Vector.P1);
    if NColumnas~=4
         DatosN_Vector.P1= DatosN_Vector.P1';
    end

    % Guarda el valor de K para la variable
    Coef=polyfit(DatosN_Vector.Vector_Abcisas(Indice_MinK:Indice_MaxK),...
                 DatosN_Vector.P1(Indice_MinK:Indice_MaxK,1)',1);
    DatosN_Vector.K.P1=Coef(1);

end
```

```
% Si hay que procesar P2
if Variables.P2

    % Guarda el valor de los datos de la variable
    DatosN_Vector.P2(:,:)=Datos_Vector.P2.Valores(Indices_Validos_TH,:);
    [NDatos,NColumnas]=size(DatosN_Vector.P2);
    if NColumnas~=4
         DatosN_Vector.P2= DatosN_Vector.P2';
    end

    % Guarda el valor de K para la variable
    Coef=polyfit(DatosN_Vector.Vector_Abcisas(Indice_MinK:Indice_MaxK),...
                 DatosN_Vector.P2(Indice_MinK:Indice_MaxK,1)',1);
    DatosN_Vector.K.P2=Coef(1);

end


% Si hay que procesar Qe
if Variables.Qe

    % Guarda el valor de los datos de la variable
    DatosN_Vector.Qe(:,:)=Datos_Vector.Qe.Valores(Indices_Validos_TH,:);
    [NDatos,NColumnas]=size(DatosN_Vector.Qe);
    if NColumnas~=4
         DatosN_Vector.Qe= DatosN_Vector.Qe';
    end

    % Guarda el valor de K para la variable
    Coef=polyfit(DatosN_Vector.Vector_Abcisas(Indice_MinK:Indice_MaxK),...
                 DatosN_Vector.Qe(Indice_MinK:Indice_MaxK,1)',1);
    DatosN_Vector.K.Qe=Coef(1);

end


% Si hay que procesar W
if Variables.W

    % Guarda el valor de los datos de la variable
    DatosN_Vector.W(:,:)=Datos_Vector.W.Valores(Indices_Validos_TH,:);
    [NDatos,NColumnas]=size(DatosN_Vector.W);
    if NColumnas~=4
         DatosN_Vector.W= DatosN_Vector.W';
    end

    % Guarda el valor de K para la variable
    Coef=polyfit(DatosN_Vector.Vector_Abcisas(Indice_MinK:Indice_MaxK),...
                 DatosN_Vector.W(Indice_MinK:Indice_MaxK,1)',1);
    DatosN_Vector.K.W=Coef(1);

end


% Si hay que procesar nCarnot
if Variables.nCarnot

    % Guarda el valor de los datos de la variable
    DatosN_Vector.nCarnot(:,:)=Datos_Vector.nCarnot.Valores(Indices_Validos_TH,:);
    [NDatos,NColumnas]=size(DatosN_Vector.nCarnot);
    if NColumnas~=4
         DatosN_Vector.nCarnot= DatosN_Vector.nCarnot';
    end

    % Guarda el valor de K para la variable
    Coef=polyfit(DatosN_Vector.Vector_Abcisas(Indice_MinK:Indice_MaxK),...
                 DatosN_Vector.nCarnot(Indice_MinK:Indice_MaxK,1)',1);
    DatosN_Vector.K.nCarnot=Coef(1);

end


% Si hay que procesar COP
if Variables.COP

    % Guarda el valor de los datos de la variable
    DatosN_Vector.COP(:,:)=Datos_Vector.COP.Valores(Indices_Validos_TH,:);
    [NDatos,NColumnas]=size(DatosN_Vector.COP);
    if NColumnas~=4
         DatosN_Vector.COP= DatosN_Vector.COP';
    end
```

```
       % Guarda el valor de K para la variable
    Coef=polyfit(DatosN_Vector.Vector_Abcisas(Indice_MinK:Indice_MaxK),...
                 DatosN_Vector.COP(Indice_MinK:Indice_MaxK,1)',1);
    DatosN_Vector.K.COP=Coef(1);

end


% Si hay que procesar W
if Variables.COPc
    % Guarda el valor de los datos de la variable
    DatosN_Vector.COPc(:,:)=Datos_Vector.COPc.Valores(Indices_Validos_TH,:);
    [NDatos,NColumnas]=size(DatosN_Vector.COPc);
    if NColumnas~=4
         DatosN_Vector.COPc= DatosN_Vector.COPc';
    end

    % Guarda el valor de K para la variable
    Coef=polyfit(DatosN_Vector.Vector_Abcisas(Indice_MinK:Indice_MaxK),...
                 DatosN_Vector.COPc(Indice_MinK:Indice_MaxK,1)',1);
    DatosN_Vector.K.COPc=Coef(1);

end


% Si hay que procesar T1
if Variables.T1

    % Guarda el valor de los datos de la variable
    DatosN_Vector.T1(:,:)=Datos_Vector.T1.Valores(Indices_Validos_TH,:);
    [NDatos,NColumnas]=size(DatosN_Vector.T1);
    if NColumnas~=4
         DatosN_Vector.T1= DatosN_Vector.T1';
    end

    % Guarda el valor de K para la variable
    Coef=polyfit(DatosN_Vector.Vector_Abcisas(Indice_MinK:Indice_MaxK),...
                 DatosN_Vector.T1(Indice_MinK:Indice_MaxK,1)',1);
    DatosN_Vector.K.T1=Coef(1);

end


% Si hay que procesar T2
if Variables.T2

    % Guarda el valor de los datos de la variable
    DatosN_Vector.T2(:,:)=Datos_Vector.T2.Valores(Indices_Validos_TH,:);
    [NDatos,NColumnas]=size(DatosN_Vector.T2);
    if NColumnas~=4
         DatosN_Vector.T2= DatosN_Vector.T2';
    end

    % Guarda el valor de K para la variable
    Coef=polyfit(DatosN_Vector.Vector_Abcisas(Indice_MinK:Indice_MaxK),...
                 DatosN_Vector.T2(Indice_MinK:Indice_MaxK,1)',1);
    DatosN_Vector.K.T2=Coef(1);

end


% Si hay que procesar T3
if Variables.T3

    % Guarda el valor de los datos de la variable
    DatosN_Vector.T3(:,:)=Datos_Vector.T3.Valores(Indices_Validos_TH,:);
    [NDatos,NColumnas]=size(DatosN_Vector.T3);
    if NColumnas~=4
         DatosN_Vector.T3= DatosN_Vector.T3';
    end

    % Guarda el valor de K para la variable
    Coef=polyfit(DatosN_Vector.Vector_Abcisas(Indice_MinK:Indice_MaxK),...
                 DatosN_Vector.T3(Indice_MinK:Indice_MaxK,1)',1);
    DatosN_Vector.K.T3=Coef(1);

end


% Si hay que procesar T4
```

```
if Variables.T4

    % Guarda el valor de los datos de la variable
    DatosN_Vector.T4(:,:)=Datos_Vector.T4.Valores(Indices_Validos_TH,:);
    [NDatos,NColumnas]=size(DatosN_Vector.T4);
    if NColumnas~=4
         DatosN_Vector.T4= DatosN_Vector.T4';
    end

    % Guarda el valor de K para la variable
    Coef=polyfit(DatosN_Vector.Vector_Abcisas(Indice_MinK:Indice_MaxK),...
                 DatosN_Vector.T4(Indice_MinK:Indice_MaxK,1)',1);
    DatosN_Vector.K.T4=Coef(1);

end


% Si hay que procesar DeltaTe
if Variables.DeltaTe

    % Guarda el valor de los datos de la variable
    DatosN_Vector.DeltaTe(:,:)=Datos_Vector.DeltaTe.Valores(Indices_Validos_TH,:);
    [NDatos,NColumnas]=size(DatosN_Vector.DeltaTe);
    if NColumnas~=4
         DatosN_Vector.DeltaTe= DatosN_Vector.DeltaTe';
    end

    % Guarda el valor de K para la variable
    Coef=polyfit(DatosN_Vector.Vector_Abcisas(Indice_MinK:Indice_MaxK),...
                 DatosN_Vector.DeltaTe(Indice_MinK:Indice_MaxK,1)',1);
    DatosN_Vector.K.DeltaTe=Coef(1);

end


% Si hay que procesar DeltaTc
if Variables.DeltaTc

    % Guarda el valor de los datos de la variable
    DatosN_Vector.DeltaTc(:,:)=Datos_Vector.DeltaTc.Valores(Indices_Validos_TH,:);
    [NDatos,NColumnas]=size(DatosN_Vector.DeltaTc);
    if NColumnas~=4
         DatosN_Vector.DeltaTc= DatosN_Vector.DeltaTc';
    end

    % Guarda el valor de K para la variable
    Coef=polyfit(DatosN_Vector.Vector_Abcisas(Indice_MinK:Indice_MaxK),...
                 DatosN_Vector.DeltaTc(Indice_MinK:Indice_MaxK,1)',1);
    DatosN_Vector.K.DeltaTc=Coef(1);

end


% Si hay que procesar DeltaTsh
if Variables.DeltaTsh

    % Guarda el valor de los datos de la variable
    DatosN_Vector.DeltaTsh(:,:)=Datos_Vector.DeltaTsh.Valores(Indices_Validos_TH,:);
    [NDatos,NColumnas]=size(DatosN_Vector.DeltaTsh);
    if NColumnas~=4
         DatosN_Vector.DeltaTsh= DatosN_Vector.DeltaTsh';
    end

    % Guarda el valor de K para la variable
    Coef=polyfit(DatosN_Vector.Vector_Abcisas(Indice_MinK:Indice_MaxK),...
                 DatosN_Vector.DeltaTsh(Indice_MinK:Indice_MaxK,1)',1);
    DatosN_Vector.K.DeltaTsh=Coef(1);

end


% Si hay que procesar DeltaTsc
if Variables.DeltaTsc

    % Guarda el valor de los datos de la variable
    DatosN_Vector.DeltaTsc(:,:)=Datos_Vector.DeltaTsc.Valores(Indices_Validos_TH,:);
    [NDatos,NColumnas]=size(DatosN_Vector.DeltaTsc);
    if NColumnas~=4
         DatosN_Vector.DeltaTsc= DatosN_Vector.DeltaTsc';
    end

    % Guarda el valor de K para la variable
    Coef=polyfit(DatosN_Vector.Vector_Abcisas(Indice_MinK:Indice_MaxK),...
```

```
                DatosN_Vector.DeltaTsc(Indice_MinK:Indice_MaxK,1)',1);
        DatosN_Vector.K.DeltaTsc=Coef(1);

    end

end
```

```
% ------------------------------------------------------------------------------------------
%                Función para la selección de los puntos de trabajo a estudiar
%
%       VARIABLES DE ENTRADA
%
%       Ciclos:                     Estructura con los puntos de operación válidos.
%                                   Contiene los siguientes campos:
%
%               RPM:                Vector con los valores de RPM de los datos brutos
%               Av:                 Vector con los valores de apertura (%) de los datos brutos

%               PtosOP_Q:           Vector de tres componentes con los puntos de operación
%                                   aptos encontrados. Contiene tres columnas:
%                                       Col1: Índice para Temp_Test
%                                       Col2: Índice para RPM
%                                       Col3: Índice para Av
%                                       Col4: Caudal (kg/s)
%
%               P1:                 Vector de n filas con la presión (bares) en el evaporador
%                                   para cada una de las filas de Temp_Test
%               P2:                 Vector de n filas con la presión (bares) en el condensador
%                                   para cada una de las filas de Temp_Test
%               T1:                 Vector de n filas con la temperatura (ºC) a la entrada
%                                   del compresor para cada una de las filas de Temp_Test
%               T2:                 Vector de n filas con la temperatura (ºC) a la salida
%                                   del compresor para cada una de las filas de Temp_Test
%               T3:                 Vector de n filas con la temperatura (ºC) a la entrada
%                                   de la válvula de expansión para cada una de las filas de
%                                   Temp_Test
%               T4:                 Vector de n filas con la temperatura (ºC) a la salida de
%                                   la válvula de expansión para cada una de las filas
%                                   de Temp_Test
%               H14:                Vector de n filas con la diferencia de entalpía entre
%                                   los estados 1 y 4 en kJ/kg para cada una de las filas
%                                   de Temp_Test
%               H21:                Vector de n filas con la diferencia de entalpía entre
%                                   los estados 2 y 1 en kJ/kg para cada una de las filas
%                                   de Temp_Test
%               COPc:               Vector de nfilas con el COP de Carnot para cada una de
%                                   las filas de Temp_Test
%
%               Temp_Test:          Matriz con n filas conteniendo las temperaturas empleadas:
%                                      Col1: Delta_Te (ºC)
%                                      Col2: Delta_Tc (ºC)
%                                      Col3: Delta_Tsh (ºC)
%                                      Col4: Delta_Tsc (ºC)
%                                      Col5: TH (ºC)
%
%       Datos_Grid:                 Estructura con un campo por cada una de las variables
%                                   ajustadas al grid:
%                                   (m, P1, P2, Qe, W, nCarnot, COP,COPc, T1, T2, T3, T4,
%                                   DeltaTe, DeltaTc, DeltaTsh, DeltaTsc y TH).
%                                   Cada uno de ellos es, a su vez, otra estructura con
%                                   estos campos:
%
%               Valores:            Matriz con los datos ajustados al Grid. Consta de 4
%                                   columnas:
%                                       Col1: Valor medio
%                                       Col2: Valor mínimo
%                                       Col3: Valor máximo
%                                       Col4: Número de datos brutos ajustados al punto
%               Min:                Valor mínimo encontrado en los datos brutos
%               Max:                Valor máximo encontrado en los datos brutos
%               PtosSuperan_Porc    Índices con los puntos que han superado PorcLimite
%
%       Vector_RPM:                 Vector con los valores de RPM del grid
%       Vector_Av:                  Vector con los valores de Av del grid
%
%       Graficas:                   Estructura con dos campos Qe y nCarnot.Cada uno de ellos
%                                   tiene los siguientes campos:
%
%               Surf:               Indica que hay que mostrar la gráfica SURF
%               Contour:            Indica que hay que mostrar la gráfica CONTOUR
%               PtoOp:              Indica cual de los puntos de operación
%                                   calculados hay que mostrar sobre el dibujo
%               Rango_RPM:          Vector conteniendo el rango de valores
%                                   para RPM. Si es [0 0] se ajusta automáticamente.
%               Rango_Av:           Vector conteniendo el rango de valores
%                                   para Av. Si es [0 0] se ajusta automáticamente.
%               RangoVariable:      Vector conteniendo el rango de la
%                                   variable a mostrar.
```

```
%               AzEl:               Vector con dos componentes indicando la
%                                   orientación y azimut del dibujo
%               Anotacion.YMin:     Contiene la posición de Y normalizada donde colocar
%                                   la primera etiqueta
%               Anotacion.YDelta:   Contiene el paso entre etiquetas
%
%       PtosEstudio:                Estructura con los siguientes campos para cada uno
%                                   de los puntos a estudiar:
%               Porcentaje:         Porcentaje con respecto a Qemax
%               Variacion:          Variación (+-%) sobre el porcentaje
%               NPuntosMax:         Número de puntos a considerar que cumplan la condición
%
%       BoolAscendente:             Indica si se deben considerar los NPuntosMax de
%                                   mayor a menor Qe
%
% ------------------------------------------------------------------------------------------

function Sel_PtosOP_Ciclos (Ciclos,Datos_Grid,Vector_RPM,Vector_Av,Graficas,...
                            PtosEstudio,Bool_Ascendente)


   % Fija el modo de ordenación
   if Bool_Ascendente
      Modo='ascend';
   else
      Modo='descend';
   end

   % Inicializa los vectores que contendrán las puntos de operación a estudiar
   Indice_RPM_Estudio=zeros(length(PtosEstudio),1);
   Indice_Av_Estudio=zeros(length(PtosEstudio),1);

   % Obtiene el rango obtenido para RPM, Av y m
   RPM_Min=fix(min(Ciclos.RPM(Ciclos.PtosOp_Q(:,2))));
   RPM_Max=fix(max(Ciclos.RPM(Ciclos.PtosOp_Q(:,2)))+1);
   Av_Min=fix(min(Ciclos.Av(Ciclos.PtosOp_Q(:,3))));
   Av_Max=fix(max(Ciclos.Av(Ciclos.PtosOp_Q(:,3)))+1);

   % Fija los límites para Qe
   if Graficas.Qe.Rango_RPM(1)==0 && Graficas.Qe.Rango_RPM(2)==0
      Graficas.Qe.RPMg=[RPM_Min RPM_Max];
   else
      Graficas.Qe.RPMg=Graficas.Qe.Rango_RPM;
   end

   if Graficas.Qe.Rango_Av(1)==0 && Graficas.Qe.Rango_Av(2)==0
      Graficas.Qe.Avg=[Av_Min Av_Max];
   else
      Graficas.Qe.Avg=Graficas.Qe.Rango_Av;
   end

   % Fija los límites para We
   if Graficas.We.Rango_RPM(1)==0 && Graficas.We.Rango_RPM(2)==0
      Graficas.We.RPMg=[RPM_Min RPM_Max];
   else
      Graficas.We.RPMg=Graficas.We.Rango_RPM;
   end

   if Graficas.We.Rango_Av(1)==0 && Graficas.We.Rango_Av(2)==0
      Graficas.We.Avg=[Av_Min Av_Max];
   else
      Graficas.We.Avg=Graficas.We.Rango_Av;
   end


   % Fija los límites para nCarnot
   if Graficas.nCarnot.Rango_RPM(1)==0 && Graficas.nCarnot.Rango_RPM(2)==0
      Graficas.nCarnot.RPMg=[RPM_Min RPM_Max];
   else
      Graficas.nCarnot.RPMg=Graficas.nCarnot.Rango_RPM;
   end

   if Graficas.nCarnot.Rango_Av(1)==0 && Graficas.nCarnot.Rango_Av(2)==0
      Graficas.nCarnot.Avg=[Av_Min Av_Max];
   else
      Graficas.nCarnot.Avg=Graficas.nCarnot.Rango_Av;
   end


   % Genera el grid para las gráficas
```

```
[Grid_RPM,Grid_Av]=meshgrid(Vector_RPM,Vector_Av);

% Muestra el rango a analizar
fprintf ('RANGO DE ESTUDIO:\n');
fprintf('RPM (min):%5d\t\tRPM(max): %5d \tAv(min):%3d \tAv(max):%3d\n',...
        RPM_Min,RPM_Max,Av_Min,Av_Max);

% Muestra los posibles puntos de operación con los parámetros indicados
for i=1:length(PtosEstudio)

   % Obtiene los índices de los puntos de operación que cumplen los criterios establecidos
   [Indice_RPM0,Indice_Av0]=Sel_PtosOP_Qe(Datos_Grid.Qe.Valores(:,:,1),...
                                          Datos_Grid.nCarnot.Valores(:,:,1),...
                                          PtosEstudio(i),Modo);

   % Muestra por pantalla los puntos de operación que cumplen los criterios establecidos
   Sel_PtosOP_Qe_Mostrar_Texto(Vector_RPM,Vector_Av,Indice_RPM0,Indice_Av0,...
                               Datos_Grid.Qe.Valores(:,:,1),...
                               Datos_Grid.nCarnot.Valores(:,:,1),PtosEstudio(i));

   if ~isempty(Indice_RPM0)
      Indice_RPM_Estudio(i)=Indice_RPM0(1);
   else
      Indice_RPM_Estudio(i)=0;
   end

   if ~isempty(Indice_Av0)
      Indice_Av_Estudio(i)=Indice_Av0(1);
   else
      Indice_Av_Estudio(i)=0;
   end

end


% Si hay que mostrar SURF de Qe
if Graficas.Qe.Surf

   % Elimina los valores no deseados
   [RPM_Aux Av_Aux]=find (Datos_Grid.Qe.Valores(:,:,1)<Graficas.Qe.RangoVariable(1) | ...
                          Datos_Grid.Qe.Valores(:,:,1)>Graficas.Qe.RangoVariable(2));
   for i=1:length(RPM_Aux)
      Datos_Grid.Qe.Valores(RPM_Aux(i),Av_Aux(i),:)=0;
   end

   % Muestra (SURF) la capacidad frigorífica Qe (W)
   Figura_Qe1 = figure('Color',[1 1 1]);
   Plot_Qe1 = subplot(1,1,1,'Parent',Figura_Qe1,'CLim',...
                      [0,max(max(Datos_Grid.Qe.Valores(:,:,1)))]);
   title('Capacidad frigorifica Qe (W)','FontWeight','bold','FontSize',12,'FontName','Arial');
   xlabel('N(rpm)');ylabel('Av(%)');zlabel('Qe(W)');
   colorbar('peer',Plot_Qe1);
   view(Plot_Qe1,Graficas.Qe.AzEl);
   grid(Plot_Qe1,'on');
   hold(Plot_Qe1,'all');
   surf(Grid_RPM,Grid_Av,Datos_Grid.Qe.Valores(:,:,1)','LineStyle','none',...
        'FaceColor','interp');
   xlim(Graficas.Qe.RPMg);ylim(Graficas.Qe.Avg);zlim(Graficas.Qe.RangoVariable);

   % Si hay que mostrar los puntos de operación a seleccionar
   if Graficas.Qe.PtoOp>0

      [Indice_RPM0,Indice_Av0]=Sel_PtosOP_Qe(Datos_Grid.Qe.Valores(:,:,1),...
                                             Datos_Grid.nCarnot.Valores(:,:,1),...
                                             PtosEstudio(Graficas.Qe.PtoOp),Modo);


      Sel_PtosOP_Qe_Mostrar_Plot(Vector_RPM,Vector_Av,Indice_RPM0,Indice_Av0,...
                                 Datos_Grid.Qe.Valores(:,:,1),...
                                 Datos_Grid.nCarnot.Valores(:,:,1),true,...
                                 Graficas.Qe.Anotacion.YMin,Graficas.Qe.Anotacion.YDelta);
   end

end


% Si hay que mostrar CONTOUR de Qe
if Graficas.Qe.Contour

   % Muestra (CONTOUR) la capacidad frigorífica Qe(W)
   Figura_Qe2 = figure('Color',[1 1 1]);
   Plot_Qe2 = subplot(1,1,1,'Parent',Figura_Qe2,'Layer','top');
   title('Capacidad frigorifica Qe (W)','FontWeight','bold','FontSize',12,'FontName','Arial');
```

```
    xlabel('N(rpm)');ylabel('Av(%)');zlabel('Qe(W)');
    colorbar('peer',Plot_Qe2);
    grid(Plot_Qe2,'on');
    hold(Plot_Qe2,'all');
    contour(Grid_RPM,Grid_Av,Datos_Grid.Qe.Valores(:,:,1)','LineStyle','none','LineColor',...
            [0 0 0],'Fill','on','Parent',Plot_Qe2);
    xlim(Graficas.Qe.RPMg);ylim(Graficas.Qe.Avg);zlim(Graficas.Qe.RangoVariable);

end



% Si hay que mostrar SURF de We
 if Graficas.We.Surf

   % Elimina los valores no deseados
    [RPM_Aux Av_Aux]=find (Datos_Grid.W.Valores(:,:,1)<Graficas.We.RangoVariable(1) | ...
                           Datos_Grid.W.Valores(:,:,1)>Graficas.We.RangoVariable(2));
    for i=1:length(RPM_Aux)
       Datos_Grid.W.Valores(RPM_Aux(i),Av_Aux(i),:)=0;
    end

    % Muestra (SURF) la potencia de compresión We (W)
    Figura_We1 = figure('Color',[1 1 1]);
    Plot_We1 = subplot(1,1,1,'Parent',Figura_We1,'CLim',...
                       [0,max(max(Datos_Grid.W.Valores(:,:,1)))]);
    title('Potencia de compresión W (W)','FontWeight','bold','FontSize',12,'FontName','Arial');
    xlabel('N(rpm)');ylabel('Av(%)');zlabel('W (W)');
    colorbar('peer',Plot_We1);
    view(Plot_We1,Graficas.Qe.AzEl);
    grid(Plot_We1,'on');
    hold(Plot_We1,'all');
    surf(Grid_RPM,Grid_Av,Datos_Grid.W.Valores(:,:,1)','LineStyle','none',...
         'FaceColor','interp');
    xlim(Graficas.We.RPMg);ylim(Graficas.We.Avg);zlim(Graficas.We.RangoVariable);

end


 % Si hay que mostrar CONTOUR de We
 if Graficas.We.Contour

    % Muestra (CONTOUR) la potencia de compresión We(W)
    Figura_We2 = figure('Color',[1 1 1]);
    Plot_We2 = subplot(1,1,1,'Parent',Figura_We2,'Layer','top');
    title('Potencia de compresión W (W)','FontWeight','bold','FontSize',12,'FontName','Arial');
    xlabel('N(rpm)');ylabel('Av(%)');zlabel('We(W)');
    colorbar('peer',Plot_We2);
    grid(Plot_We2,'on');
    hold(Plot_We2,'all');
    contour(Grid_RPM,Grid_Av,Datos_Grid.W.Valores(:,:,1)','LineStyle','none','LineColor',...
            [0 0 0],'Fill','on','Parent',Plot_We2);
    xlim(Graficas.We.RPMg);ylim(Graficas.We.Avg);zlim(Graficas.We.RangoVariable);

 end


% Si hay que mostrar SURF de nCarnot
if Graficas.nCarnot.Surf

    % Elimina los valores no deseados
   [RPM_Aux Av_Aux]=find (Datos_Grid.nCarnot.Valores(:,:,1)<Graficas.nCarnot.RangoVariable(1) | ...
                          Datos_Grid.nCarnot.Valores(:,:,1)>Graficas.nCarnot.RangoVariable(2));
   for i=1:length(RPM_Aux)
      Datos_Grid.nCarnot.Valores(RPM_Aux(i),Av_Aux(i),:)=0;
   end

   % Muestra (SURF) el rendimiento de Carnot (nCarnot)
   Figura_nCarnot1 = figure('Color',[1 1 1]);
   Plot_nCarnot1 = subplot(1,1,1,'Parent',Figura_nCarnot1,'CLim',...
                           [0.8,max(max(Datos_Grid.nCarnot.Valores(:,:,1)))]);
   title('Rendimiento de CARNOT (n)','FontWeight','bold','FontSize',12,'FontName','Arial');
   xlabel('N(rpm)');ylabel('Av(%)');zlabel('nCarnot');
   colorbar('peer',Plot_nCarnot1);
   view(Plot_nCarnot1,Graficas.nCarnot.AzEl);
   grid(Plot_nCarnot1,'on');
   hold(Plot_nCarnot1,'all');
   surf(Grid_RPM,Grid_Av,Datos_Grid.nCarnot.Valores(:,:,1)',...
        'LineStyle','none','FaceColor','interp');
   xlim(Graficas.nCarnot.RPMg);ylim(Graficas.nCarnot.Avg);
```

```
      zlim(Graficas.nCarnot.RangoVariable);

      % Si hay que mostrar los puntos de operación a seleccionar
      if Graficas.nCarnot.PtoOp>0

         [Indice_RPM0,Indice_Av0]=Sel_PtosOP_Qe(Datos_Grid.Qe.Valores(:,:,1),...
                                                Datos_Grid.nCarnot.Valores(:,:,1),...
                                                PtosEstudio(Graficas.nCarnot.PtoOp),Modo);

         Sel_PtosOP_Qe_Mostrar_Plot(Vector_RPM,Vector_Av,Indice_RPM0,Indice_Av0,...
                                    Datos_Grid.Qe.Valores,Datos_Grid.nCarnot.Valores(:,:,1),...
                                    false,Graficas.nCarnot.Anotacion.YMin,...
                                    Graficas.nCarnot.Anotacion.YDelta);

      end

   end

   % Si hay que mostrar CONTOUR de nCarnot
   if Graficas.nCarnot.Contour

      % Muestra (CONTOUR) el rendimiento de Carnot (nCarnot)
      Figura_nCarnot2 = figure('Color',[1 1 1]);
      Plot_nCarnot2 = subplot(1,1,1,'Parent',Figura_nCarnot2,'Layer','top');
      title('Rendimiento de CARNOT (n)','FontWeight','bold','FontSize',12,'FontName','Arial');
      xlabel('N(rpm)');ylabel('Av(%)');zlabel('nCarnot');
      colorbar('peer',Plot_nCarnot2);
      grid(Plot_nCarnot2,'on');
      hold(Plot_nCarnot2,'all');
      contour(Grid_RPM,Grid_Av,Datos_Grid.nCarnot.Valores(:,:,1)','LineStyle','none',...
              'LineColor',[0 0 0],'Fill','on','Parent',Plot_nCarnot2);
      xlim(Graficas.nCarnot.RPMg);ylim(Graficas.nCarnot.Avg);
      zlim(Graficas.nCarnot.RangoVariable);

   end

end
```

```
% ------------------------------------------------------------------------------------------
%                    Función para seleccionar los puntos de operación que
%                             cumplen con los criterios indicados
%
%       VARIABLES DE ENTRADA
%
%       Qe:                         Matriz con los valores medios de Qe
%       nCarnot:                    Matriz con los valores medios de nCarnot
%
%       P0:                         Estructura con los siguientes campos:
%               Porcentaje:         Porcentaje de valores de Qemax a mostrar
%                                   (Porcentaje +- Variacion)
%               Variación:          Variación del porcentaje. (Porcentaje +- Variacion)
%       Modo:                       Ascend o Descend
%
%       VARIABLES DE SALIDA
%
%       Indice_RPM0:                Vector con los índices ordenados de los
%                                   puntos seleccionados (RPM)
%       Indice_Av0:                 Vector con los índices ordenados de los
%                                   puntos seleccionados (Av)
% ------------------------------------------------------------------------------------------

function [Indice_RPM0,Indice_Av0]=Sel_PtosOP_Qe(Qe, nCarnot,P0,Modo)


   % Obtiene el valor máximo de Qe
   Qe_Max=max(max(Qe));

   % Obtiene los valores de los índices que cumplen el criterio establecido
   [Indice_RPM,Indice_Av]=find(Qe>=0.01*(P0.Porcentaje-P0.Variacion)*Qe_Max & ...
                               Qe<=0.01*(P0.Porcentaje+P0.Variacion)*Qe_Max);

   [Indice_RPM0,Indice_Av0]=Grid_Ordenar_Indices(Indice_RPM,Indice_Av,nCarnot,Modo);

   if P0.NPuntosMax>0

      Indice_RPM0(P0.NPuntosMax+1:length(Indice_RPM0))=[];
      Indice_Av0(P0.NPuntosMax+1:length(Indice_Av0))=[];

   end

end
```

```
% ------------------------------------------------------------------------------------------
%              Función para mostrar los puntos de trabajo seleccionados en gráfico
%
%       VARIABLES DE ENTRADA
%
%
%       Vector_RPM:                 Vector conteniendo los valores a utilizar en el grid RPM.
%       Vector_Av:                  Vector conteniendo los valores a utilizar en el grid Av.
%       Indice_RPM:                 Vector conteniendo los índices de RPM a mostrar.
%       Indice_Av:                  Vector conteniendo los índices de Av a mostrar.
%       Qe:                         Matriz con los valores medios de Qe
%       nCarnot:                    Matriz con los valores medios de nCarnot
%       BoolQe:                     Indica si el punto debe estar referenciado a Qe o a
%                                   nCarnot en el gráfico.
%       yPosMin:                    Valor mínimo en coordenadas normalizadas en donde mostrar
%                                   el texto de las anotaciones.
%       yPosDelta:                  Incremento en coordenadas normalizadas para la posición
%                                   Y de cada una de las anotaciones.
%
% ------------------------------------------------------------------------------------------


function Sel_PtosOP_Qe_Mostrar_Plot(Vector_RPM,Vector_Av,Indice_RPM,Indice_Av,...
                                    Qe,nCarnot,BoolQe,yPosMin,yPosDelta)

   % Recorre cada uno de los puntos a mostrar
   for i=1:length(Indice_RPM)

      RPM0= Vector_RPM(Indice_RPM(i));
      Av0=Vector_Av(Indice_Av(i));
      Qe0=Qe(Indice_RPM(i),Indice_Av(i));
      nCarnot0=nCarnot(Indice_RPM(i),Indice_Av(i));

      % Obtiene las coordenadas normalizadas en donde insertar la anotación
      if BoolQe
         [xPos, yPos] = ds2fig(RPM0,Av0,Qe0);
      else
         [xPos, yPos] = ds2fig(RPM0,Av0,nCarnot0);
      end

      % Prepara el texto de la anotación
      Texto=sprintf(' (%0.1f,%0.1f) \nQ%7.2f n%6.4f ',RPM0,Av0,Qe0,nCarnot0);

      % Inserta la anotación
      yPosAux=yPosMin+(i-1)*yPosDelta;
      if yPosAux>1
         yPosAux=1;
      end
      annotation('textarrow', [xPos, xPos], [yPosAux, yPos], ...
                 'String', Texto,'LineWidth',1,'FontWeight','bold','FontName','Arial',...
                 'TextColor',[1 1 1],'TextBackgroundColor',[1 0 0],...
                 'HorizontalAlignment','center');

   end

end
```

```
% ------------------------------------------------------------------------------------------
%              Función para mostrar los puntos de trabajo seleccionados en texto
%
%       VARIABLES DE ENTRADA
%
%
%       Vector_RPM:                 Vector conteniendo los valores a utilizar en el grid RPM.
%       Vector_Av:                  Vector conteniendo los valores a utilizar en el grid Av.
%       Indice_RPM:                 Vector conteniendo los índices de RPM a mostrar.
%       Indice_Av:                  Vector conteniendo los índices de Av a mostrar.
%       Qe:                         Matriz con los valores medios de Qe
%       nCarnot:                    Matriz con los valores medios de nCarnot
%       P0:                         Estructura con los siguientes campos:
%               Porcentaje:         Porcentaje de valores de Qemax a mostrar
%                                   (Porcentaje +- Variacion)
%               Variación:          Variación del porcentaje. (Porcentaje +- Variacion)
%
% ------------------------------------------------------------------------------------------


function Sel_PtosOP_Qe_Mostrar_Texto(Vector_RPM,Vector_Av,Indice_RPM,Indice_Av,...
                                     Qe,nCarnot,P0)

  % Obtiene el valor máximo de Qe
   Qe_Max=max(max(Qe));

   fprintf('\n\nQe max: %8.2f\t Porcentaje (%%): %6.3f\t\t Variación (%%):
%6.3f\tN°Puntos:%5d\n\n',...
           Qe_Max,P0.Porcentaje,P0.Variacion,length(Indice_RPM));

   fprintf('Índice RPM\tÍndice Av\tRPM\t\tAv\t\tQe(W)\t\tnCarnot\n');
   fprintf('-------------------------------------------------------------\n');

   for i=1:length(Indice_RPM)

      fprintf('%6d\t %6d\t\t %8.2f %8.2f\t %8.3f %7.4f\n',...
              Indice_RPM(i), Indice_Av(i),...
              Vector_RPM(Indice_RPM(i)),Vector_Av(Indice_Av(i)),...
              Qe(Indice_RPM(i),Indice_Av(i)),...
              nCarnot(Indice_RPM(i),Indice_Av(i)));
   end

end
```

```
% ------------------------------------------------------------------------------------------
%                Función para ajustar los datos al grid para TH
%
%       VARIABLES DE ENTRADA
%
%       Datos:                      Matriz con los datos brutos de los puntos de operación:
%                                       Col1: Índice para Vector_RPM_O (RPM)
%                                       Col2: Índice para Vector Av_O (Av%)
%                                       Col3: Variable a ajustar al grid (m, P1, P2, etc)
%
%       Temp_Test:                  Matriz con n filas conteniendo las temperaturas a utilizar
%                                      Col1: Delta_Te (°C)
%                                      Col2: Delta_Tc (°C)
%                                      COl3: Delta_Tsh (°C)
%                                      Col4: Delta_Tsc (°C)
%                                      COl5: TH (°C)
%
%       Vector_TH:                  Vector conteniendo los valores de TH
%
%       PorcLimite:                 Límite máximo admitido entre el valor máximo/mínimo
%                                   y el medio a partir del cual se incluye la coordenada
%                                   en [RPM_Sup Av_Sup]
%
%       VARIABLES DE SALIDA
%
%       Grid_Datos:                 Estructura conteniendo los siguientes campos:
%           Valores:                Matriz con los datos ajustados al Grid. Consta de 4
%                                   columnas:
%                                       Col1: Valor medio
%                                       Col2: Valor mínimo
%                                       Col3: Valor máximo
%                                       Col4: Número de datos brutos ajustados al punto
%           Min:                    Valor mínimo encontrado en los datos brutos
%           Max:                    Valor máximo encontrado en los datos brutos
%           PtosSuperan_Porc        Índices con los puntos que han superado PorcLimite
%
% ------------------------------------------------------------------------------------------


function Vector_Datos= ...
         Vector_Ajustar_Datos(Datos,Temp_Test,Vector_TH,PorcLimite)

   % Obtiene el paso para el Vector
   PasoVector=Vector_TH(2)-Vector_TH(1);

   % Obtiene el valor mínimo del vector para TH
   Vector_TH_Min=Vector_TH(1);

   % Inicializa el vector que contendrá los datos discretizados:
   % (1)-Dato (2)-Min (3)-Max (4)-N°Datos detectados
   Vector_Datos.Valores=zeros(length(Vector_TH),4);

   % Posiciona cada uno de los puntos de operación obtenidos sobre el punto
   % mas cercano del grid guardando los valores máximos, mínimos y medios
   % para cada uno de ellos.
   for i=1:size(Datos,1)

      % Obtiene el punto de operación a analizar
      TH0=Temp_Test(Datos(i,1),5);
      Valor=Datos(i,2);

      % Obtiene las coordenadas que le corresponden en el vector
      [i_TH,i_Aux]=Grid_Obt_Indices(TH0,1,Vector_TH_Min,1,PasoVector,1);

      % Si es la primera vez que se analiza ese punto de operación
      if Vector_Datos.Valores(i_TH,4)==0

         Vector_Datos.Valores(i_TH,1)=Valor;   % Guarda el valor medio
         Vector_Datos.Valores(i_TH,2)=Valor;   % Guarda el valor mínimo
         Vector_Datos.Valores(i_TH,3)=Valor;   % Guarda el valor máximo
         Vector_Datos.Valores(i_TH,4)=1;       % Guarda el número de datos

      else

         MediaActual=Vector_Datos.Valores(i_TH,1);
         MinimoActual=Vector_Datos.Valores(i_TH,2);
         MaximoActual=Vector_Datos.Valores(i_TH,3);
         NDatosActual=Vector_Datos.Valores(i_TH,4);
         Vector_Datos.Valores(i_TH,1)=...
              (MediaActual*NDatosActual+Valor)/(NDatosActual+1); % Guarda el nuevo valor medio

         % Si hay que actualizar el mínimo
         if Valor<MinimoActual
```

```
            Vector_Datos.Valores(i_TH,2)=Valor;
        end

        % Si hay que actualizar el máximo
        if Valor>MaximoActual
            Vector_Datos.Valores(i_TH,3)=Valor;
        end

        % Guarda el nuevo número de datos
        Vector_Datos.Valores(i_TH,4)=NDatosActual+1;

    end

end

Vector_Datos.Min=min(Datos(:,2));
Vector_Datos.Max=max(Datos(:,2));
Variacion_Max=(Vector_Datos.Max-Vector_Datos.Min)*0.01*PorcLimite;

% Calcula la diferencia en valor absoluto entre el valor medio y el
% valor minimo. Si el resultado es mayor que la variación máxima
% permitida activa el flag de error Error_Min.
Error_Min=abs((Vector_Datos.Valores(:,1)-Vector_Datos.Valores(:,2)))-Variacion_Max>0;

% Calcula la diferencia en valor absoluto entre el valor medio y el
% valor maximo. Si el resultado es mayor que la variación máxima
% permitida activa el flag de error Error_Max.
Error_Max=abs((Vector_Datos.Valores(:,1)-Vector_Datos.Valores(:,3)))-Variacion_Max>0;

% Obtiene los índices de los elementos del vector que han incumplido el límite establecido
[i_TH_Sup]=find(Error_Min+Error_Max>0);

Vector_Datos.Ptos_Superan_Porc=sort([i_TH_Sup]);

end
```

```
% ------------------------------------------------------------------------------------------
%                Función para la discretización de los puntos de trabajo
%                factibles para un conjunto de ciclos de refrigeración
%
%       VARIABLES DE ENTRADA
%
%       Fichero_Matlab_Out          Fichero donde se guardarán los datos obtenidos:
%                                   (Ciclos,Datos_Grid,Vector_RPM,Vector_Av)
%
%        Parametros_Grid:           Estructura con los parámetros de configuración
%                                   para el ajuste al GRID. Se utilizan los
%                                   siguientes campos:
%
%               Paso.RPM_Porc:      Paso en % sobre el rango de barrido para RPM
%               Paso.RPM_Abs:       Paso en valor absoluto para RPM. Utilizar uno u otro
%                                   y colocar el otro a cero.
%
%               Paso.Av_Porc:       Paso en % sobre el rango de barrido para Av
%               Paso.Av_Abs:        Paso en valor absoluto para Av. Utilizar uno u otro
%                                   y colocar el otro a cero.
%
%               Variables:          Estructura con dos campos por cada variable a tratar.
%                                   El primero indica si debe ajustarse al grid y el
%                                   segundo el porcentaje máximo de desviación antes
%                                   de incluirlo en los puntos a comprobar
%                   m,m_PorcentajeMax
%                   P1,P1_PorcentajeMax, P2,P2_PorcentajeMax
%                   Qe,Qe_PorcentajeMax, W,W_PorcentajeMax
%                   nCarnot.nCarnot_PorcentajeMax,COP,COP_PorcentajeMax
%                   COPc,COPc_PorcentajeMax
%                   T1,T1_PorcentajeMax,T2,T2_PorcentajeMax
%                   T3,T3_PorcentajeMax,T4,T4_PorcentajeMax
%                   DeltaTe,DeltaTe_PorcentajeMax,DeltaTc,DeltaTc_PorcentajeMax
%                   DeltaTsh,DeltaTsh_PorcentajeMax,DeltaTsc,DeltaTsc_PorcentajeMax
%                   TH,TH_PorcentajeMax
%
%       Ciclos:                     Estructura con los puntos de operación válidos.
%                                   Contiene los siguientes campos:
%
%               RPM:                Vector con los valores de RPM de los datos brutos
%               Av:                 Vector con los valores de apertura (%) de los datos brutos

%               PtosOP_Q:           Vector de tres componentes con los puntos de operación
%                                   aptos encontrados. Contiene tres columnas:
%                                       Col1: Índice para Temp_Test
%                                       Col2: Índice para RPM
%                                       Col3: Índice para Av
%                                       Col4: Caudal (kg/s)
%
%               P1:                 Vector de n filas con la presión (bares) en el evaporador
%                                   para cada una de las filas de Temp_Test
%               P2:                 Vector de n filas con la presión (bares) en el condensador
%                                   para cada una de las filas de Temp_Test
%               T1:                 Vector de n filas con la temperatura (ºC) a la entrada
%                                   del compresor para cada una de las filas de Temp_Test
%               T2:                 Vector de n filas con la temperatura (ºC) a la salida
%                                   del compresor para cada una de las filas de Temp_Test
%               T3:                 Vector de n filas con la temperatura (ºC) a la entrada
%                                   de la válvula de expansión para cada una de las filas de
%                                   Temp_Test
%               T4:                 Vector de n filas con la temperatura (ºC) a la salida de
%                                   la válvula de expansión para cada una de las filas
%                                   de Temp_Test
%               H14:                Vector de n filas con la diferencia de entalpía entre
%                                   los estados 1 y 4 en kJ/kg para cada una de las filas
%                                   de Temp_Test
%               H21:                Vector de n filas con la diferencia de entalpía entre
%                                   los estados 2 y 1 en kJ/kg para cada una de las filas
%                                   de Temp_Test
%               COPc:               Vector de nfilas con el COP de Carnot para cada una de
%                                   las filas de Temp_Test
%
%               Temp_Test:          Matriz con n filas conteniendo las temperaturas empleadas:
%                                      Col1: Delta_Te (ºC)
%                                      Col2: Delta_Tc (ºC)
%                                      Col3: Delta_Tsh (ºC)
%                                      Col4: Delta_Tsc (ºC)
%                                      Col5: TH (ºC)
%
%       VARIABLES DE SALIDA
%
```

```
%           Vector_TH:              Vector con los valores de TH del grid
%
%           Datos_Vector:           Estructura con un campo por cada una de las variables
%                                   ajustadas al grid:
%                                   (m, P1, P2, Qe, W, nCarnot, COP,COPc, T1, T2, T3, T4,
%                                   DeltaTe, DeltaTc, DeltaTsh, DeltaTsc y TH).
%                                   Cada uno de ellos es, a su vez, otra estructura con
%                                   estos campos:
%
%               Valores:            Matriz con los datos ajustados al Grid. Consta de 4
%                                   columnas:
%                                       Col1: Valor medio
%                                       Col2: Valor mínimo
%                                       Col3: Valor máximo
%                                       Col4: Número de datos brutos ajustados al punto
%               Min:                Valor mínimo encontrado en los datos brutos
%               Max:                Valor máximo encontrado en los datos brutos
%               PtosSuperan_Porc    Índices con los puntos que han superado PorcLimite
%
% ------------------------------------------------------------------------------------------


function [Vector_TH,Datos_Vector]=Vector_Ajustar_PtosOp_Ciclos_TH (Fichero_Matlab_Out,...
                                                                   Parametros_Grid,...
                                                                   Ciclos)


   % Obtiene los parámetros
   PasoGrid=Parametros_Grid.Paso;
   Variables=Parametros_Grid.Variables;


   % Obtiene el rango obtenido para TH
   RPM0=Ciclos.RPM(1);
   Av0=Ciclos.Av(1);
   TH_Min=fix(min(Ciclos.Temp_Test(:,5)));
   TH_Max=fix(max(Ciclos.Temp_Test(:,5)));

   % Si el paso de TH para el grid se ha indicado en % sobre el rango
   if PasoGrid.TH_Porc>0 && PasoGrid.TH_Abs==0

      % Obtiene el paso en valor absoluto a utilizar en el barrido de RPM
      PasoGrid.TH=0.01*PasoGrid.TH_Porc*(TH_Max-TH_Min);

   else

      PasoGrid.TH=PasoGrid.TH_Abs;

   end

   % Inicializa las variables de salida
   Datos_Grid.m=false; Datos_Grid.P1=false; Datos_Grid.P2=false;
   Datos_Grid.Qe=false;Datos_Grid.W=false;Datos_Grid.nCarnot=false;
   Datos_Grid.COP=false;Datos_Grid.COPc=false;Datos_Grid.T1=false;
   Datos_Grid.T2=false;Datos_Grid.T3=false;Datos_Grid.T4=false;
   Datos_Grid.DeltaTe=false;Datos_Grid.DeltaTc=false;Datos_Grid.DeltaTsh=false;
   Datos_Grid.DeltaTsc=false;Datos_Grid.TH=false;

   % Muestra el rango a analizar
   fprintf ('RANGO DE ESTUDIO:\n');
   fprintf('TH (min):%5.2f\t\tTH(max): %5.2f \t\tRPM:%8.2f \t\tAv:%5.2f\n',...
           TH_Min,TH_Max,RPM0,Av0);

   % Genera el vector a utilizar en el estudio
   [Vector_TH]=Vector_Generar(TH_Min,TH_Max,PasoGrid.TH);

   % Guarda en Datos_Temp el índice identificativo del ciclo
   Datos_Temp(:,1)=Ciclos.PtosOp_Q(:,1);

   % Si hay que tratar el caudal (m)
   if (Variables.m)

      % Almacena el valor del caudal en g/s en Datos_Temp
      Datos_Temp(:,2)= Ciclos.PtosOp_Q(:,4)*1000;

      % Ajusta los datos del caudal (g/s) al vector
      Datos_Vector.m=Vector_Ajustar_Datos(Datos_Temp,Ciclos.Temp_Test,Vector_TH,...
                                          Variables.m_PorcentajeMax);

      % Muestra los datos que incumplen el porcentaje establecido
```

```
    Vector_Mostrar_Info_Variable('m (g/s)',Datos_Vector.m,Vector_TH);

end


% Ajusta la presión P1 (bares) con los parámetros establecidos previamente
if (Variables.P1)

    % Almacena el valor de P1 (bares) en Datos_Temp
    Datos_Temp(:,2)= Ciclos.P1(Ciclos.PtosOp_Q(:,1));

    % Ajusta los datos de P1 (bares) al vector
    Datos_Vector.P1=Vector_Ajustar_Datos(Datos_Temp,Ciclos.Temp_Test,Vector_TH,...
                                         Variables.P1_PorcentajeMax);

    % Muestra los datos que incumplen el porcentaje establecido
    Vector_Mostrar_Info_Variable('P1 (bares)',Datos_Vector.P1,Vector_TH);

end


% Ajusta la presión P2 (bares) con los parámetros establecidos previamente
if (Variables.P2)

    % Almacena el valor de P2 (bares) en Datos_Temp
    Datos_Temp(:,2)= Ciclos.P2(Ciclos.PtosOp_Q(:,1));

    % Ajusta los datos de P2 (bares) al vector
    Datos_Vector.P2=Vector_Ajustar_Datos(Datos_Temp,Ciclos.Temp_Test,Vector_TH,...
                                        Variables.P2_PorcentajeMax);

    % Muestra los datos que incumplen el porcentaje establecido
    Vector_Mostrar_Info_Variable('P2 (bares)',Datos_Vector.P2,Vector_TH);

end


% Ajusta la potencia frigorífica Qe(W) con los parámetros establecidos previamente
if (Variables.Qe)

    % Almacena el valor de Qe (W) en Datos_Temp
    Datos_Temp(:,2)=Ciclos.H14(Ciclos.PtosOp_Q(:,1)).*...
                    Ciclos.PtosOp_Q(:,4)*1000;

    % Ajusta los datos de Qe (W) al vector
    Datos_Vector.Qe=Vector_Ajustar_Datos(Datos_Temp,Ciclos.Temp_Test,Vector_TH,...
                                         Variables.Qe_PorcentajeMax);

    % Muestra los datos que incumplen el porcentaje establecido
    Vector_Mostrar_Info_Variable('Qe(W)',Datos_Vector.Qe,Vector_TH);

end


% Ajusta el trabajo (W) con los parámetros establecidos previamente
if (Variables.W)

    % Almacena el valor de W (W) en Datos_Temp
    Datos_Temp(:,2)=Ciclos.H21(Ciclos.PtosOp_Q(:,1)).*...
                    Ciclos.PtosOp_Q(:,4)*1000;

    % Ajusta los datos de W (W) al vector
    Datos_Vector.W=Vector_Ajustar_Datos(Datos_Temp,Ciclos.Temp_Test,Vector_TH,...
                                        Variables.W_PorcentajeMax);

    % Muestra los datos que incumplen el porcentaje establecido
    Vector_Mostrar_Info_Variable('W(W)',Datos_Vector.W,Vector_TH);

end


% Ajusta el rendimiento de CARNOT con los parámetros establecidos previamente
if (Variables.nCarnot)

    % Almacena el valor de nCarnot en Datos_Temp
    Datos_Temp(:,2)= (Ciclos.H14(Ciclos.PtosOp_Q(:,1))./...
                      Ciclos.H21(Ciclos.PtosOp_Q(:,1)))./...
                      Ciclos.COPc(Ciclos.PtosOp_Q(:,1));

    % Ajusta los datos de nCarnot al vector
    Datos_Vector.nCarnot=Vector_Ajustar_Datos(Datos_Temp,Ciclos.Temp_Test,Vector_TH,...
                                              Variables.nCarnot_PorcentajeMax);
```

```
   % Muestra los datos que incumplen el porcentaje establecido
   Vector_Mostrar_Info_Variable('nCarnot',Datos_Vector.nCarnot,Vector_TH);

end


% Ajusta el COP con los parámetros establecidos previamente
if (Variables.COP)

   % Almacena el valor del COP en Datos_Temp
   Datos_Temp(:,2)=Ciclos.H14(Ciclos.PtosOp_Q(:,1))./...
                   Ciclos.H21(Ciclos.PtosOp_Q(:,1));


   % Ajusta los datos del COP al vector
   Datos_Vector.COP=Vector_Ajustar_Datos(Datos_Temp,Ciclos.Temp_Test,Vector_TH,...
                                         Variables.COP_PorcentajeMax);

   % Muestra los datos que incumplen el porcentaje establecido
   Vector_Mostrar_Info_Variable('COP',Datos_Vector.COP,Vector_TH);

end


% Ajusta el COPc con los parámetros establecidos previamente
if (Variables.COPc)

   % Almacena el valor del COPc en Datos_Temp
   Datos_Temp(:,2)=Ciclos.COPc(Ciclos.PtosOp_Q(:,1));

   % Ajusta los datos del COPc al vector
   Datos_Vector.COPc=Vector_Ajustar_Datos(Datos_Temp,Ciclos.Temp_Test,Vector_TH,...
                                          Variables.COPc_PorcentajeMax);

   % Muestra los datos que incumplen el porcentaje establecido
   Vector_Mostrar_Info_Variable('COPc',Datos_Vector.COPc,Vector_TH);

end


% Ajusta la temperatura T1 (°C) con los parámetros establecidos previamente
if (Variables.T1)

   % Almacena el valor de T1 (°C) en Datos_Temp
   Datos_Temp(:,2)= Ciclos.T1(Ciclos.PtosOp_Q(:,1));


   % Ajusta los datos de T1 (°C) al  vector
   Datos_Vector.T1=Vector_Ajustar_Datos(Datos_Temp,Ciclos.Temp_Test,Vector_TH,...
                                        Variables.T1_PorcentajeMax);

   % Muestra los datos que incumplen el porcentaje establecido
   Vector_Mostrar_Info_Variable('T1 (°C)',Datos_Vector.T1,Vector_TH);

end


% Ajusta la temperatura T2 (°C) con los parámetros establecidos previamente
if (Variables.T2)

   % Almacena el valor de T2 (°C) en Datos_Temp
   Datos_Temp(:,2)= Ciclos.T2(Ciclos.PtosOp_Q(:,1));

   % Ajusta los datos de T2 (°C) al vector
   Datos_Vector.T2=Vector_Ajustar_Datos(Datos_Temp,Ciclos.Temp_Test,Vector_TH,...
                                        Variables.T2_PorcentajeMax);

   % Muestra los datos que incumplen el porcentaje establecido
   Vector_Mostrar_Info_Variable('T2 (°C)',Datos_Vector.T2,Vector_TH);

end


% Ajusta la temperatura T3 (°C) con los parámetros establecidos previamente
if (Variables.T3)

   % Almacena el valor de T3 (°C) en Datos_Temp
   Datos_Temp(:,2)= Ciclos.T3(Ciclos.PtosOp_Q(:,1));

   % Ajusta los datos de T3 (°C) al vector
```

```
    Datos_Vector.T3=Vector_Ajustar_Datos(Datos_Temp,Ciclos.Temp_Test,Vector_TH,...
                                         Variables.T3_PorcentajeMax);

    % Muestra los datos que incumplen el porcentaje establecido
    Vector_Mostrar_Info_Variable('T3 (°C)',Datos_Vector.T3,Vector_TH);

end


% Ajusta la temperatura T4 (°C) con los parámetros establecidos previamente
if (Variables.T4)

    % Almacena el valor de T4 (°C) en Datos_Temp
    Datos_Temp(:,2)= Ciclos.T4(Ciclos.PtosOp_Q(:,1));

    % Ajusta los datos de T4 (°C) al vector
    Datos_Vector.T4=Vector_Ajustar_Datos(Datos_Temp,Ciclos.Temp_Test,Vector_TH,...
                                         Variables.T4_PorcentajeMax);

    % Muestra los datos que incumplen el porcentaje establecido
    Vector_Mostrar_Info_Variable('T4 (°C)',Datos_Vector.T4,Vector_TH);

end


% Ajusta la temperatura Delta_Te (°C) con los parámetros establecidos previamente
if (Variables.DeltaTe)

    % Almacena el valor de DeltaTe (°C) en Datos_Temp
    Datos_Temp(:,2)= Ciclos.Temp_Test(Ciclos.PtosOp_Q(:,1),1);

    % Ajusta los datos de DeltaTe (°C) al vector
    Datos_Vector.DeltaTe=Vector_Ajustar_Datos(Datos_Temp,Ciclos.Temp_Test,Vector_TH,...
                                              Variables.DeltaTe_PorcentajeMax);

    % Muestra los datos que incumplen el porcentaje establecido
    Vector_Mostrar_Info_Variable('Delta Te (°C)',Datos_Vector.DeltaTe,Vector_TH);

end


% Ajusta la temperatura Delta_Tc (°C) con los parámetros establecidos previamente
if (Variables.DeltaTc)

    % Almacena el valor de DeltaTc (°C) en Datos_Temp
    Datos_Temp(:,2)= Ciclos.Temp_Test(Ciclos.PtosOp_Q(:,1),2);

    % Ajusta los datos de DeltaTc (°C) al vector
    Datos_Vector.DeltaTc=Vector_Ajustar_Datos(Datos_Temp,Ciclos.Temp_Test,Vector_TH,...
                                              Variables.DeltaTc_PorcentajeMax);

    % Muestra los datos que incumplen el porcentaje establecido
    Vector_Mostrar_Info_Variable('Delta Tc (°C)',Datos_Vector.DeltaTc,Vector_TH);

end


% Ajusta la temperatura Delta_Tsh (°C) con los parámetros establecidos previamente
if (Variables.DeltaTsh)

    % Almacena el valor de DeltaTsh (°C) en Datos_Temp
    Datos_Temp(:,2)= Ciclos.Temp_Test(Ciclos.PtosOp_Q(:,1),3);

    % Ajusta los datos de DeltaTsh (°C) al vector
    Datos_Vector.DeltaTsh=Vector_Ajustar_Datos(Datos_Temp,Ciclos.Temp_Test,Vector_TH,...
                                               Variables.DeltaTsh_PorcentajeMax);

    % Muestra los datos que incumplen el porcentaje establecido
    Vector_Mostrar_Info_Variable('Delta Tsh (°C)',Datos_Vector.DeltaTsh,Vector_TH);

end


% Ajusta la temperatura Delta_Tsc (°C) con los parámetros establecidos previamente
if (Variables.DeltaTsc)

    % Almacena el valor de DeltaTsc (°C) en Datos_Temp
    Datos_Temp(:,2)= Ciclos.Temp_Test(Ciclos.PtosOp_Q(:,1),4);

    % Ajusta los datos de DeltaTsc (°C) al vector
    Datos_Vector.DeltaTsc=Vector_Ajustar_Datos(Datos_Temp,Ciclos.Temp_Test,Vector_TH,...
                                               Variables.DeltaTsc_PorcentajeMax);
```

```
      % Muestra los datos que incumplen el porcentaje establecido
      Vector_Mostrar_Info_Variable('Delta Tsc (°C)',Datos_Vector.DeltaTsc,Vector_TH);

   end


   % Guarda todas las variables para un posterior análisis en el fichero indicado
   if ~isempty(Fichero_Matlab_Out)
      delete (Fichero_Matlab_Out);
      fprintf ('\nGuardando variables en fichero Matlab.....\n');
      save (Fichero_Matlab_Out,'Ciclos','Datos_Vector','Vector_TH','-v7.3');
   end

   fprintf ('\nProceso finalizado correctamente.\n');

end
```

```
% ------------------------------------------------------------------------------------------
%     Función para obtener los vectores necesarios para la discretización de los datos
%
%       VARIABLES DE ENTRADA
%
%       Valor_Min:                  Valor mínimo de TH
%       Valor_Max_Ini:              Valor máximo inicial de TH
%       PasoVector:                 Paso del vector para TH
%
%       VARIABLES DE SALIDA
%
%       Vector:                     Vector para TH
%
% ------------------------------------------------------------------------------------------

function [Vector]=Vector_Generar(Valor_Min,Valor_Max_Ini,PasoVector)

   % Obtiene el vector
   Vector=[Valor_Min:PasoVector:Valor_Max_Ini];

   if Vector(length(Vector))<Valor_Max_Ini
      Vector(length(Vector)+1)=Vector(length(Vector))+PasoVector;
   end

end
```

```
% ------------------------------------------------------------------------------------------
%                Función para mostrar información sobre los puntos ajustados al grid
%
%       VARIABLES DE ENTRADA
%
%       Variable_Desc:              String descriptivo de la variable.
%       Vector_Datos:               Estructura con los siguientes campos:
%
%               Valores:            Matriz con los datos ajustados al Grid. Consta de 4
%                                   columnas:
%                                       Col1: Valor medio
%                                       Col2: Valor mínimo
%                                       Col3: Valor máximo
%                                       Col4: Número de datos brutos ajustados al punto
%               Min:                Valor mínimo encontrado en los datos brutos
%               Max:                Valor máximo encontrado en los datos brutos
%               PtosSuperan_Porc    Índices con los puntos que han superado PorcLimite
%

%       Vector_TH:                  Vector conteniendo los valores a utilizar en el grid
%                                   para TH.
% ------------------------------------------------------------------------------------------

function Vector_Mostrar_Info_Variable(Variable_Desc,Vector_Datos,Vector_TH)

   % Muestra por pantalla el parámetro y su valor mínimo y máximo en el grid
   fprintf('\nPARÁMETRO: %s \t VALOR MÍNIMO:%7.4f \t VALOR MÁXIMO:%7.4f \n', ...
           Variable_Desc,Vector_Datos.Min,Vector_Datos.Max);

   % Si hay datos que han incumplido el límite establecido
   if length(Vector_Datos.Ptos_Superan_Porc)>0

      % Muestra el mensaje informativo
      fprintf ('INCUMPLEN el límite establecido:\n\n');

      i_TH_Sup=Vector_Datos.Ptos_Superan_Porc;

      % Recorre los puntos del grid que han incumplido el límite
      for i=1:length(i_TH_Sup)

         Dato_Med=Vector_Datos.Valores(i_TH_Sup(i),1); % Valor medio en el punto
         Dato_Min=Vector_Datos.Valores(i_TH_Sup(i),2); % Valor mínimo en el punto
         Dato_Max=Vector_Datos.Valores(i_TH_Sup(i),3); % Valor máximo en el punto

         % Obtiene los porcentajes de error para el dato mínimo y máximo
         Porc_Min=100*abs((Dato_Med-Dato_Min)/(Vector_Datos.Max-Vector_Datos.Min));
         Porc_Max=100*abs((Dato_Med-Dato_Max)/(Vector_Datos.Max-Vector_Datos.Min));

         % Muestra la información por pantalla
         fprintf('Indice TH: %d\t\t',i_TH_Sup(i));
         fprintf('TH:% 5.2f \t\t',Vector_TH(i_TH_Sup(i)));
         fprintf('Min:% 9.3f(% 6.2f%%) \t\tMed:% 9.3f \t\t',Dato_Min,Porc_Min,Dato_Med);
         fprintf('Max:% 9.3f(% 6.2f%%) \n',Dato_Max,Porc_Max);

      end

   else
       fprintf('TODOS los puntos CUMPLEN el límite establecido.\n');
   end

end
```

D. REFERENCIAS

[1] Rasmussen, B. P. «Dynamic modeling and advanced control of air conditioning and refrigeration systems» PhD Thesis, University of Illinois, Urbana-Champaign, USA, 2005.

[2] Bejarano, G., Ortega, M.G., Rubio, F.R. y Morilla, F., «Modelado simplificado y orientado al control de sistemas de refrigeración». XXXIV Jornadas de Automática, Terrassa, 2013.

[3] Baxter, V. D. «Advances in supermarket refrigeration systems», Oak Ridge Natl. Lab.,Oak Ridge, Tennessee, pp. 37 831–6070, 2002.

[4] Pérez-Lombard, L., Ortiz, J., Pout, C. «A review on buildings energy consumption information. Energy and Buildings», 40, 3, 394-398.

[5] Morilla, F., «Optimización y Control Robusto Multivariable de Sistemas de Refrigeración DPI2012-37580-C02-01 y 02». XIII Simposio CEA de Ingeniería de Control (2015).

[6] Borgnakke – Sonntag, «Fundamentals of Thermodynamics». 8th edition.

[7] Wark, Kenneth., Richards, Donald E., «Termodinámica». Sexta edición.

[8] N., Alleyne, A., «Comparison of SISO and MIMO control techniques for a diagonally dominant vapor compression system. American Control Conference, 2009.

[9] Jain, N., Li, B., Keir, M., Hencey, B., Alleyne, A., «Decentralized Feedback Structures of a vapor compression cycle system». IEEE Trans. on Control Systems Technology, 18-1,185-193. 2010.

[10] Jain,N., «Thermodynamics-based optimization and control of integrated energy systems»,Ph.D. dissertation, University of Illinois, Urbana-Champaign, USA, 2013.

[11] Zhao, Lei., Cai, Wen-Jian., Ding, Xu-dong., Chang, Wei-chung., «Model-based optimization for vapor compression refrigeration cycle» (2013)

[12] Zhao, Lei., Cai, Wen-Jian., Ding, Xu-dong., Chang, Wei-chung.,«Decentralized optimization for vapor compression refrigeration cycle». (2012)

[13] Librerías termodinámicas. http://www.coolprop.org/.

[14] Programa de cálculos termodinámicos: http://en.ipu.dk/Indhold/refrigeration-and-energy-technology/coolpack.aspx

[15] Morilla, F, «Apuntes de Control Multivariable. Análisis de sistemas multivariables ». Master de Ingeniería de Sistemas y Control, 2013