



Universidad de Valladolid



**ESCUELA DE INGENIERÍAS
INDUSTRIALES**

UNIVERSIDAD DE VALLADOLID

ESCUELA DE INGENIERIAS INDUSTRIALES

Grado en Ingeniería Eléctrica

**Potencia óptima de los transformadores
asociados a plantas de energía solar
fotovoltaica**

Autor:

Caloca Mendo, Daniel

Tutor:

**Serrano Sanz, José Andrés
Departamento
Ingeniería Eléctrica**

Valladolid, junio de 2019.

Resumen:

La potencia nominal del transformador de una planta de generación fotovoltaica viene determinada por la máxima potencia que es capaz de entregar a través de sus inversores.

La peculiar entrega de potencia de las plantas fotovoltaicas, que tienen cada mediodía un pico de potencia, permite que el transformador a instalar sea de una potencia nominal más baja que la máxima entregada por la planta. Esta inferior potencia nominal generará un ahorro en el coste de adquisición del transformador.

Como consecuencia se producirá una sobrecarga a las horas del pico de potencia, que ocasionará una elevación de su temperatura interna por encima de la esperada en régimen nominal.

La empresa Ibérica Solar, propone al Departamento de Ingeniería Eléctrica de la Universidad de Valladolid, desarrollar un programa en Matlab, que tiene como objetivo calcular la potencia nominal óptima del transformador de la planta fotovoltaica, desde las perspectivas técnica y económica.

Palabras claves: transformador, Matlab, temperatura, planta fotovoltaica.

Índice

| | |
|---|----|
| ÍNDICE DE FIGURAS | 5 |
| ÍNDICE DE TABLAS | 6 |
| ÍNDICE DE FÓRMULAS..... | 6 |
| GLOSARIO DE TÉRMINOS Y ABREVIATURAS..... | 8 |
| 1.INTRODUCCIÓN..... | 10 |
| 1.1 INTRODUCCIÓN AL PROYECTO..... | 10 |
| 1.2 OBJETIVOS | 12 |
| 1.3 INTRODUCCIÓN SOBRE PLANTAS FOTOVOLTAICAS..... | 13 |
| 1.4 INTRODUCCIÓN SOBRE TRANSFORMADORES | 16 |
| 2. NORMATIVA | 18 |
| 2.1 NORMATIVA EUROPEA..... | 18 |
| 2.1 NORMATIVA AMERICANA | 25 |
| 3.PÉRDIDAS DEL TRANSFORMADOR | 29 |
| 4.OBTENCIÓN DE DATOS A TRAVES DE PVSYSY..... | 35 |
| 5.SOFTWARE | 38 |
| 5.1 PROGRAMACIÓN DE LECTURA DE DATOS..... | 38 |
| 5.2 PROGRAMACIÓN NORMATIVA IEC | 39 |
| 5.3 PROGRAMACIÓN NORMATIVA IEEE..... | 41 |
| 5.4 PROGRAMACIÓN ANÁLISIS DE DATOS | 43 |
| 5.5 INTERFAZ DEL PROGRAMA..... | 47 |
| 5.5 INSTALACIÓN DEL PROGRAMA | 54 |
| 6. ESTUDIO DE VARIAS PLANTAS..... | 57 |
| 7. CONCLUSIONES | 61 |
| 8. BIBLIOGRAFÍA..... | 62 |
| 8.1 FUENTES DE LAS IMÁGENES..... | 63 |
| ANEXO 1: CÓDIGO DEL PROGRAMA | 64 |
| ANEXO 2: FUNCIONES DEL PROGRAMA..... | 90 |
| ANEXO 3: VALORACIÓN FINAL DE LA EMPRESA IBÉRICA SOLAR | 92 |

ÍNDICE DE FIGURAS

Figura 1. Producción diaria planta fotovoltaica

Figura 2. Primer flujograma del programa

Figura 3. Instalación planta fotovoltaica

Figura 4. Panel fotovoltaico

Figura 5. Inversor

Figura 6. Elementos constructivos transformador.

Figura 7. Circuito equivalente de un transformador reducido al primario.

Figura 8. Función en 'steps'.

Figura 9. Flujo de Potencia planta fotovoltaica.

Figura 10. Límites de potencia P-Q.

Figura 11. Potencia activa Planta Fotovoltaica.

Figura 12. Potencia activa Planta Fotovoltaica con pérdidas.

Figura 13. Captura de pantalla PVsyst.

Figura 14. Captura de pantalla PVsyst.

Figura 15. Captura de pantalla PVsyst.

Figura 16. Excel obtenido de PVsyst.

Figura 17. Flujograma de Programa con normativa IEC

Figura 18. Flujograma de Programa con normativa IEEE

Figura 19. Gráficas potencia-temperatura frente a intervalo de tiempo

Figura 20. Gráficas potencia-temperatura frente a intervalo de tiempo

Figura 21. Gráficas potencia-temperatura frente a intervalo de tiempo

Figura 22. Gráficas potencia-temperatura frente a intervalo de tiempo

Figura 23. Interfaz del programa.1

Figura 24. Interfaz del programa.2

Figura 25. Interfaz del programa.3

Figura 26. Interfaz del programa.4

Figura 27. Interfaz del programa.5

Figura 28. Interfaz del programa.6

Figura 29. Interfaz del programa.7

Figura 30. Instalación del programa. Paso1

Figura 31. Instalación del programa. Paso2

Figura 32. Instalación del programa. Paso3

Figura 33. Instalación del programa. Paso4

Figura 34. Instalación del programa. Paso5

Figura 35. Instalación del programa. Paso6

Figura 36. La Solanilla 50MW.1

Figura 37. La Solanilla 50MW.2

Figura 38. La Solanilla 50MW.3

Figura 39. El_pedregal_10MW.1

Figura 40. El_pedregal_10MW.2

Figura 41. El_pedregal_10MW.3

ÍNDICE DE TABLAS

Tabla1. Temperatura límite (fuente IEC60076-7 parte 7, pp. 18)

Tabla2. Tabla de potencia de la planta fotovoltaica. (fuente PVsyst)

Tabla 3. Constantes térmicas, IEC (fuente IEC60076-7 parte 7, pp. 30)

Tabla 4. Constantes térmicas, IEEE (fuente IEE C57.91 95, pp. 47)

Tabla 5. Temperatura límite, IEEE (fuente IEE C57.91 95, pp. 55)

ÍNDICE DE FÓRMULAS

Fórmula 1. Relative ageing rate (factor de envejecimiento). (fuente IEC60076-7 parte 7)

Fórmula 2. Loss of life (horas). (fuente IEC60076-7 parte 7)

Fórmula 3. Hot-spot en aumento de temperatura. (fuente IEC60076-7 parte 7)

Fórmula 4. Incremento relativo del aumento de temperatura y el top-oil en estado estacionario. (fuente IEC60076-7 parte 7)

Fórmula 5. Gradiente de temperatura entre el hot-spot al top-oil en estado estacionario en caso de aumento de temperatura. (fuente IEC60076-7 parte 7)

Fórmula 6. Temperatura del top-oil inicial del siguiente step.

Fórmula 7. Temperatura del hot-spot inicial del siguiente step.

Fórmula 8. Hot-spot.

Fórmula 9. Top-oil.

Fórmula 10. Cálculo del hot-spot en disminución de temperatura. (fuente IEC60076-7 parte 7)

Fórmula 11. Gradiente de temperatura del hot-spot al top-oil en estado estacionario en caso de disminución de temperatura. (fuente IEC60076-7 parte 7)

Fórmula 12. Temperatura del top-oil inicial del siguiente step

Fórmula 13. Temperatura del hot-spot inicial del siguiente step.

Fórmula 14. Temperatura del hot-spot inicial para el primer step. (fuente IEC60076-7 parte 7)

Fórmula 15. Temperatura del top-oil inicial para el primer step. (fuente IEC60076-7 parte 7)

Fórmula 16. Temperatura media. (fuente IEC60076-7 parte 7)

Fórmula 17. Hot-spot. (fuente IEE C57.91 95)

Fórmula 18. Top-oil. (fuente IEE C57.91 95)

Fórmula 19. Variación de top-oil sobre la temperatura ambiente. (fuente IEE C57.91 95)

Fórmula 20. Variación inicial de top-oil sobre la temperatura ambiente. (fuente IEE C57.91 95)

Fórmula 21. Constante de tiempo del top-oil. (fuente IEE C57.91 95)

Fórmula 22. Constante de tiempo del top-oil. (fuente IEE C57.91 95)

Fórmula 23. Variación de hot-spot sobre la temperatura ambiente. (fuente IEE C57.91 95)

Fórmula 24. Variación inicial de hot-spot sobre la temperatura ambiente. (fuente IEE C57.91 95)

Fórmula 25. Valor nominal del aumento del hot-spot sobre el top-oil. (fuente IEE C57.91 95)

Fórmula 26. Eficiencia de un transformador. (fuente MÁQUINAS ELÉCTRICAS JESÚS FRAILE MORA quinta edición, edición Mc Graw Hill)

Fórmula 27. Índice de carga. (fuente MÁQUINAS ELÉCTRICAS JESÚS FRAILE MORA quinta edición, edición Mc Graw Hill)

Fórmula 28. Pérdidas en el cobre.

Fórmula 29. Pérdidas en el hierro.

Fórmula 30. Pérdidas de potencia activa de un transformador. (fuente MÁQUINAS ELÉCTRICAS JESÚS FRAILE MORA quinta edición, edición Mc Graw Hill)

Fórmula 31. Consumo de energía reactiva del Transformador. (fuente <https://fornieles.es/energia-reactiva/compensacion-reactiva-transformador/>)

Fórmula 32. Reactancia.

GLOSARIO DE TÉRMINOS Y ABREVIATURAS

Hot-spot: temperatura del punto más caliente de las bobinas del transformador de aceite.

Top-oil: temperatura del aceite almacenado en la cuba del transformador.

Relative thermal ageing rate: Tasa de envejecimiento por sobretemperatura.

Loss of life: horas de vida útil del transformador perdidas debido a la sobrecarga de potencia producida.

Índice de carga (C): Relación entre la carga y la carga correspondiente a la potencia nominal del transformador. Fórmula 27.

θ_h =hot-spot (°C).

$\Delta\theta_h$ =variación de temperatura del hot-spot (°C).

$\Delta\theta_{hi}$ =variación inicial de temperatura del hot-spot (°C).

θ_a =temperatura ambiente (°C).

θ_o =top-oil (°C).

$\Delta\theta_o$ =variación de temperatura del top-oil (°C).

$\Delta\theta_{oi}$ =variación inicial de temperatura del top-oil (°C).

V=relative ageing (factor de envejecimiento).

L=loss of life (pérdida en horas de vida útil del transformador).

θ_a =temperatura ambiente (°C).

K11=constante térmica de la norma europea para calcular la función f1.

K21=constante térmica de la norma europea para calcular la función f2.

K22=constante térmica de la norma europea para calcular la función f3.

H= factor de hot-spot que depende del bobinado del transformador.

R= relación entre las pérdidas a plena carga y las pérdidas en vacío.

τ_o = constante térmica de tiempo medio de aceite (min).

τ_w =constante térmica de tiempo de las bobinas (min).

$\Delta\theta_{or}$ =variación de temperatura del Top oil (°C).

x=exponente de la función potencial que da la expresión de las pérdidas totales referido al top-oil.

y=exponente de la función potencial que da la expresión de corriente frente al aumento de la temperatura del devanado.

θ_E = temperatura ambiente media (°C).

θ_{ya} = temperatura anual media (C°).

θ_{ma-max} = temperatura media del mes más caluroso (°C).

m = constante usada para el cálculo de la variación del hot-spot con cambios de carga.

n = constante usada para el cálculo de la variación del top-oil con cambios de carga.

$P_{T,R}$ = pérdida total en vatios a carga nominal.

$\Delta\theta_{H/A,R}$ = es el aumento de la temperatura del punto más caliente de las bobinas sobre la temperatura del aceite a carga nominal.

$\Delta\theta_{TO,R}$ = es el aumento de la temperatura del aceite sobre la temperatura ambiente en carga nominal.

P_{cc} = pérdidas del cobre a plena carga.

P_{Fe} = pérdidas en el hierro.

U_{cc} = tensión relativa de cortocircuito en %.

$\tau_{TO,R}$ = es la constante de tiempo para la carga nominal que comienza con el aumento inicial de la temperatura del aceite superior de 0 °C, horas

T = es la duración de la sobrecarga en horas.

1.INTRODUCCIÓN

1.1 INTRODUCCIÓN AL PROYECTO

La empresa Ibérica Solar propuso el presente proyecto con el principal objetivo de desarrollar un programa que permita dimensionar de manera óptima el transformador. En las plantas fotovoltaicas, los transformadores tienen una curva de carga como la mostrada en la figura 1.

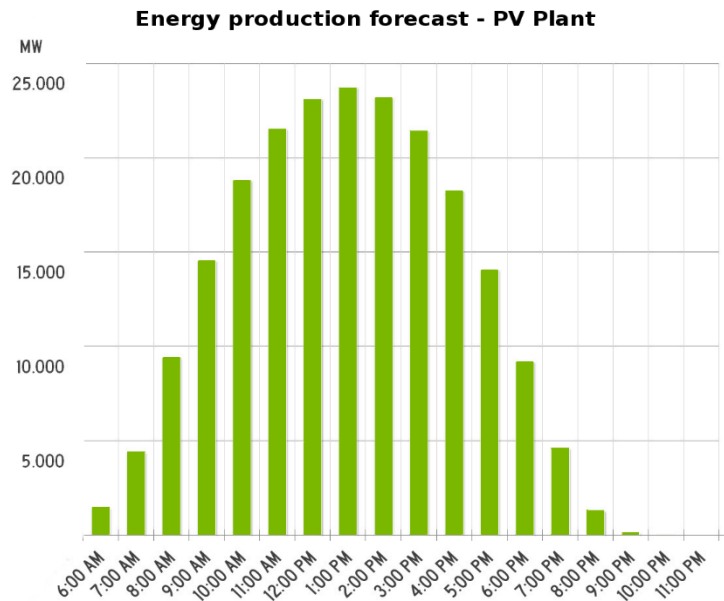


Figura 1. Producción diaria planta fotovoltaica

El transformador estará sometido a plena carga durante unas pocas horas del día, por lo que, si se utiliza un transformador cuya capacidad sea la potencia de pico de la planta fotovoltaica, este solo trabajará a plena carga durante 5 horas al día aproximadamente.

La normativa IEEE (Comité Electrotécnico americano) e IEC (Comité Electrotécnico europeo), con la que se trabaja en este proyecto nos explica la relación que tiene el transformador con el índice de carga y la temperatura interna de la máquina. Por lo que a medida que aumenta el índice de carga aumenta su temperatura interna. Estas normativas permiten una temperatura máxima del transformador. Se llega a la conclusión de que, aunque el transformador esté sobrecargado 5 horas al día si esta temperatura interna de la máquina no supera la establecida por la normativa vigente, obtendremos un transformador óptimo, en el sentido técnico-económico, para la planta fotovoltaica.

El programa que se quiere diseñar utilizará un Excel obtenido del PVsyst (Software de cálculo fotovoltaico) que proveerá la potencia activa obtenida a cada hora de la planta fotovoltaica a estudiar. El usuario introducirá una potencia inicial, inferior a la potencia instalada de la planta fotovoltaica, de iteración y una tolerancia de iteración. Se calculará la temperatura interna del transformador y si la temperatura supera la norma, se sumará a la potencia inicial la potencia de tolerancia. Al aumentar la potencia del transformador, el índice de carga disminuirá, disminuyendo a su vez la temperatura. Este ciclo se realizará hasta que la temperatura interna cumpla la de la norma. Esta idea es el principal objetivo del programa y su funcionamiento se puede ver en el primer flujograma del programa, figura 2.

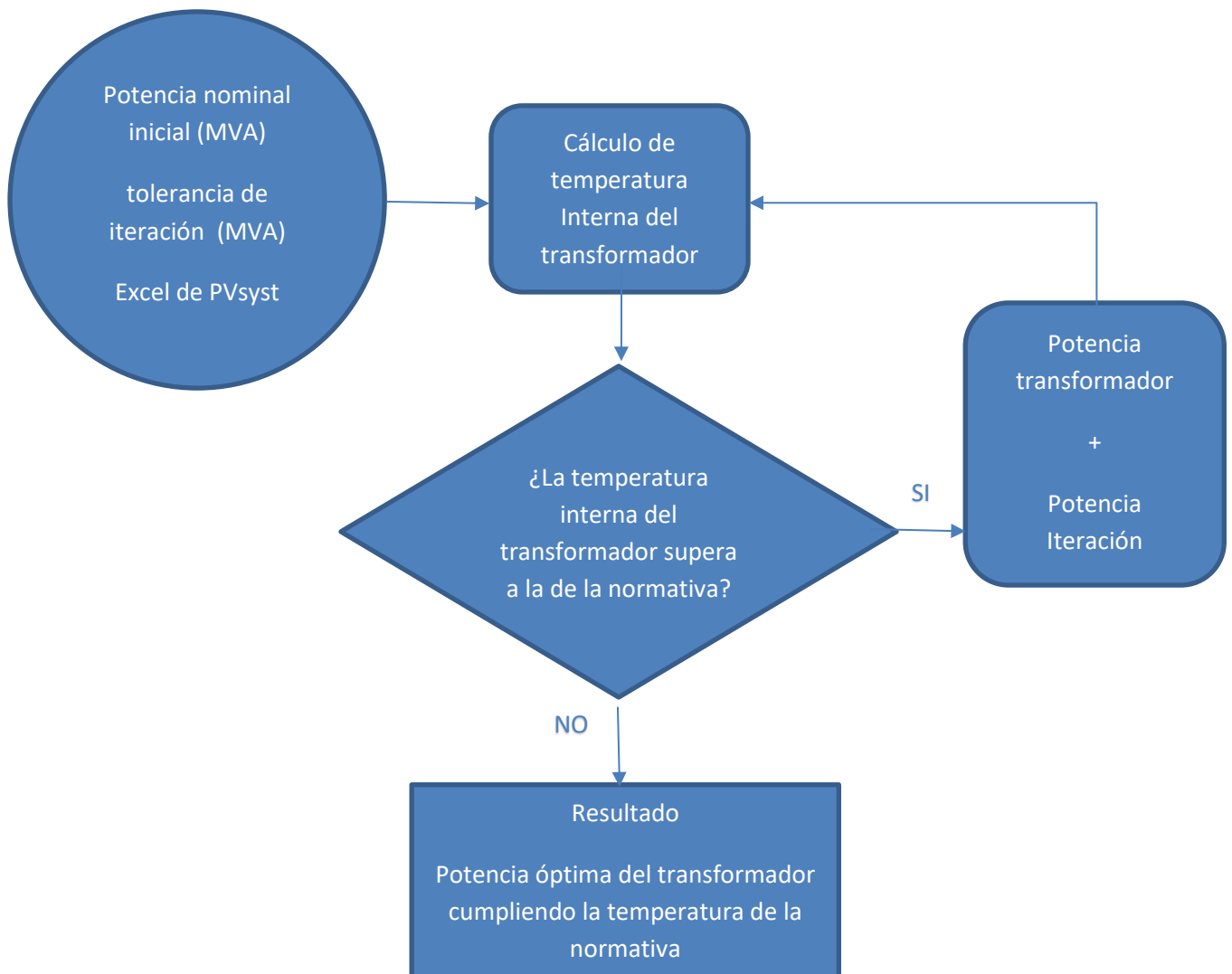


Figura 2. Primer flujograma del programa

1.2 OBJETIVOS

El objetivo principal del proyecto es el cálculo de la potencia nominal óptima del transformador desde el punto de vista técnico-económico (ajuste de la potencia nominal mínimo técnicamente viable).

Tras una serie de reuniones con la empresa Ibérica Solar se desarrollaron los siguientes objetivos secundarios:

-Calcular el envejecimiento del transformador debido al deterioro causado por un incremento de temperatura superior a la normal.

-Comparar las normas IEC e IEEE.

-Realizar un programa con la herramienta Matlab, que permita particularizarse para cualquier planta fotovoltaica que se desee.

-Visualizar los aumentos de temperatura del transformador de la planta fotovoltaica para un día en concreto

- **Paneles fotovoltaicos**

Al ser instalaciones destinadas a la generación de energía y no al autoconsumo, poseerán una gran cantidad de paneles fotovoltaicos. La cantidad de paneles y sus propias características determinarán la cantidad de energía que la planta fotovoltaica es capaz de entregar a la red.



Figura 4. Panel fotovoltaico

- **Inversores**

Estos podrán ser monofásicos o trifásicos dependiendo de las características de la planta y de la decisión del proyectista. Algunas de las funciones que realiza el inversor en las plantas fotovoltaicas son:

- Convertir la energía de corriente continua a corriente alterna para así poder transportarla.
- Optimización de la energía, maximizando la producción de las placas solares.
- Protección de la planta fotovoltaica, facilitando todo tipo de información en tiempo real sobre la situación de la planta solar.



Figura 5. Inversor

- **Protecciones**

Las instalaciones fotovoltaicas tienen todo tipo de protecciones tanto para la instalación como para las personas.

Las protecciones de la instalación son:

- Limitador de tensión máxima y mínima.
- Limitador de frecuencia máxima y mínima.
- Protección contra contactos directos.
- Protección contra sobrecarga.
- Protección contra cortocircuitos.
- Protección contra sobretensiones.

- **Contadores**

Realizan la medida de potencia que la planta fotovoltaica vende a la red.

- **Centro de transformación**

Para poder transportar la energía obtenida en la planta fotovoltaica se necesita aumentar la tensión de la planta fotovoltaica a una tensión de transporte para su posterior uso. El estudio se basará en esta parte de la planta fotovoltaica y en su correcto dimensionamiento. Más adelante profundizaremos en él, en su comportamiento en las plantas fotovoltaicas a lo largo de cada día.

Una vez repasado los elementos de una planta fotovoltaica y habiendo puesto un contexto a este trabajo, nos situaremos la mayoría de trabajo en la parte de transformación de la planta. Sin embargo, todos los elementos están relacionados entre sí, por lo que no dejaré de mencionarlos y tenerlos en cuenta en todo momento sin perder de vista el objetivo principal que es el dimensionar correctamente el transformador.

1.4 INTRODUCCIÓN SOBRE TRANSFORMADORES

Este proyecto se basa directamente en los transformadores de aceite por lo que se pondrá en contexto esta máquina eléctrica.

Esta máquina eléctrica, destinada para uso de corriente alterna, está formada por varios bobinados primarios y secundarios permitiendo transformar la energía eléctrica. Los transformadores tienen devanados de baja tensión y otros de alta tensión, el objetivo de estas máquinas es la elevación o disminución de tensión. En este caso los transformadores se encontrarán en plantas fotovoltaicas por lo que su objetivo será el de elevar la tensión a la salida de la planta fotovoltaica para su posterior transporte. Elevando la tensión conseguiremos reducir las pérdidas de energía eléctrica en el transporte hasta su llegada al punto de consumo. El devanado de baja tensión será el correspondiente a la planta fotovoltaica, mientras que el de alta tensión será el correspondiente al de la red de transporte.

En la figura 6 podemos observar los distintos elementos constructivos que posee un transformador convencional.

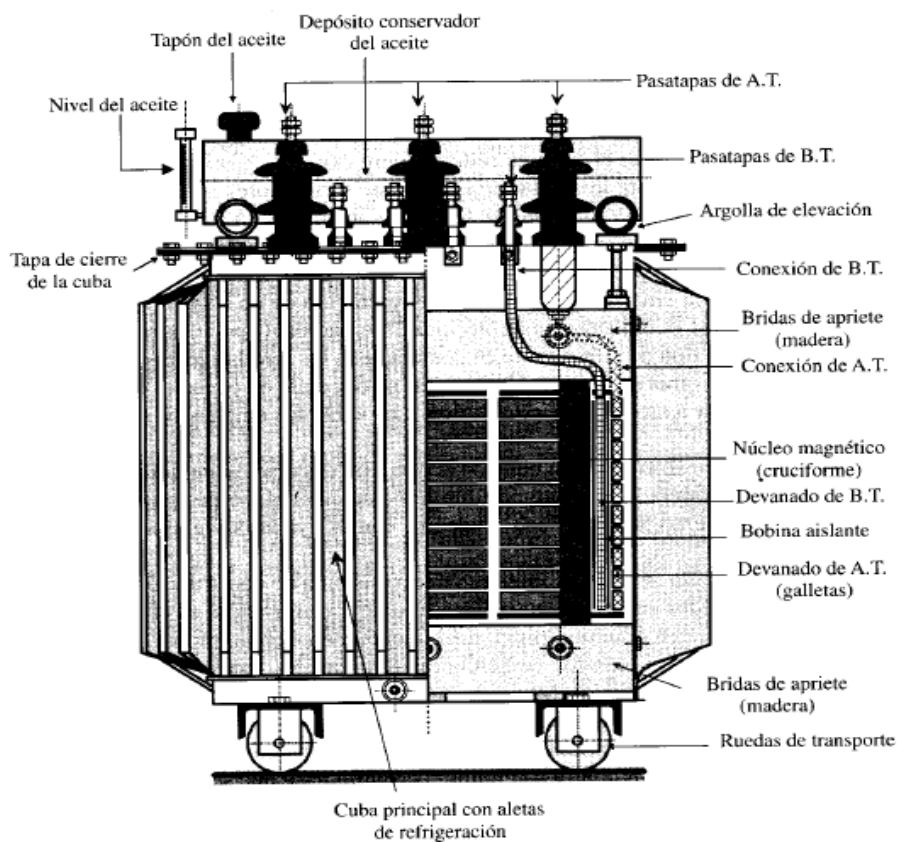


Figura 6. Elementos constructivos transformador.

El circuito equivalente del transformador es algo fundamental para mejorar su comprensión y realizar cualquier tipo de cálculo. Se puede observar en la Figura 7 el circuito del transformador equivalente reducido al primario. En este circuito el número de espiras del secundario es reducido a las del primario. También se pueden encontrar en paralelo la resistencia R_{Fe} cuyas pérdidas de efecto Joule son las pérdidas en el hierro del transformador y la reactancia X_{μ} por la que circula la corriente de magnetización.

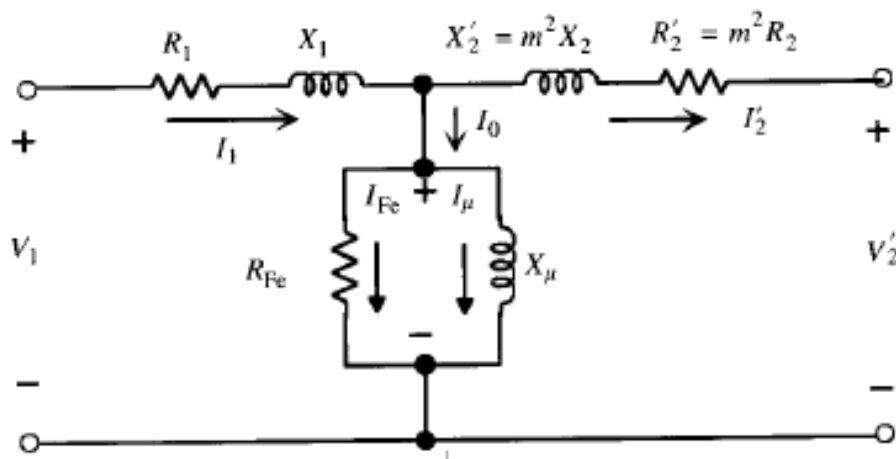


Figura 7. Circuito equivalente de un transformador reducido al primario.

2. NORMATIVA

El programa se contempla en dos normas sobre la carga de los transformadores refrigerados por aceite:

-IEC 60076-7 parte 7 (Comité Electrotécnico Europeo), en lo sucesivo se llamará 'norma europea'.

-IEEE, (Comité Electrotécnico Americano), en lo sucesivo se llamará 'norma americana'.

Se hablará de cada una de ellas, centrándose en el tema que nos interesa que es el del cálculo de la temperatura en la situación de carga de una planta fotovoltaica. Ambas normas tienen cosas en común, sin embargo, hay cálculos que se realizan de forma diferente.

2.1 NORMATIVA EUROPEA

LOSS OF LIFE (PÉRDIDA DE VIDA ÚTIL)

La norma europea trata del estado del transformador cuando éste se encuentra en situaciones de carga más allá de las dispuestas en su placa de características.

El envejecimiento que sufre el transformador ante estas sobrecargas es uno de los principales análisis que formula la normativa.

$$V = e^{\left(\frac{15000}{110+273} - \frac{15000}{\theta_h+273}\right)}$$

Fórmula 1. Relative ageing rate (factor de envejecimiento).

Con la fórmula 1, donde θ_h es el *hot-spot* (°C) se obtiene el *relative ageing rate*.

Cada hora del día obtendrá un *relative ageing rate* dependiendo del *hot-spot* del transformador.

La pérdida de vida del transformador se obtiene mediante la fórmula 2.

$$L = \sum_{n=1}^N V_n * t_n$$

Fórmula 2. Loss of life (horas)

Donde:

- V_n es el factor de envejecimiento durante el intervalo n.
- t_n es el tiempo del intervalo n.

Este sumatorio calcula la pérdida de vida en horas del transformador, la cual depende exclusivamente de la temperatura del *hot-spot*.

LÍMITES DE CORRIENTE Y TEMPERATURA PARA CARGAS SUPERIORES A LAS DESCRITAS EN LA PLACA DE CARACTERÍSTICAS

En la tabla 1 podemos observar que dependiendo del tamaño del transformador los límites de temperatura cambian. Para esta aplicación de transformadores de plantas fotovoltaicas, siempre se cumplirán que son *Large Power Transformers* aquellos de más de 100MVA y *Medium Power Transformers* los de más de 2.500KVa.

También en esta aplicación siempre se encontrarán en *long-time emergency load*, es decir en tiempos de sobrecarga de más de 30 min.

Por lo que atendiendo a estas dos cuestiones anteriores y observando la tabla 1, se llega a la conclusión del que el *hot-spot* y el *top-oil* tendrán una temperatura límite de 140°C y 115 °C respectivamente.

| Types of loading | Distribution transformers (see Note) | Medium power transformers (see Note) | Large power transformers (see Note) |
|--|--------------------------------------|--------------------------------------|-------------------------------------|
| Normal cyclic loading | | | |
| Current (p.u.) | 1,5 | 1,5 | 1,3 |
| Winding hot-spot temperature and metallic parts in contact with cellulosic insulation material (°C) | 120 | 120 | 120 |
| Other metallic hot-spot temperature (in contact with oil, aramid paper, glass fibre materials) (°C) | 140 | 140 | 140 |
| Top-oil temperature (°C) | 105 | 105 | 105 |
| Long-time emergency loading | | | |
| Current (p.u.) | 1,8 | 1,5 | 1,3 |
| Winding hot-spot temperature and metallic parts in contact with cellulosic insulation material (°C) | 140 | 140 | 140 |
| Other metallic hot-spot temperature (in contact with oil, aramid paper, glass-fibre materials) (°C) | 160 | 160 | 160 |
| Top-oil temperature (°C) | 115 | 115 | 115 |
| Short-time emergency loading | | | |
| Current (p.u.) | 2,0 | 1,8 | 1,5 |
| Winding hot-spot temperature and metallic parts in contact with cellulosic insulation material (°C) | See 7.2.1 | 160 | 160 |
| Other metallic hot-spot temperature (in contact with oil, aramid paper, glass fibre materials) (°C) | See 7.2.1 | 180 | 180 |
| Top-oil temperature (°C) | See 7.2.1 | 115 | 115 |
| NOTE The temperature and current limits are not intended to be valid simultaneously. The current may be limited to a lower value than that shown in order to meet the temperature limitation requirement. Conversely, the temperature may be limited to a lower value than that shown in order to meet the current limitation requirement. | | | |

Tabla1. Temperatura límite

TOP-OIL Y HOT-SPOT

La norma habla de dos maneras de obtener estos valores, aunque solo se tratará con una, ya que la otra se utiliza en el caso de que obtengamos datos de manera monitorizada de la planta fotovoltaica.

Como el programa sólo se usará para predecir, no tiene sentido la otra manera de obtener el *hot-spot* y *top-oil*.

La forma de obtener estos datos se basa en una función *step*. Esto quiere decir que la curva de potencia que obtendremos del Pvsyst la convertiremos en una función dividida en *steps* o saltos de potencia constante durante una hora.

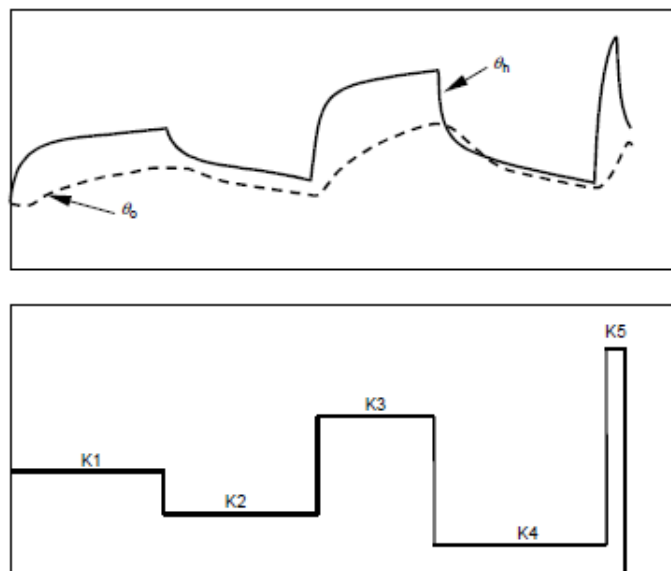


Figura 8. Función en 'steps'.

Cada salto siempre será de 1 hora ya que del Pvsyst se obtendrá lo mostrado en la tabla 2.

Como se observa en la tabla 2 la potencia a las 09:00 de la planta fotovoltaica que se quiere analizar es de 52,8 MW y a las 10:00 la potencia obtenida es de 142,6 MW. En la realidad la potencia entregada no aumentará de esa manera, sin embargo, la norma permite que se simule así:

La planta de 09:00 a 10:00 produce 52,8MW, y que de 10:00 a 11:00 produce 142,6MW.

| Hora | Temperatura ambiente | Potencia(W) |
|-------|----------------------|-------------|
| 0:00 | 8,9999 | -223596 |
| 1:00 | 7,9 | -223596 |
| 2:00 | 6,9997 | -223596 |
| 3:00 | 6,4002 | -223596 |
| 4:00 | 5,8998 | -223596 |
| 5:00 | 5,4001 | -223596 |
| 6:00 | 6,5998 | -223596 |
| 7:00 | 10 | 19008603 |
| 8:00 | 13,1 | 24984299 |
| 9:00 | 16,9 | 52882391 |
| 10:00 | 20,8 | 142625173 |
| 11:00 | 23,3 | 155120493 |
| 12:00 | 24,4 | 161447667 |
| 13:00 | 24,7 | 165896762 |
| 14:00 | 24,5 | 161757108 |
| 15:00 | 22,6 | 146600401 |
| 16:00 | 19,5 | 69368578 |
| 17:00 | 17,6 | -223161 |
| 18:00 | 16,3 | -223161 |
| 19:00 | 15,3 | -223161 |
| 20:00 | 14,8 | -223161 |
| 21:00 | 14,3 | -223161 |
| 22:00 | 13,6 | -223161 |
| 23:00 | 12,9 | -223161 |

Tabla 2. Tabla de potencia de la planta fotovoltaica.

La función dará saltos de potencia cada hora y se calculará un *hot-spot* y un *top-oil* para cada hora, se pueden encontrar 3 casos para el cálculo de estos datos.

CASO 1: EL STEP N ES MAYOR QUE EL STEP N+1

En este caso se usará la fórmula 3, la cual se puede desglosar para entenderse mejor.

$$\theta_h(t) = \theta_a + \Delta\theta_{oi} + \left[\Delta\theta_{or} * \left[\frac{1 + R * K^2}{1 + R} \right]^x - \Delta\theta_{oi} \right] * f1(t) + \Delta\theta_{hi} \\ + (H * gr * K^y - \Delta\theta_{hi}) * f2(t)$$

Fórmula 3. Hot-spot en aumento de temperatura.

$$f1(t) = (1 - e^{(-t)/(k_{11} * \tau_0)})$$

Fórmula 4. Incremento relativo del aumento de temperatura del top-oil en estado estacionario.

$$f2(t) = k_{21} * \left(1 - e^{\frac{-t}{k_{22} * \tau_w}} \right) - (k_{21} - 1) * \left(1 - e^{\frac{-t}{\tau_0/k_{22}}} \right)$$

Fórmula 5. Gradiente de temperatura entre el hot-spot y el top-oil en estado estacionario en caso de aumento de temperatura.

La fórmula 3 se puede desglosar en tres partes:

$$\Delta\theta_{oi_{n+1}} = \Delta\theta_{oi} + \left[\Delta\theta_{or} * \left[\frac{1 + R * K^2}{1 + R} \right]^x - \Delta\theta_{oi} \right] * f1(t) \theta_h(t)$$

Fórmula 6. Temperatura del top-oil inicial del siguiente step.

$$\Delta\theta_{hi_{n+1}}(t) = \Delta\theta_{hi} + (H * gr * K^y - \Delta\theta_{hi}) * f2(t)$$

Fórmula 7. Temperatura del hot-spot inicial del siguiente step.

$$\theta_h(t) = \theta_a + \Delta\theta_{oi_{n+1}} + \Delta\theta_{hi_{n+1}}$$

Fórmula 8. Hot-spot.

$$\theta_o(t) = \theta_a + \Delta\theta_{oi_{n+1}}$$

Fórmula 9. Top-oil.

En la fórmula 8 se comprueba mucho mejor como el *hot-spot* es la suma de la temperatura ambiente, más la variación de temperatura en el *top-oil* y más la variación de temperatura en el *hot-spot*. En cada 'step' de 1 hora se calculará el *hot-spot* y de ella se obtendrán los valores de $\Delta\theta_{oi}$ y de $\Delta\theta_{hi}$ para el siguiente step. En la fórmula 8 usada en el step 'n' se calculará que $\Delta\theta_{oi_{n+1}}$ será el $\Delta\theta_{oi}$ del step 'n+1', e igual con $\Delta\theta_{hi_{n+1}}(t)$ y $\Delta\theta_{hi}$. En el caso del primer step se calcularán unos valores iniciales ya que no se sabrán $\Delta\theta_{oi}$ y de $\Delta\theta_{hi}$ para el primer step.

CASO 2: EL STEP N ES MENOR QUE STEP N+1

$$\theta_h(t) = \theta_a + \Delta\theta_{or} * \left[\frac{1 + R * C^2}{1 + R} \right]^x + \left[\Delta\theta_{oi} - \Delta\theta_{or} * \left[\frac{1 + R * C^2}{1 + R} \right]^x \right] * f3(t) + (H * gr * C^y)$$

Fórmula 10. Hot-spot en disminución de temperatura.

$$f3(t) = e^{(-t)/(k_{11} * \tau_0)}$$

Fórmula 11. Gradiente de temperatura del top-oil a la temperatura ambiente en estado estacionario en caso de disminución de temperatura.

La fórmula 9 al igual que la 3 se puede desglosar de la misma manera.

$$\Delta\theta_{oi_{n+1}} = \Delta\theta_{or} * \left[\frac{1 + R * C^2}{1 + R} \right]^x + \left[\Delta\theta_{oi} - \Delta\theta_{or} * \left[\frac{1 + R * C^2}{1 + R} \right]^x \right] * f3(t)$$

Fórmula 12. Temperatura del top-oil inicial del siguiente step.

$$\Delta\theta_{hi_{n+1}}(t) = (H * gr * C^y)$$

Fórmula 13. Temperatura del hot-spot inicial del siguiente step.

En este caso tanto $\Delta\theta_{oi_{n+1}}$ como $\Delta\theta_{hi_{n+1}}$ cambian con respecto al caso anterior, pero se mantiene el mismo concepto. En la fórmula 8 usada en el step 'n' se calculará que $\Delta\theta_{oi_{n+1}}$ será el $\Delta\theta_{oi}$ del step 'n+1', e igual con $\Delta\theta_{hi_{n+1}}(t)$ y $\Delta\theta_{hi}$

CASO 3: VALORES INICIALES

Tanto la fórmula 14 como la fórmula 15, se utilizarán para calcular el *hot-spot* y el *top-oil* del primer step de todo, y así tener un primer punto de referencia.

$$\Delta\theta_{hi_initial}(t) = (k_{21} * \Delta\theta_{hr} * C^y) - [(k_{21} - 1) * \Delta\theta_{hr} * C^y]$$

Fórmula 14. Temperatura del *hot-spot* inicial para el primer step.

$$\Delta\theta_{oi_initial}(t) = \theta_a + \Delta\theta_{or} * \left[\frac{(1 + R * C^2)}{(1 + R)} \right]^x$$

Fórmula 15. Temperatura del *top-oil* inicial para el primer step.

CONSTANTES TÉRMICAS EN FUNCIÓN DEL TIPO DE TRANSFORMADOR

Dependiendo del tipo de refrigeración que posea el transformador se usarán unas constantes u otras. Estas constantes se utilizarán para calcular tanto el *hot-spot* como el *top-oil*. Las constantes se obtendrán de la tabla 3.

| | Distribution transformers | Medium and large power transformers | | | | | | |
|------------------------|---------------------------|-------------------------------------|----------------------------|------|----------------------------|------|--------------------------|-----|
| | | ONAN | ONAN restricted (see Note) | ONAN | ONAF restricted (see Note) | ONAF | OF restricted (see Note) | OF |
| Oil exponent x | 0,8 | 0,8 | 0,8 | 0,8 | 0,8 | 1,0 | 1,0 | 1,0 |
| Winding exponent y | 1,6 | 1,3 | 1,3 | 1,3 | 1,3 | 1,3 | 1,3 | 2,0 |
| Constant k_{11} | 1,0 | 0,5 | 0,5 | 0,5 | 0,5 | 1,0 | 1,0 | 1,0 |
| Constant k_{21} | 1,0 | 3,0 | 2,0 | 3,0 | 2,0 | 1,45 | 1,3 | 1,0 |
| Constant k_{22} | 2,0 | 2,0 | 2,0 | 2,0 | 2,0 | 1,0 | 1,0 | 1,0 |
| Time constant τ_0 | 180 | 210 | 210 | 150 | 150 | 90 | 90 | 90 |
| Time constant τ_w | 4 | 10 | 10 | 7 | 7 | 7 | 7 | 7 |

NOTE If a winding of an ON or OF-cooled transformer is zigzag-cooled, a radial spacer thickness of less than 3 mm might cause a restricted oil circulation, i.e. a higher maximum value of the function $f_2(t)$ than obtained by spacers ≥ 3 mm.

Tabla 3. Constantes térmicas, IEC

TEMPERATURA AMBIENTE

La temperatura ambiente es muy importante para el cálculo del *hot-spot* y el *top-oil*, la norma explica una manera de calcular la temperatura media. En este apartado se expondrá utilizando la fórmula 16, sin embargo, no la usaré en la programación ya que del Pvsyst se obtendrá la temperatura horaria de cada día, la cual será mucho más exacta que si calcula una media actual.

$$\theta_E = \theta_{ya} + 0.01 * [2 * (\theta_{ma-max} - \theta_{ya})]^{1.85}$$

Fórmula 16. Temperatura media.

2.1 NORMATIVA AMERICANA

En este apartado se hablará sobre la norma americana, la cual calcula las mismas variables que la normativo europea, pero utilizando diferentes formas de cálculo.

LOSS OF LIFE (PÉRDIDA DE VIDA ÚTIL)

El cálculo de envejecimiento del transformador es igual que en la norma europea. Por lo que se usarán las fórmulas ya comentadas anteriormente (fórmula 1 y fórmula 2).

TOP-OIL Y HOT-SPOT

El cálculo del *top-oil* y el *hot-spot* varía con respecto a la otra normativa. En esta normativa no importará si la temperatura aumenta o disminuye siempre se utilizarán las siguientes fórmulas 17-25 para el cálculo del *hot-spot* y el *top-oil*.

$$\theta_h = \theta_a + \Delta\theta_{TO} + \Delta\theta_H$$

Fórmula 17. Hot-spot

$$\theta_{TO}(t) = \theta_a + \Delta\theta_{TO}$$

Fórmula 18. Top-oil

En este caso la diferencia entre $\Delta\theta_{TO,i}$ y $\Delta\theta_{TO,U}$ es la variación de temperatura del step anterior y del step actual respectivamente. $\Delta\theta_{TO,U}$ se calcula de la misma forma que $\Delta\theta_{TO,i}$, pero tienen índice de carga diferente(K).

Por ejemplo, si se calcula el *top-oil* para el step 4, el cual sería entre la hora 3 y la hora 4 del primer día de la planta, se utilizaría para $\Delta\theta_{TO,U}$ el índice de carga del step 4 y para $\Delta\theta_{TO,i}$ el índice de carga del step 3.

$$\Delta\theta_{TO} = (\Delta\theta_{TO,U} - \Delta\theta_{TO,i}) * \left(1 - e^{\frac{-1}{\tau_{TO}}}\right) + \Delta\theta_{TO,i}$$

Fórmula 19. Variación de top-oil sobre la temperatura ambiente

$$\Delta\theta_{TO,i} = \Delta\theta_{TO,R} * \left[\frac{(R * C^2 + 1)}{(R + 1)}\right]^n$$

Fórmula 20. Variación inicial de top-oil sobre la temperatura ambiente

Las fórmulas 21 y 22 son constantes que habrá que calcular para poder calcular el *hot-spot* y el *top-oil*.

$$\tau_{TO,R} = \frac{C * \Delta\theta_{TO,R}}{P_{T,R}}$$

Fórmula 21. Constante de tiempo del top-oil

$$\tau_{TO} = \tau_{TO,R} \frac{\frac{\left[\frac{\Delta\theta_{TO,U}}{\Delta\theta_{TO,R}}\right] - \left[\frac{\Delta\theta_{TO,i}}{\Delta\theta_{TO,R}}\right]}{\left[\frac{\Delta\theta_{TO,U}}{\Delta\theta_{TO,R}}\right]^{\frac{1}{n}} - \left[\frac{\Delta\theta_{TO,i}}{\Delta\theta_{TO,R}}\right]^{\frac{1}{n}}}}$$

Fórmula 22. Constante de tiempo del top-oil

El cálculo de la variación del *hot-spot* es similar al de la variación del *top-oil*. El índice de carga K variará dependiendo del step que se esté calculando.

$$\Delta\theta_H = (\Delta\theta_{H,U} - \Delta\theta_{H,i}) * \left(1 - e^{\frac{-t}{\tau_H}}\right) + \Delta\theta_{H,i}$$

Fórmula 23. Variación de hot-spot sobre la temperatura ambiente

$$\Delta\theta_{H,i} = \Delta\theta_{H,R} * C^{2m}$$

Fórmula 24. Variación inicial de hot-spot sobre la temperatura ambiente

$$\Delta\theta_{H,R} = \Delta\theta_{H/A,R} * \Delta\theta_{TO,R}$$

Fórmula 25. Valor nominal del aumento del hot-spot sobre el top-oil.

CONSTANTES TÉRMICAS EN FUNCIÓN DEL TIPO DE TRANSFORMADOR

Dependiendo del tipo de refrigeración que posea el transformador se usarán unas constantes u otras. Estas constantes se utilizarán para calcular tanto el *hot-spot* como el *top-oil*. Estas constantes se obtendrán de la tabla 4.

| Type of cooling | <i>m</i> | <i>n</i> |
|-------------------------|----------|----------|
| OA | 0.8 | 0.8 |
| FA | 0.8 | 0.9 |
| Non-directed FOA or FOW | 0.8 | 0.9 |
| Directed FOA or FOW | 1.0 | 1.0 |

Tabla 4. Constantes térmicas, IEEE

LÍMITES DE CORRIENTE Y TEMPERATURA PARA CARGAS MÁS ALLÁ DE LA PLACA DE CARACTERÍSTICAS

Las temperaturas límites para esta normativa serán de 140 °C el *hot-spot*. Se utilizará la columna de Normal life Expectancy loading ya que estas sobrecargas de las plantas serán diarias y en ningún caso serán de forma extraordinaria

| | Normal life expectancy loading | Planned loading beyond nameplate rating | Long-time emergency loading | Short-time emergency loading |
|---|---------------------------------------|--|------------------------------------|-------------------------------------|
| Insulated conductor hottest-spot temperature, °C | 120 ^a | 130 | 140 | 180 ^b |
| Other metallic hot-spot temperature (in contact and not in contact with insulation), °C | 140 | 150 | 160 | 200 |
| Top-oil temperature, °C | 105 | 110 | 110 | 110 |

^a100 °C on a continuous 24 h basis

^bGassing may produce a potential risk to the dielectric strength of the transformer. This risk should be considered when this guide is applied refer to annex A.

Tabla 5. Temperatura límite, IEEE

3. PÉRDIDAS DEL TRANSFORMADOR

Uno de los factores más importantes que se ha visto en el apartado anterior de normativa es el índice de carga, los cálculos de temperatura explicados dependen de esta variable. El índice de carga depende de la potencia que le llega al transformador. PVsyst refleja los datos de potencia diarios del transformador en el lado de baja tensión, por lo que si se utiliza el índice de carga del lado de baja del transformador se estarían despreciando las pérdidas.

En la figura 9 se puede observar un esquema del flujo de potencia de una planta fotovoltaica y todos los cálculos de las normativas se deben realizar con el factor de potencia en el punto 1 de la figura 9.

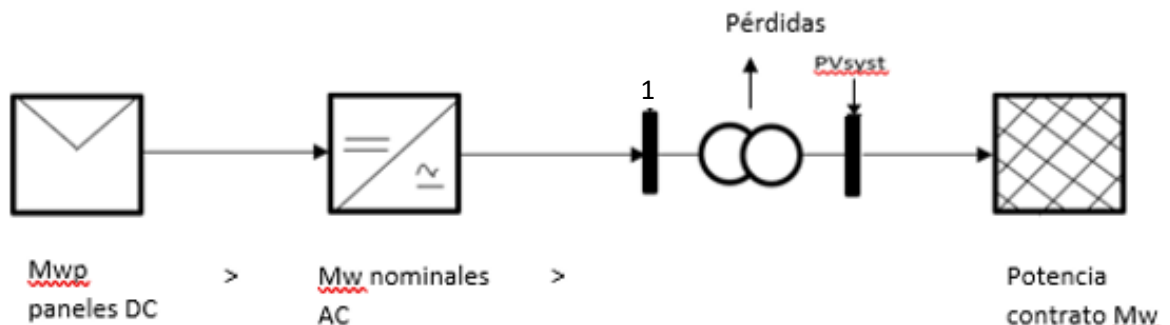


Figura 9. Flujo de Potencia planta fotovoltaica.

Para calcular el *hot-spot* y el *top-oil* de manera correcta se sumará las pérdidas de cada *step* de potencia a la potencia obtenida en el PVsyst.

La empresa Ibérica Solar trabaja con contratos eléctricos los cuales les obligan a verter a la red una potencia reactiva y es por ello por lo que el programa desarrollado permite la opción de elegir la potencia reactiva que se debe verter a la red eléctrica.

Esta potencia reactiva será una potencia fija en MVAR o se fijará con un factor de potencia, tal y como se observa en la gráfica de potencia de la Figura 10. En esta figura se muestra el vector de potencia aparente en el caso de que la planta estuviese fijada mediante un factor de potencia o mediante una potencia reactiva fija.

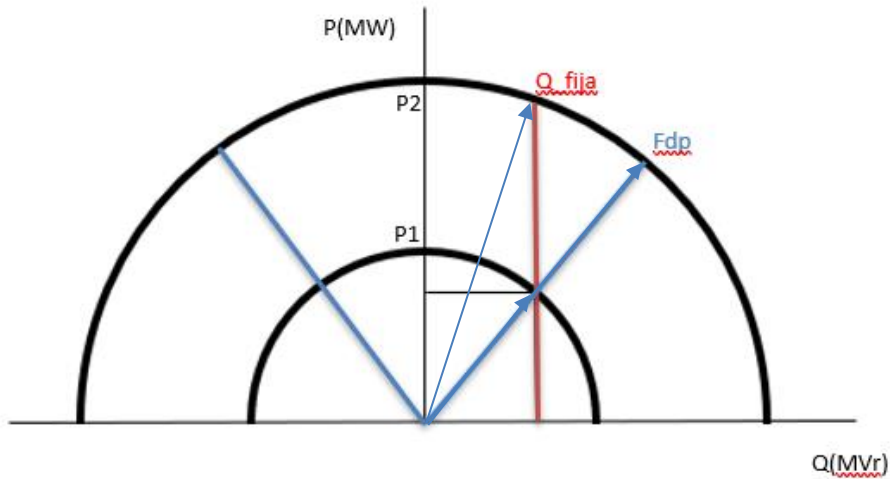


Figura 10. Límites de potencia P-Q.

El programa pedirá los siguientes datos para poder realizar los cálculos requeridos:

- Eficiencia, la cual se encuentra en la fórmula 26 y será un dato del transformador.
- ϵ_{cc} , Caída de tensión relativa en porcentaje.
- Pfe/Pk %, que es la relación en porcentaje entre las pérdidas en el hierro y las pérdidas en el cobre.

$$\mu = \frac{C * V_2 * I_{2n} * \cos \varphi_2}{C * V_2 * I_{2n} * \cos \varphi_2 + C^2 * P_{cc}}$$

Fórmula 26. Eficiencia de un transformador.

$$C = \left(\frac{S}{S_n} \right)^2$$

Fórmula 27. Índice de carga.

Para calcular las pérdidas primero se calculan las pérdidas de potencia activa y las pérdidas de potencia reactiva por separado:

PÉRDIDAS DE LA POTENCIA ACTIVA

Con los datos de P_{fe}/P_k % y la eficiencia podremos obtener las pérdidas en el cobre y las pérdidas en el hierro con las siguientes fórmulas.

$$P_{cu} = \frac{\left[\left(1 - \left(\frac{E_{fi}}{100} \right) \right) * P_n \right]}{1 + \frac{P_{fe}}{100}}$$

Fórmula 28. Pérdidas en el cobre.

$$P_{Fe} = P_{cu} * \left(\frac{P_{fe}}{100} \right)$$

Fórmula 29. Pérdidas en el hierro.

Una vez obtenidas las pérdidas en el cobre y en el hierro, mediante la fórmula 28 se calcularán las pérdidas de potencia activa cada hora.

$$\mathbf{Pérdidas\ potencia\ activa = P_{Fe} + C^2 * P_{cu_n}}$$

Fórmula 30. Pérdidas de potencia activa de un transformador.

La empresa Ibérica Solar expone también el caso de *curtailment*: en el supuesto de que el contrato de esa planta eléctrica fotovoltaica no permita superar una potencia activa acordada, debido limitaciones de regulación de los flujos de potencia en la red eléctrica. En este caso los inversores se limitarán a una potencia máxima la cual no podrán superar.

EJEMPLO DEL CURTAILMENT

En la gráfica de la Figura 8 existe una potencia activa máxima que por ejemplo será de 60MW, aunque haya días en los que la Planta Fotovoltaica genere según el Pvsyst hasta 65MW. Los inversores se usarán para limitar la energía, de tal manera que solo lleguen 60MW, como se muestra en la figura 11.

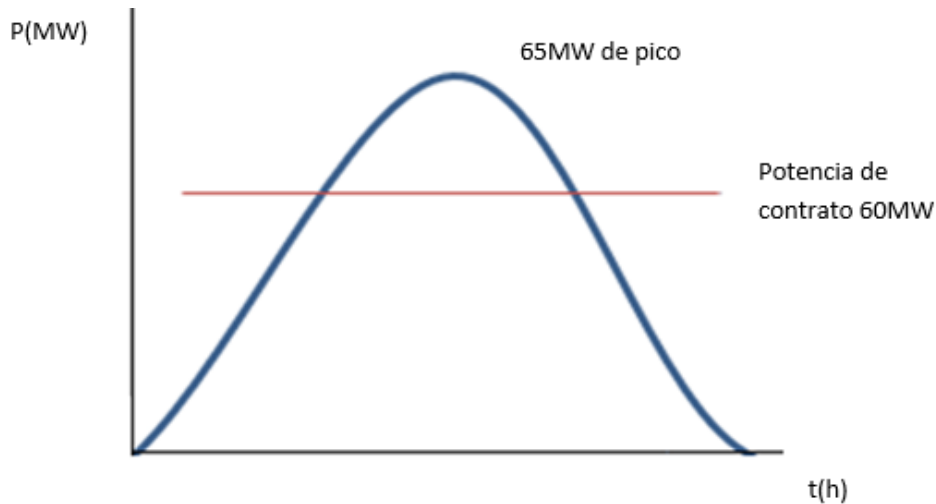


Figura 11. Potencia activa Planta Fotovoltaica.

En el momento en el que se limiten los inversores a la Potencia del contrato, la potencia final después del transformador sería la Potencia de contrato menos la de las pérdidas del transformador.

Suponiendo que fuese 1 MW de pérdidas, se verterían a red 59MW perdiendo 1MW de potencia a entregar a la red.

Por lo que para evitar esto la potencia a la que se limiten los inversores deberá de ser la de contrato más las pérdidas que tenga el transformador, tal y como se ve en la figura 12. Como se ha explicado antes el cálculo del factor de potencia se realizará con esos 61 MW y no con los 60MW.

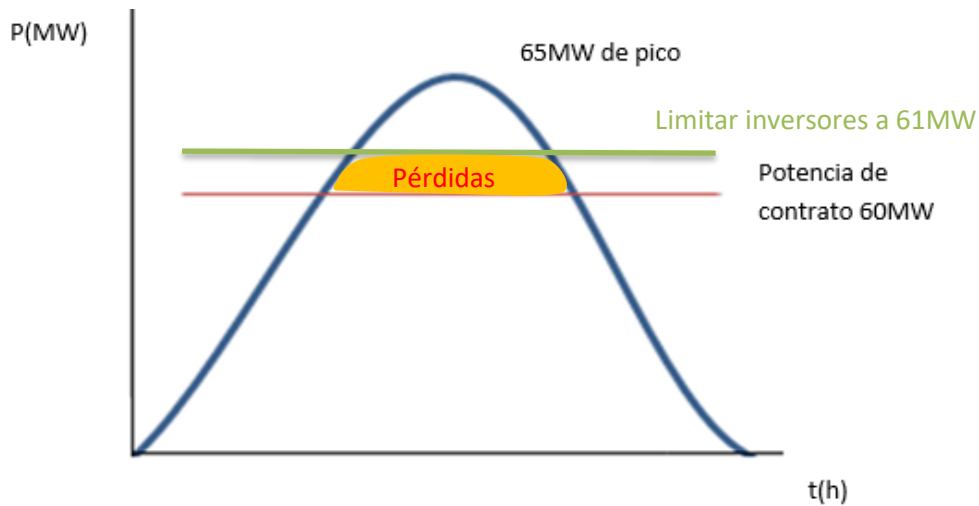


Figura 12. Potencia activa Planta Fotovoltaica con pérdidas.

POTENCIA REACTIVA

El operador del sistema eléctrico exige a las plantas fotovoltaicas contribuir a la inyección de potencia reactiva, de forma que sean capaces las plantas de operar dentro de la curva de límite de potencia P-Q. La planta deberá de tener un factor de potencia para una potencia de generación que abarque desde la potencia activa nula hasta la potencia máxima de la planta fotovoltaica. También, existirá la posibilidad de que se exija una potencia reactiva constante en todo momento. Estos dos casos los podemos observar en la curva P-Q de la figura 10.

El consumo de energía reactiva se calculará con la fórmula 31. En la fórmula 31 se pueden distinguir dos sumandos, el primero de ellos corresponde al consumo de transformador en vacío el cual despreciaré en el programa.

El otro sumando corresponde al consumo del transformador en carga, que estará muy condicionado por el índice de carga en cada instante del transformador.

$$Q_t = \frac{K}{100} * S_n + \frac{X_{cc}}{100} * S_n * C^2$$

Fórmula 31. Consumo de energía reactiva del Transformador.

Para calcular la reactancia de cortocircuito X_{cc} utilizaremos la siguiente fórmula:

$$X_{cc} = \sqrt{Z_{cc}^2 - \left(\frac{P_{cu} * C^2 + P_{Fe} * 100}{S_n} \right)^2}$$

Fórmula 32. Reactancia.

4.OBTENCIÓN DE DATOS A TRAVES DE PVSYST

El Pvsyst es una herramienta utilizada para la gestión y diseño de energía solar para cualquier uso. Esta herramienta permite realizar el diseño una planta fotovoltaica y proporcionar todo tipo de datos de esta.

En este proyecto no se explica cómo utilizar la herramienta para el diseño de una planta fotovoltaica, sin embargo, haré hincapié en la obtención de los datos que interesan una vez diseñada la planta fotovoltaica que se desea analizar.

Para mostrar un ejemplo, se puede comprobar en la figura 13 un proyecto de una planta fotovoltaica en Geneva la cual produce 17379 kwh/año. Para obtener los datos necesarios para el cálculo óptimo del transformador se deberán seguir los siguientes pasos:

1. En la simulación del proyecto se pulsará 'Simulación avanzada', tal y como se muestra en la figura 13.
2. En la 'Simulación avanzada' se pulsará en 'Archivo de salida', tal y como se muestra en la figura 14.
3. Se dejará la pestaña de 'Archivo de salida' tal y como se muestra en la figura 15, para obtener un Excel en el que las dos variables de salida sean la Temperatura ambiente y la Energía inyectada en la red.

Al seguir estos pasos se obtendrá un Excel como el que se puede observar en la figura 16. Este Excel es muy importante que se encuentre de esa manera ya que el programa que se ha creado utilizará las columnas 2 y 3, y con esos datos obtendrá los resultados. Si, por ejemplo, cambiase de lugar la columna de *Tamb* por la de *E_Grid* todo el cálculo realizado por el programa sería erróneo, es por ello que el Excel obtenido del Pvsyst debe ser tal cual se muestra en la figura 16.

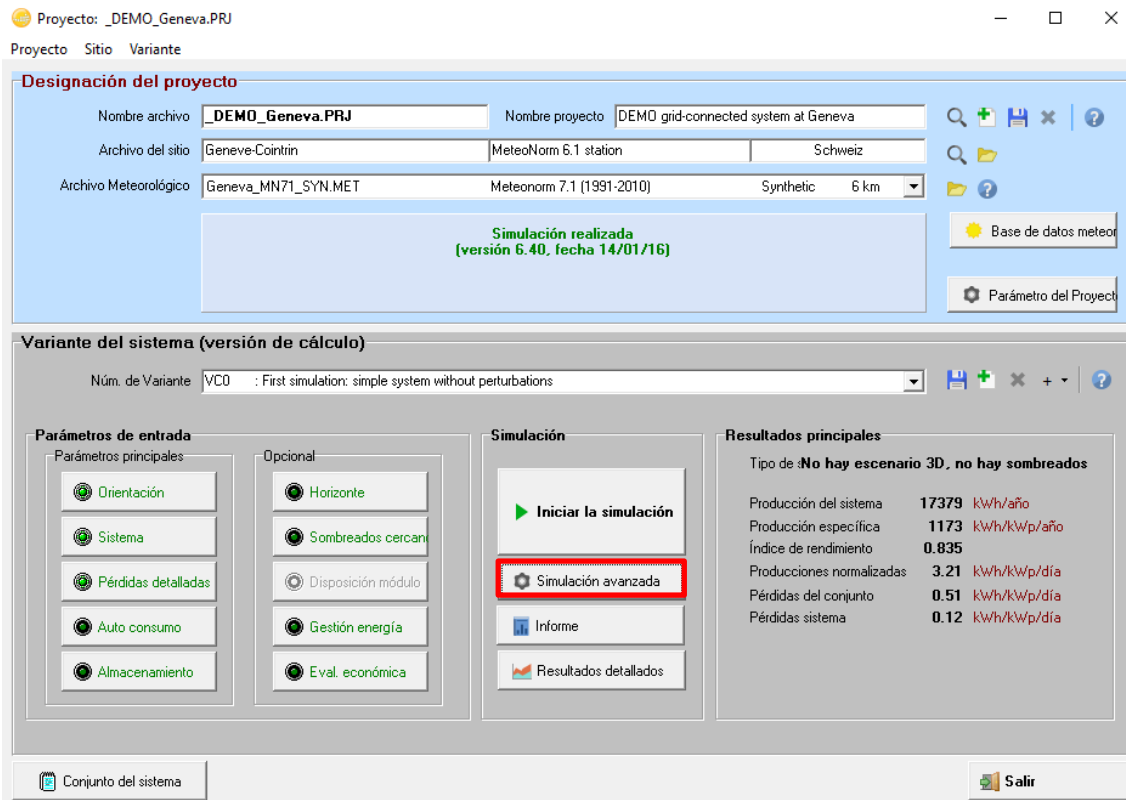


Figura 13. Captura de pantalla PVsyst

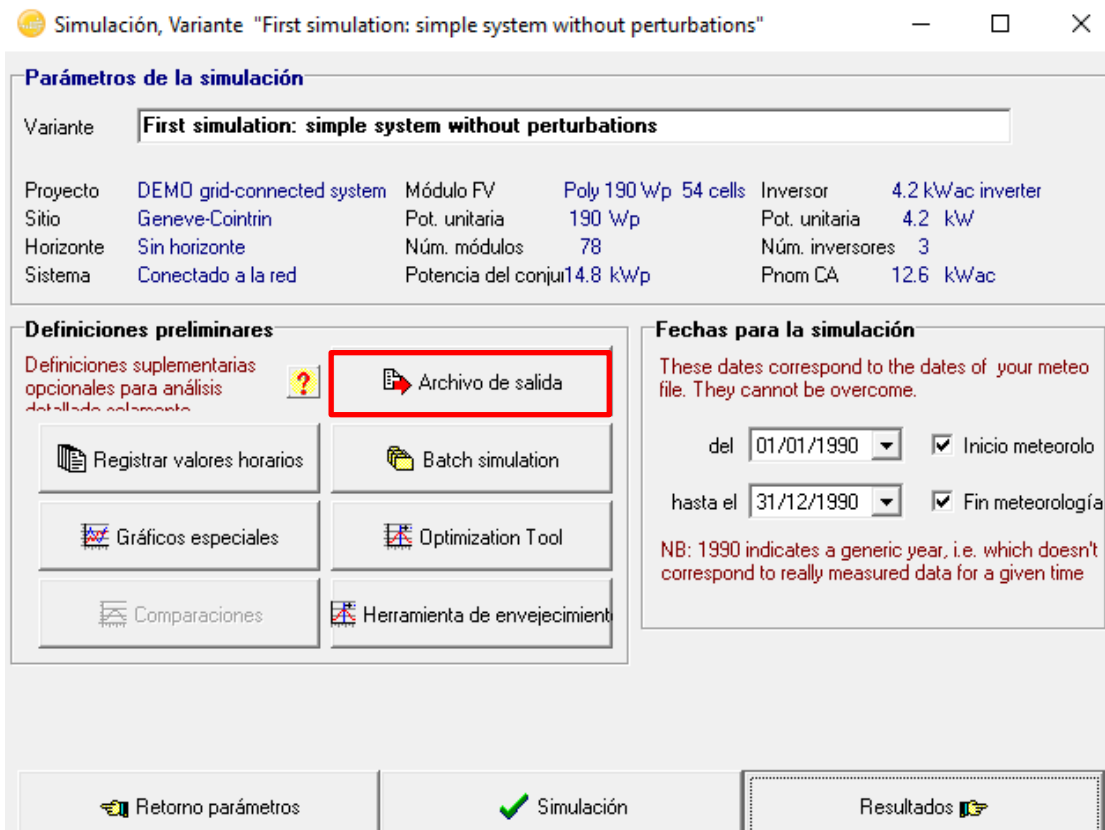


Figura 14. Captura de pantalla PVsyst

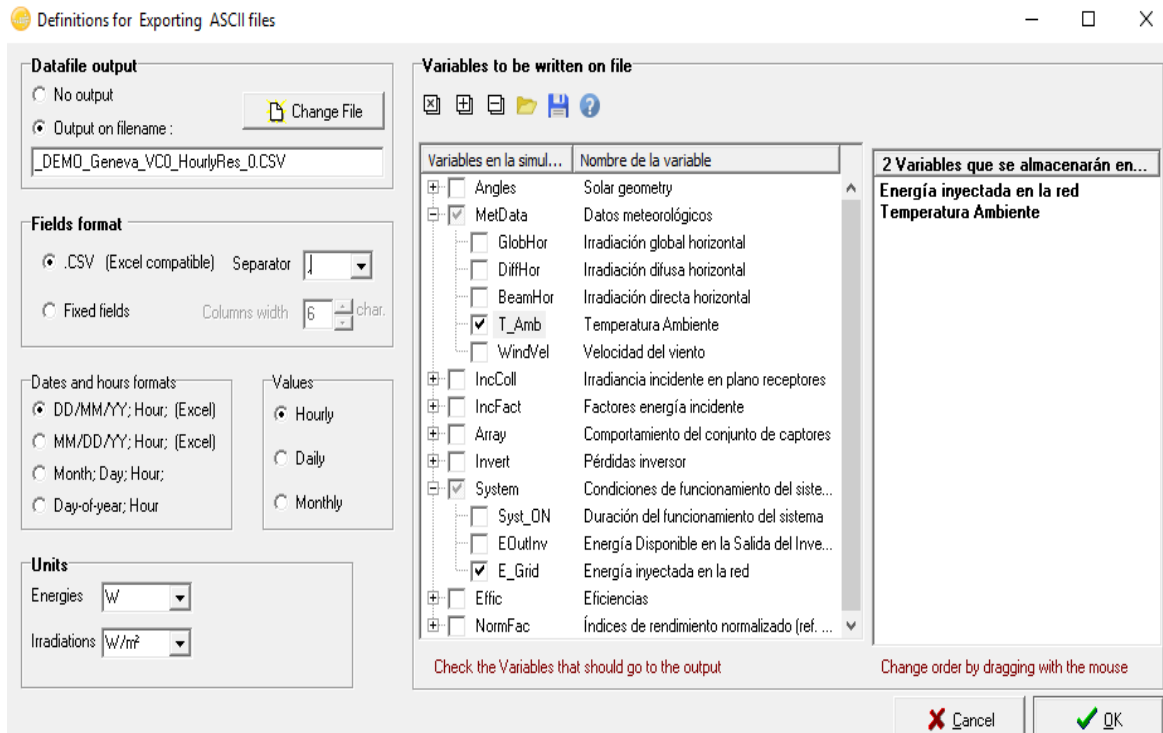


Figura 15. Captura de pantalla PVsyst

| date | T Amb °C | E_Grid W |
|------------------|----------|-----------|
| 01/01/1990 0:00 | 8,9999 | -223596 |
| 01/01/1990 1:00 | 7,9 | -223596 |
| 01/01/1990 2:00 | 6,9997 | -223596 |
| 01/01/1990 3:00 | 6,4002 | -223596 |
| 01/01/1990 4:00 | 5,8998 | -223596 |
| 01/01/1990 5:00 | 5,4001 | -223596 |
| 01/01/1990 6:00 | 6,5998 | -223596 |
| 01/01/1990 7:00 | 10 | 19008603 |
| 01/01/1990 8:00 | 13,1 | 24984299 |
| 01/01/1990 9:00 | 16,9 | 52882391 |
| 01/01/1990 10:00 | 20,8 | 142625173 |
| 01/01/1990 11:00 | 23,3 | 155120493 |
| 01/01/1990 12:00 | 24,4 | 161447667 |
| 01/01/1990 13:00 | 24,7 | 165896762 |
| 01/01/1990 14:00 | 24,5 | 161757108 |
| 01/01/1990 15:00 | 22,6 | 146600401 |
| 01/01/1990 16:00 | 19,5 | 69368578 |
| 01/01/1990 17:00 | 17,6 | -223161 |
| 01/01/1990 18:00 | 16,3 | -223161 |
| 01/01/1990 19:00 | 15,3 | -223161 |

Figura 16. Excel obtenido de PVsyst.

5.SOFTWARE

En este apartado se desarrollará y explicará cómo está diseñado el programa y sus diferentes partes.

El programa está dividido en:

1. La lectura de los datos obtenidos por el Pvsyst y su tratamiento para poder ser usados.
2. El cálculo óptimo del transformador para la normativa europea y para la normativa americana.
3. La obtención de gráficas para la interpretación de los datos obtenidos
4. La interfaz del programa.

Todas y cada una de las partes del programa se han realizado con la herramienta Matlab y la creación de la interfaz se ha desarrollado con la herramienta proporcionada por Matlab, Guide.

5.1 PROGRAMACIÓN DE LECTURA DE DATOS

El tratamiento de los datos obtenidos por el Excel se puede observar en el anexo 1, desde la línea 606 hasta la línea 648.

En este intervalo el programa obtiene los datos del Excel mediante la función de Matlab *xlsread*.

Como ya se ha explicado en el apartado 2 de normativa, se necesitará la potencia y la temperatura de la planta durante cada hora. El Excel proporciona la temperatura y la potencia entregada a red en el instante de cada hora, teniendo de esta manera 24 datos cada día. Sin embargo, se necesitan 23 datos obteniéndose de esta manera 23 intervalos que serían: intervalo día 1 hora 0-1, intervalo día 1 1-2, intervalo día 1 hora 2-3,

Así se obtendrán los 23 intervalos y se mostrará una función a *steps* tal y como se mencionaba en el apartado 2.

En estas líneas de código se observa cómo se obtiene la temperatura y la potencia activa en intervalos:

1. Para la temperatura se calcula la media entre la hora 1 y la hora 2 y así sucesivamente
2. Para la potencia se resta la potencia de la hora 2 y la potencia de la hora 1, para más tarde, en el siguiente bucle, utilizar un sumatorio de cada intervalo para obtener la función step.
3. Por último, el bucle final mostrado se utiliza para limitar la potencia activa, tal y como se explica en el apartado 3 del proyecto.

Al final del tratamiento de datos la variable 'T' será un vector que poseerá la temperatura de los intervalos y la variable *X_Var* será un vector que poseerá la potencia activa de cada intervalo en el lado de baja del transformador.

5.2 PROGRAMACIÓN NORMATIVA IEC

La programación del cálculo de temperatura siguiendo la normativa IEC se encuentra desde la línea 819 del código del anexo 1 hasta la línea 894.

El programa usa funciones para calcular todas las temperaturas y usa unas funciones u otras dependiendo de si aumenta o disminuye el índice de carga.

Estas funciones que contienen las fórmulas descritas en la normativa europea y están explicadas en el apartado 2 de este proyecto y descritas en el anexo 2.

Como se puede comprobar en la figura 17, la cual muestra el flujograma del programa, cada vez que se calcula una temperatura se comprueba si esta supera la temperatura máxima fijada por la norma. Si esta temperatura calculada supera la de norma una variable interna propia del programa 'err1' se tornará a 1.

El mismo proceso se sigue con la pérdida de vida útil máxima, ya que si esta es superada la variable interna 'err2' se tornará a 1. Por lo que todos los cálculos de temperatura y pérdida de vida útil se encuentran dentro de dos bucles *while* anidados los cuales repetirán una y otra vez los cálculos añadiendo la tolerancia a la potencia, hasta que se cumplan esas dos condiciones.

Luego todos los datos obtenidos serán volcados a la matriz 'Resultados' para el posterior análisis de estos.

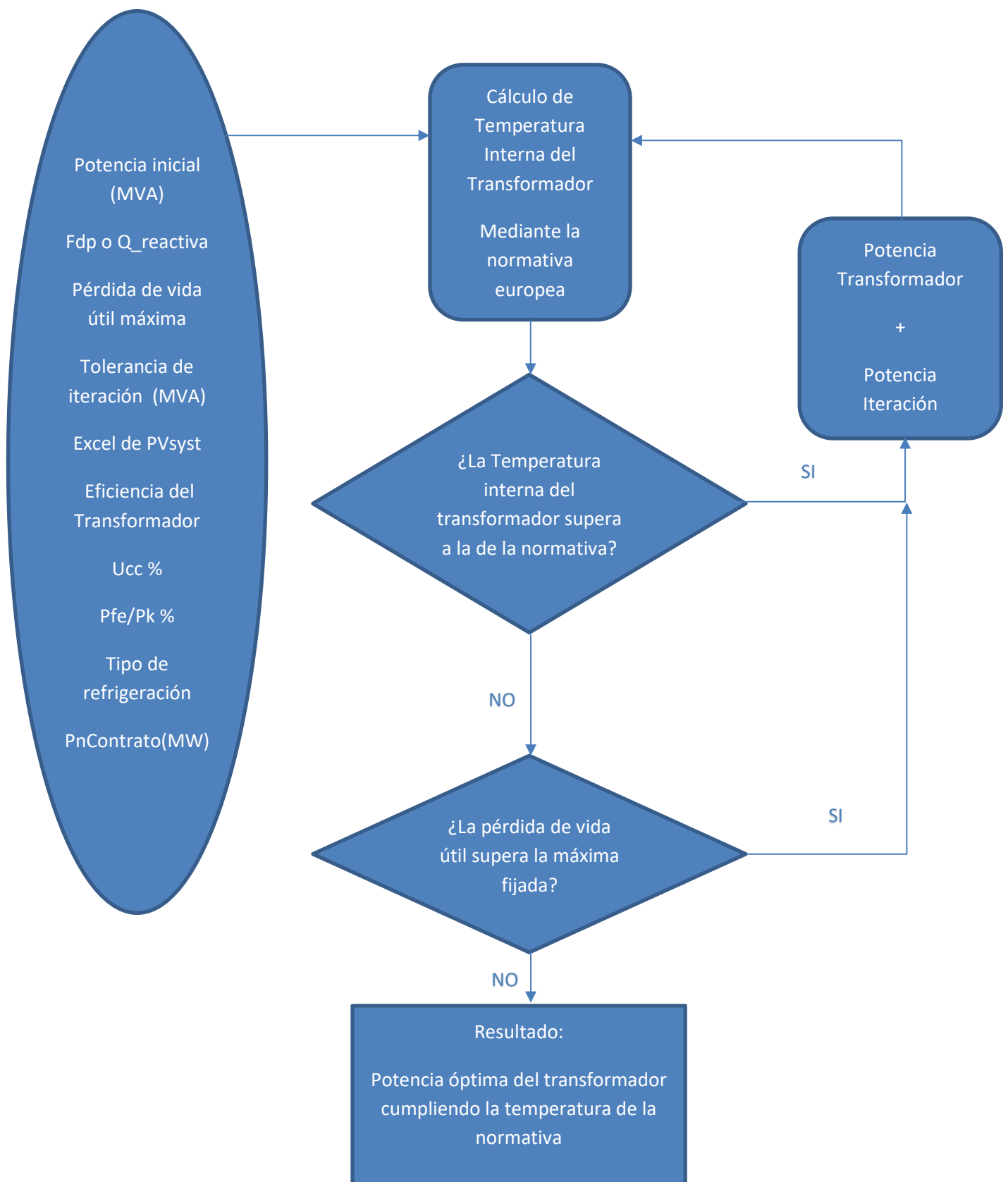


Figura 17. Flujograma de Programa con normativa IEC

5.3 PROGRAMACIÓN NORMATIVA IEEE

La programación del cálculo de temperatura siguiendo la normativa IEEE se encuentra desde la línea 895 del código del anexo 1 hasta la línea 1028. El programa usa funciones para calcular todas las temperaturas y usa unas funciones u otras dependiendo de si aumenta o disminuye el índice de carga. Estas funciones que contienen las fórmulas descritas en la normativa americana y están explicadas en el apartado 2 de este proyecto y descritas en el anexo 2. El resto es igual que la otra normativa, el único cambio es la forma de calcular las temperaturas que cambia tal y como explico en el apartado 2. Por lo demás posee también dos bucles anidados y el flujograma de la figura 18 es exactamente igual.

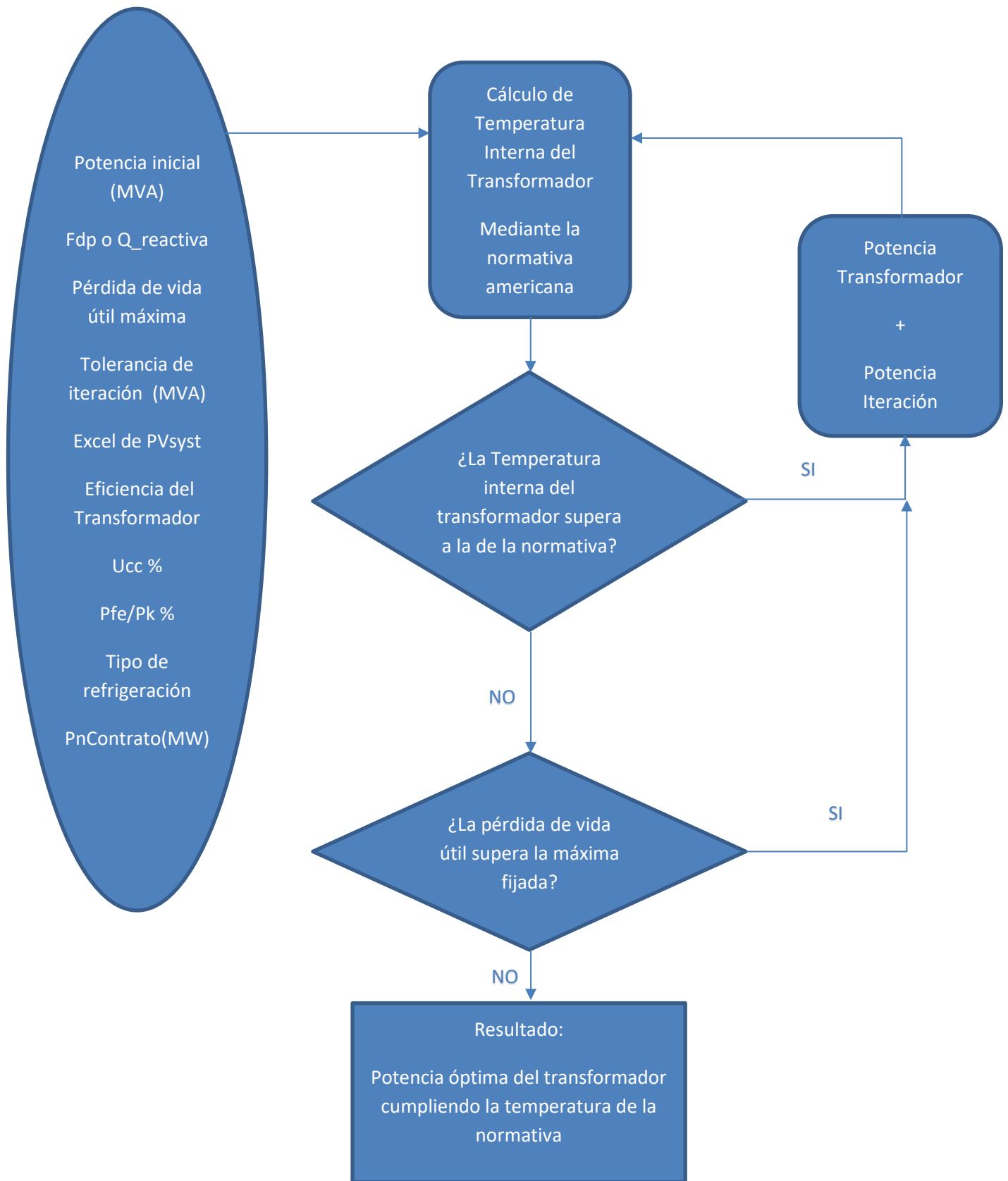


Figura 18. Flujograma de Programa con normativa IEEE

5.4 PROGRAMACIÓN ANÁLISIS DE DATOS

El análisis de datos es una parte muy importante del programa, con ella podemos verificar los resultados obtenidos o simplemente observar las modificaciones que se deseen.

Para ello el desarrollo de esta parte se basó en la realización de 3 gráficas:

1. La primera gráfica muestra el día más crítico del año en el cual el transformador alcanza la mayor temperatura en el interior, este día es el más importante ya que es el que condiciona la dimensión del transformador, y es fundamental que se puedan analizar los datos y cálculos obtenidos por el programa ese día.
2. Las otras dos gráficas serán personalizables, y el usuario podrá elegir qué día desea analizar los datos. Estas gráficas están diseñadas con el objetivo de comparar dos días del año, y seguir el proceso de aumento y disminución de temperatura de los transformadores.

Esta manera de analizar los datos permite cerciorarse de que el programa funciona coherentemente en función de la sobrecarga que posea el transformador a lo largo del año.

El análisis de datos se puede observar en el anexo 1 desde la línea 1030 del código hasta la línea 1117 del código.

Las gráficas representan el día específico que se quiera, mostrando los datos de la matriz "Resultados". La matriz "Resultados" es una matriz donde se almacenan los siguientes datos: Intervalos de tiempo, potencia aparente, temperatura de *hot-spot*, temperatura de *top-oil*, días de envejecimiento, potencia activa.

La gráfica del día crítico evalúa en la matriz "Resultados", cual es el día con un *hot-spot* más alto y muestra en la gráfica la potencia aparente de ese día, y el *hot-spot* y el *top-oil* calculados ese día.

Las gráficas de los días que el usuario desee evaluar se obtendrán utilizando los datos obtenidos de la matriz "Resultados" en el día que desee el usuario, mostrando en la gráfica la potencia aparente ese día, y el *hot-spot* y el *top-oil* calculados ese día.

La gráfica del día crítico podemos observarla en la figura 19 y las gráficas de los días elegidos por el usuario las podemos ver en las figuras 20 y 21.

En las gráficas el eje x es la hora de la planta fotovoltaica a lo largo del año, y el eje y son centígrados y potencia aparente dividida entre 1000.

Se han juntado esas dos magnitudes en el eje y para poder observar a la vez tanto la potencia como el aumento de temperatura a medida que aumente la potencia.

Hay una opción en el programa que permite cambiar la magnitud de la potencia para que en las figuras 19-21 la potencia, la cual está dividida entre 100000 para poderse analizar correctamente, pueda apreciarse en concordancia con la temperatura. Si no existiera este ajuste dependiendo del tamaño de la planta fotovoltaica en las gráficas solo se apreciaría una de las dos magnitudes, tal y como se observa en la figura 22.

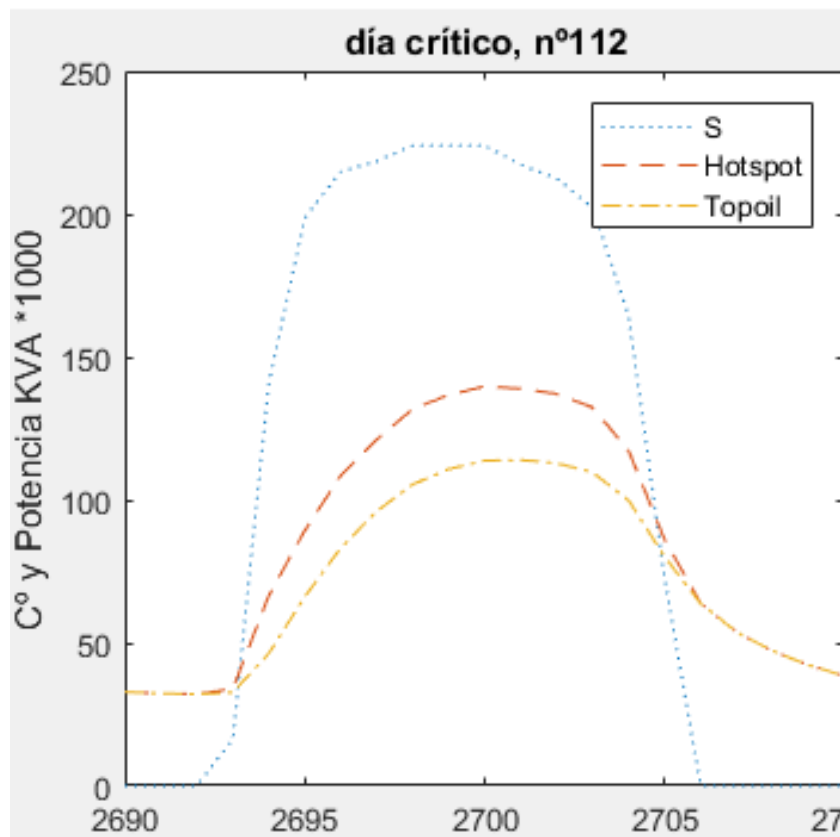


Figura 19. Gráficas potencia-temperatura frente a intervalo de tiempo

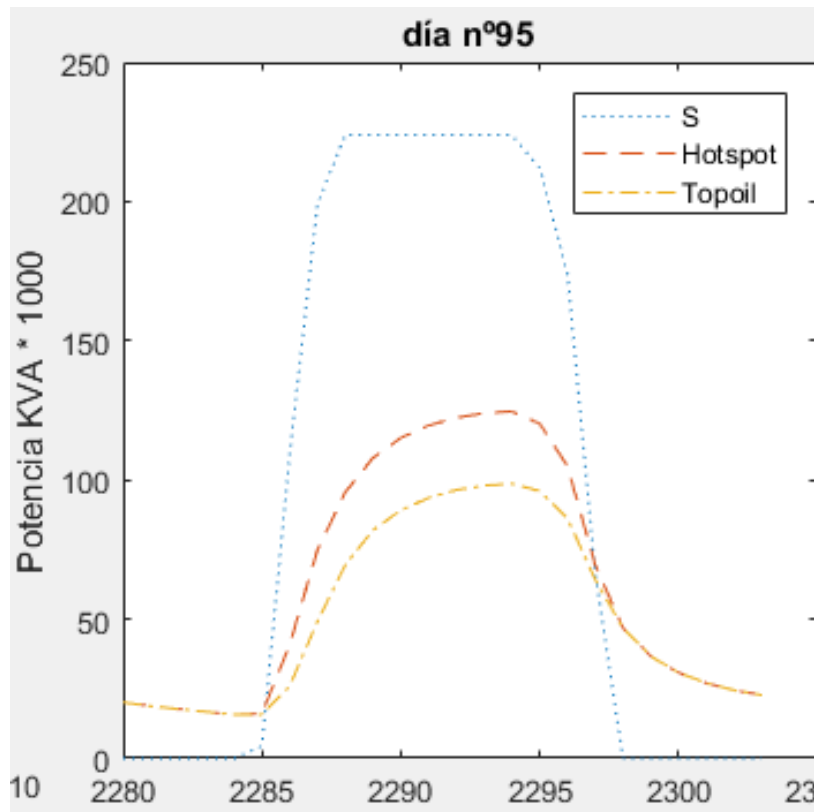


Figura 20. Gráficas potencia-temperatura frente a intervalo de tiempo

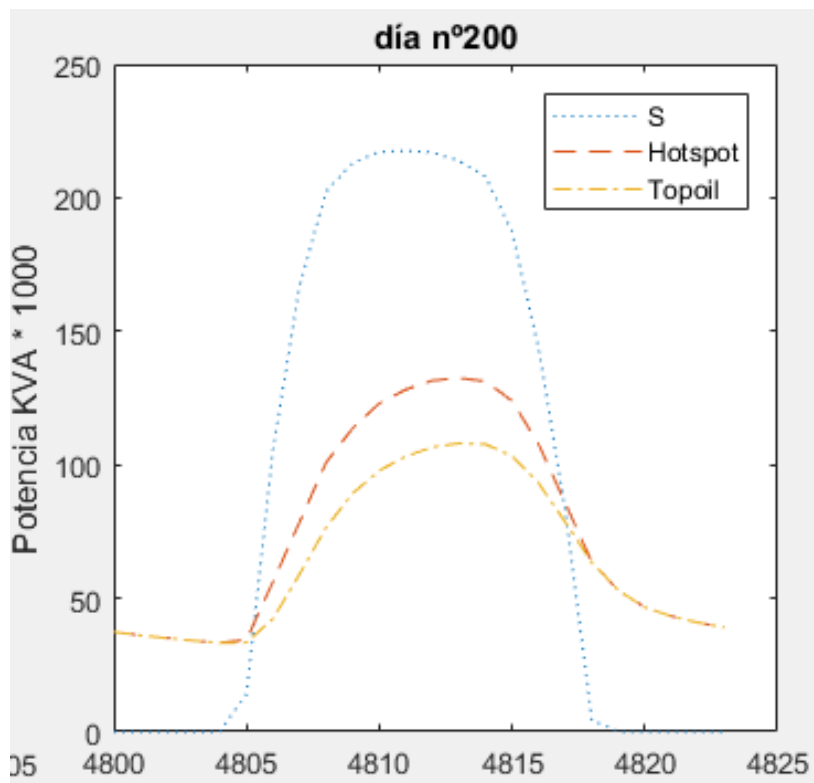


Figura 21. Gráficas potencia-temperatura frente a intervalo de tiempo

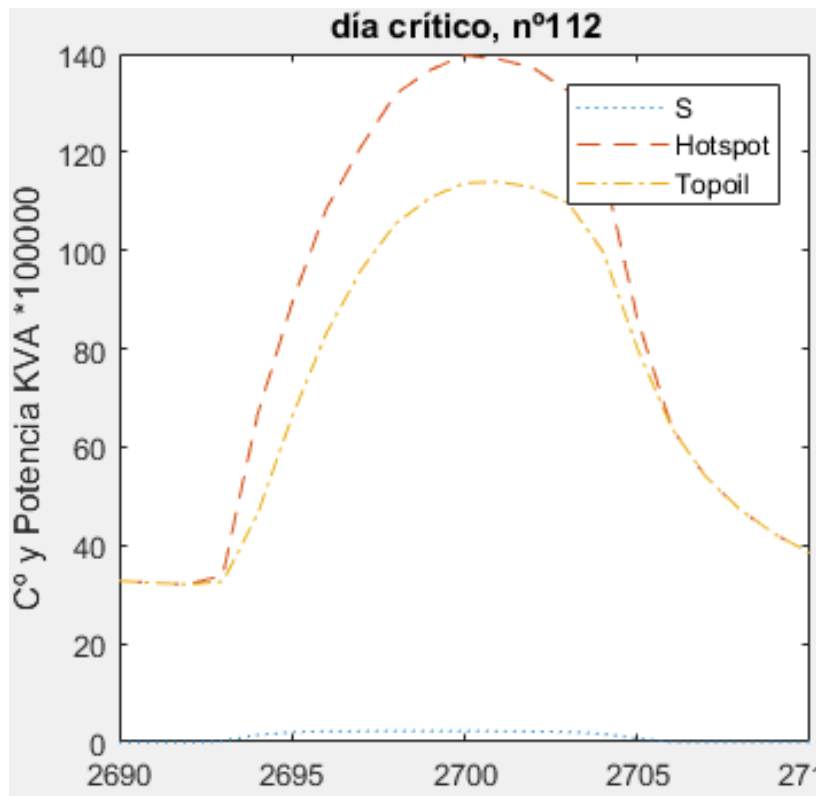


Figura 22. Gráficas potencia-temperatura frente a intervalo de tiempo

5.5 INTERFAZ DEL PROGRAMA

La interfaz gráfica del programa está hecha con la herramienta GUIDE proporcionada por Matlab. La última versión de la interfaz se observa en la Figura 21.

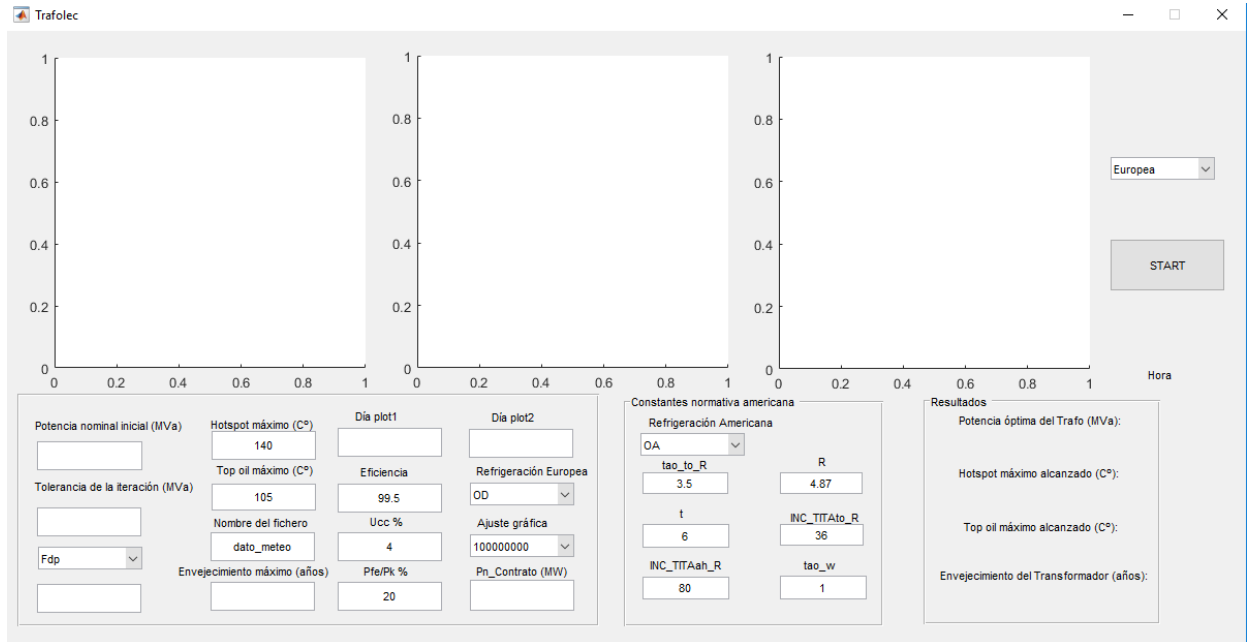


Figura 23. Interfaz del programa.1

La Interfaz gráfica la explicaré por partes tal y como se muestra en la figura 24, siendo esta figura un ejemplo de una planta fotovoltaica.

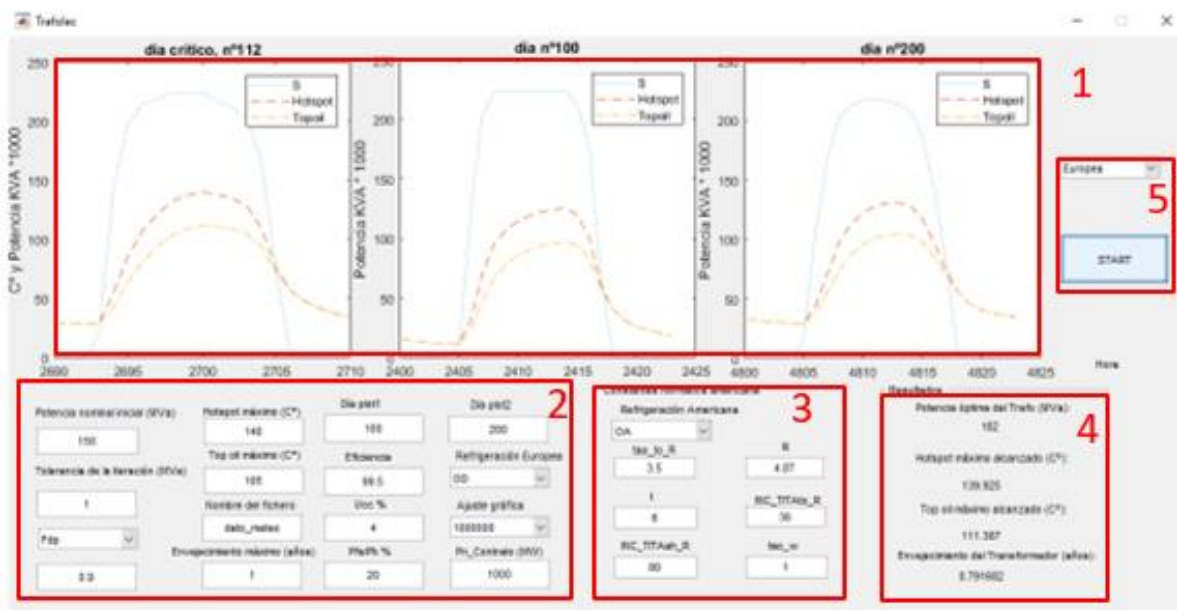


Figura 24. Interfaz del programa.2

1. Análisis de datos:

La parte de análisis de datos del programa está explicada en el apartado 5.4 del proyecto.

Se divide en 3 gráficas cuyo eje x es la hora del año a la que el Pvsyst ha obtenido ese dato, mientras que el eje y es tanto potencia en KVA multiplicado por el ajuste deseado y centígrados.

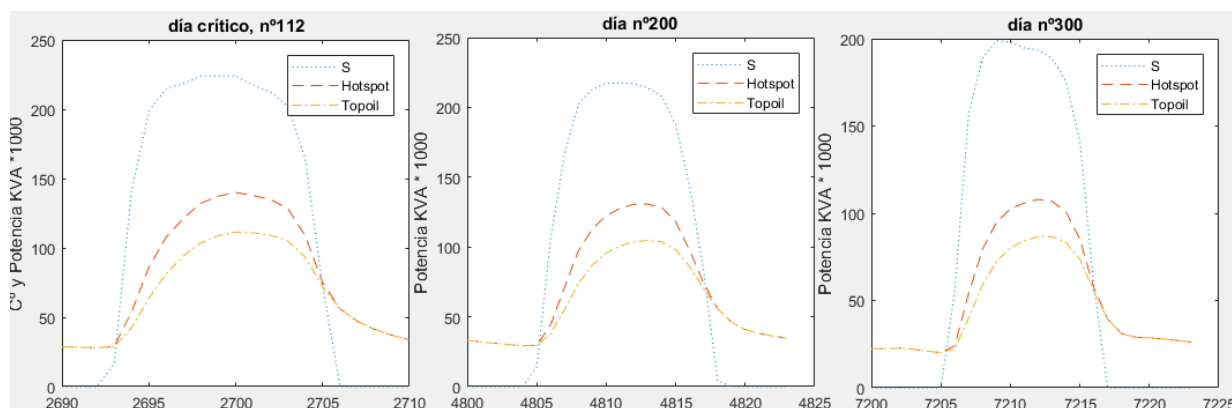


Figura 25. Interfaz del programa.3

2. Parte común para el cálculo.

La parte común para que el programa pueda realizar los cálculos es la vista en la figura 26.

A continuación, se explicarán cada uno de los rellenables de izquierda a derecha y de arriba hacia abajo:

- Potencia nominal inicial (MVA)

Potencia aparente nominal inicial con la que el programa empezará a calcular, cabe destacar que esta potencia deberá siempre ser menor a la potencia que posea la planta fotovoltaica.

- Hot-spot máximo (°C)

Hot-spot máximo que el transformador alcanzará en el resultado final. La norma establece que debe de ser de 140°C, sin embargo, el programa da la opción de cambiar este valor, aunque de forma predefinida sea de 140°C.

- Día plot1

Número del día a observar en la gráfica 2, teniendo en cuenta que este número deberá estar entre 1 y 365 ya que son los días que analiza el programa.

- Día plot2

Número del día a observar en la gráfica 3, teniendo en cuenta que este número deberá estar entre 1 y 365 ya que son los días que analiza el programa.

- Tolerancia de la iteración (MVA)

Potencia de iteración que el programa sumará a la potencia del transformador por cada iteración que el programa realice. Un número de tolerancia grande hará el programa más rápido debido a menos iteraciones, sin embargo, disminuirá su precisión. Mientras que un programa con un número de tolerancia pequeño provocará que el programa realice más iteraciones y a su vez sea más lento, pero esta vez la precisión aumentará.

- Top-oil máximo (°C)

Top-oil máximo que el transformador alcanzará en el resultado final. La norma establece que debe de ser de 105°C, sin embargo, el programa da la opción de cambiar este valor, aunque de forma predefinida sea de 105°C.

- Eficiencia

Eficiencia que posee el transformador a calcular, esta se podrá obtener de su placa de características.

- Refrigeración Europea

Coefficiente para el cálculo de las temperaturas del transformador, del cual dependerá del tipo de refrigeración que posea el transformador.

Esta refrigeración se podrá elegir en el desplegable el cual tiene las siguientes opciones: OD, OF, ONAN, ONAF.

- Nombre del Fichero

Nombre del fichero Excel explicado en el apartado 4 del proyecto.

- %Ucc

Porcentaje de tensión relativa del transformador, se puede obtener de la placa de características del mismo.

- Ajuste gráfica

Desplegable cuya utilidad es ajustar la gráfica dependiendo del tamaño de la planta fotovoltaica, tal y como se ha mencionado en el apartado 5.4.

- Fdp

Factor de potencia demandado o seleccionado para entregar a la red. Si se desea que sea una potencia reactiva fija habría que cambiar el desplegable Fdp y seleccionar Q(MVAr).

- Envejecimiento máximo(años)

Envejecimiento máximo en años que el transformador alcanzará en el resultado final.

- Pfe/Pk %

Relación en porcentaje entre la Pérdidas en el hierro y en el cobre.

- Pn_Contrato(MW)

Máxima potencia activa que se debe entregar en el punto de interconexión de la planta.

| | | | |
|----------------------------------|---|-----------------------------------|--------------------------------------|
| Potencia nominal inicial (MVA) | Hotspot máximo (C°) | Día plot1 | Día plot2 |
| <input type="text" value="100"/> | <input type="text" value="140"/> | <input type="text" value="200"/> | <input type="text" value="300"/> |
| Tolerancia de la iteración (MVA) | Top oil máximo (C°) | Eficiencia | Refrigeración Europea |
| <input type="text" value="1"/> | <input type="text" value="105"/> | <input type="text" value="99.5"/> | <input type="text" value="OD"/> |
| Fdp | Nombre del fichero | Ucc % | Ajuste gráfica |
| <input type="text" value="0.9"/> | <input type="text" value="dato_meteo"/> | <input type="text" value="4"/> | <input type="text" value="1000000"/> |
| | Envejecimiento máximo (años) | Pfe/Pk % | Pn_Contrato (MW) |
| | <input type="text" value="10"/> | <input type="text" value="20"/> | <input type="text" value="3000"/> |

Figura 26. Interfaz del programa.4

3. Zona de la norma americana.

La zona de la normativa americana de para que el programa pueda realizar los cálculos es la vista en la figura 27.

A continuación, explicaré cada uno de los rellenables

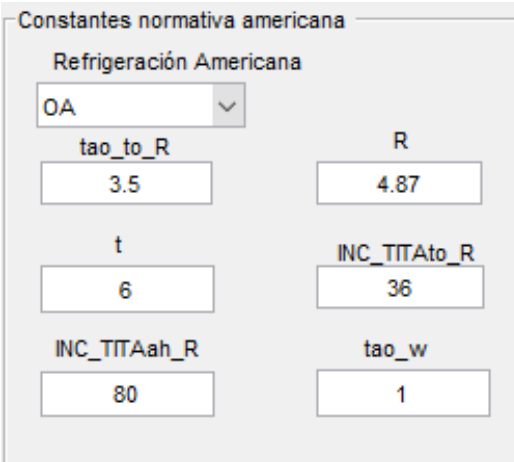
- Refrigeración Americana

Coeficiente para el cálculo de las temperaturas del transformador, del cual dependerá del tipo de refrigeración que posea el transformador.

Esta refrigeración se podrá elegir en el desplegable el cual tiene las siguientes opciones: OA, FA, Non-directed FOA or FOW, Directed FOA or FOW.

- Constantes Americanas

El resto son constantes explicadas en la IEC las cuales están predefinidas por el programa. Sin embargo, se da la posibilidad de cambiar estas constantes.



| Constantes normativa americana | |
|--------------------------------|--------------|
| Refrigeración Americana | |
| OA | |
| tao_to_R | R |
| 3.5 | 4.87 |
| t | INC_TITAto_R |
| 6 | 36 |
| INC_TITAAh_R | tao_w |
| 80 | 1 |

Figura 27. Interfaz del programa.5

4. Resultados

Los resultados se encuentran en el panel 'Resultados' que se observa en la figura 28. Los resultados mostrarán:

- Potencia óptima del transformador (MVA)

La potencia final del transformador calculada cumpliendo con la normativa indicada.

- *Hot-spot* máximo alcanzado (°C)

El *hot-spot* máximo alcanzado con la potencia final del transformador, éste nunca superará al *hot-spot* máximo fijado.

- Top-oil máximo alcanzado(°C)

El *top-oil* máximo alcanzado con la potencia final del transformador, éste nunca superará al *hot-spot* máximo fijado.

- Envejecimiento del transformador (años)

La vida perdida en años después de un año de funcionamiento.

| Resultados | |
|--|----------|
| Potencia óptima del Trafo (MVA): | 182 |
| Hotspot máximo alcanzado (C°): | 139.925 |
| Top oil máximo alcanzado (C°): | 111.387 |
| Envejecimiento del Transformador (años): | 0.791682 |

Figura 28. Interfaz del programa.6

5. Inicio y elección de normativa

En esta parte se encontrará un botón de *start* el cual ordenará al programa empezar los cálculos y un desplegable para elegir con que norma realizar los mismos.

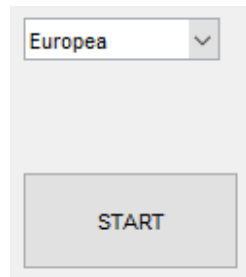
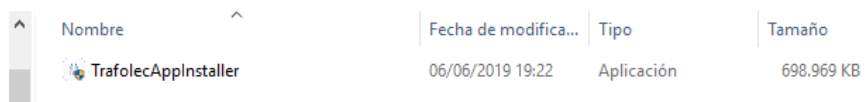


Figura 29. Interfaz del programa.7

5.5 INSTALACIÓN DEL PROGRAMA

El programa creado en Matlab se ha desarrollado en una aplicación con la herramienta de Matlab *Application compiler*.

Con el desarrollo de esta aplicación no es necesario utilizar el Matlab para usar el programa creado. Usando el instalador *TrafolecAppInstaller*, instalaremos el programa en el ordenador y se seguirán los pasos que se muestran en las siguientes figuras:



| Nombre | Fecha de modifica... | Tipo | Tamaño |
|----------------------|----------------------|------------|------------|
| TrafolecAppInstaller | 06/06/2019 19:22 | Aplicación | 698.969 KB |

Figura 30. Instalación del programa. Paso1

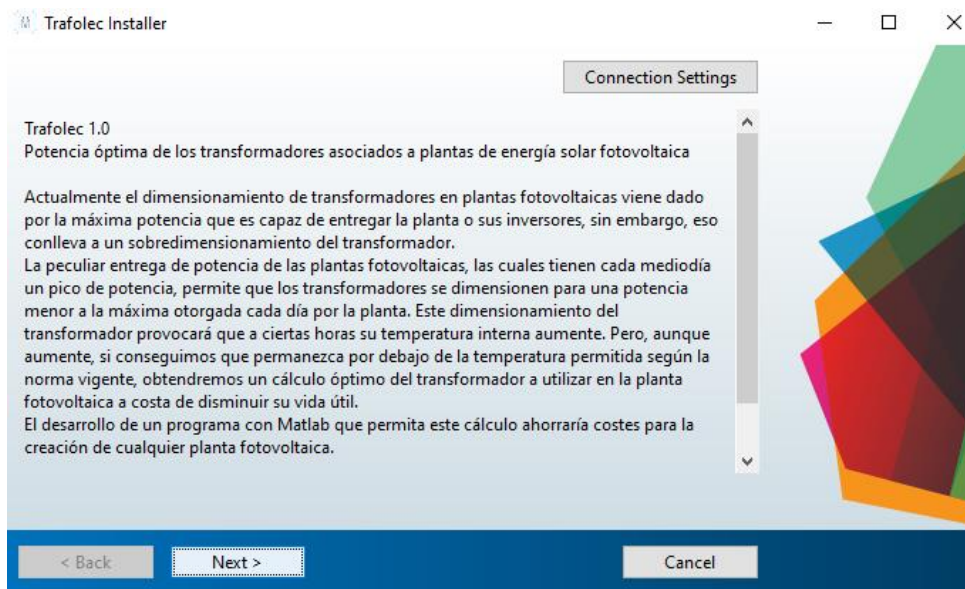


Figura 31. Instalación del programa. Paso2

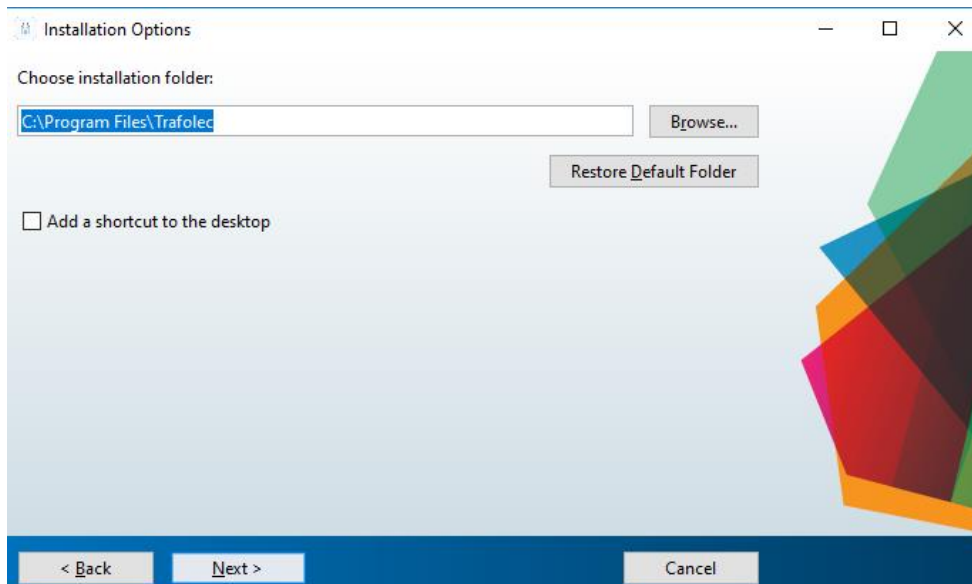


Figura 32. Instalación del programa. Paso3

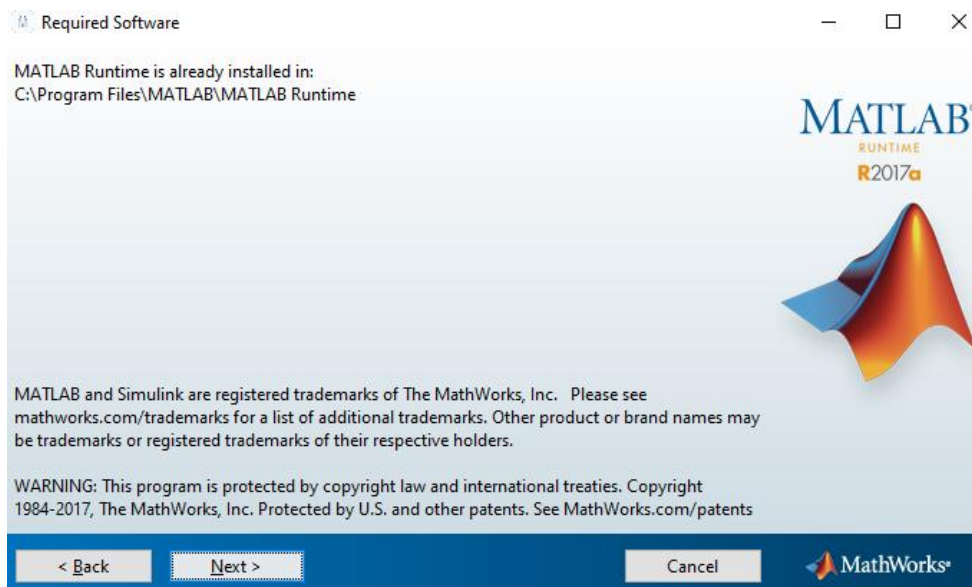


Figura 33. Instalación del programa. Paso4

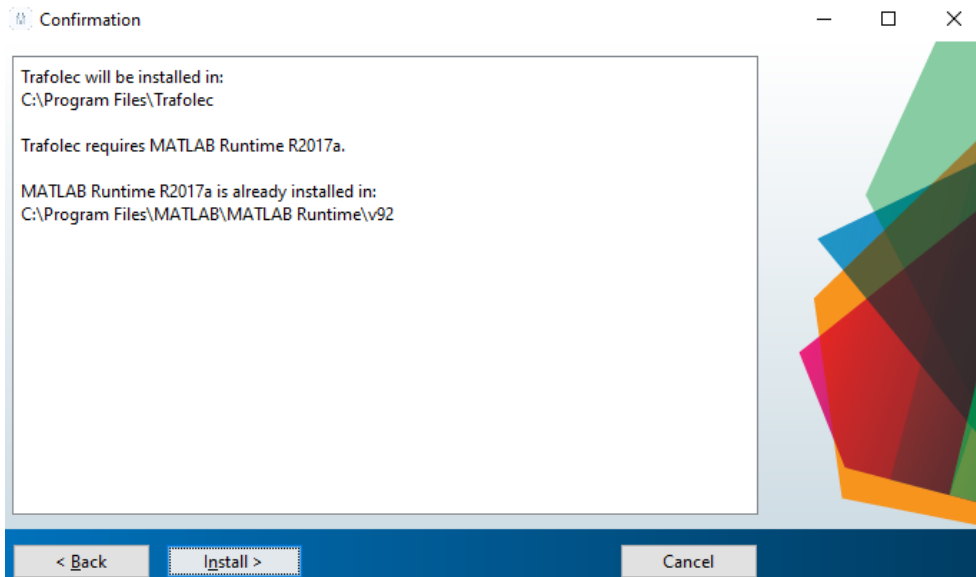


Figura 34. Instalación del programa. Paso5

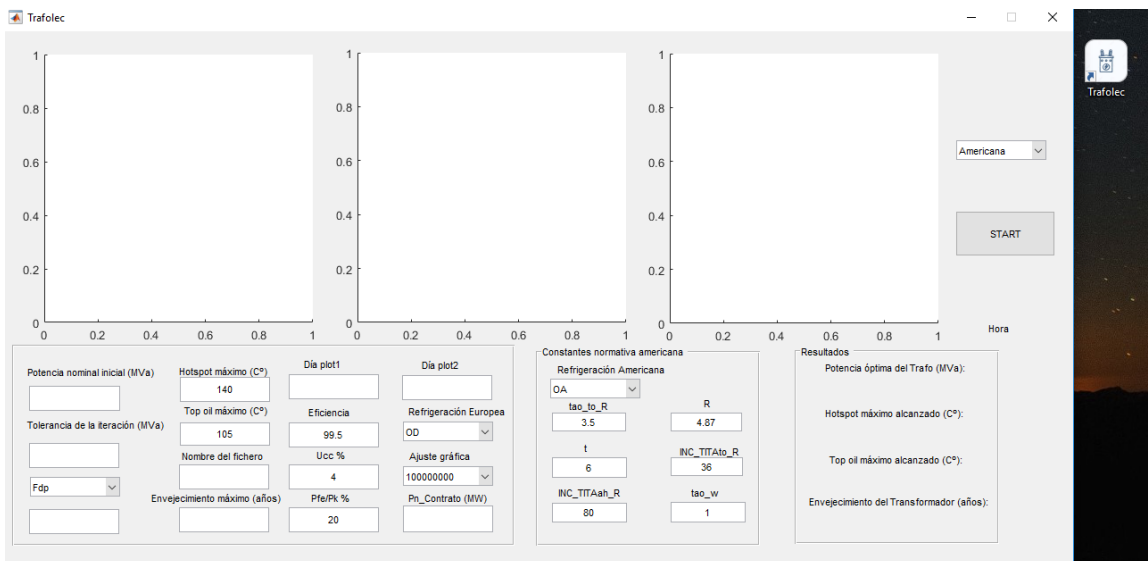


Figura 35. Instalación del programa. Paso6

Al terminar la instalación el icono del programa será el de la figura 35, y para su uso solo habrá que introducir el Excel de la planta a analizar dentro de la carpeta de Trafolec en archivos de programa.

6. ESTUDIO DE VARIAS PLANTAS

En este apartado probaremos las siguientes plantas fotovoltaicas con el programa creado.

Las plantas fotovoltaicas a estudiar serán las siguientes:

- La Solanilla 50 MW

En la figura 36 el programa utiliza el criterio europeo, alcanzando el punto crítico el día 179 el cual obtiene un *hot-spot* de 139.88 C°, cumpliendo así la norma europea.

El dimensionamiento final del transformador es de 36 MVA reduciendo bastante el tamaño del transformador si se hubiera elegido uno de 50 MVA.

Finalmente, el envejecimiento del transformador ha sido de un poco más de medio año, a lo largo de todo un año de funcionamiento.

En la figura 37 se obtienen otros resultados, ya que limitamos la potencia de los inversores a 30MW.

Y por último la figura 38 los cálculos se realizan siguiendo la norma americana, variando la potencia final obtenida y el envejecimiento, siendo esta norma un poco más estricta.

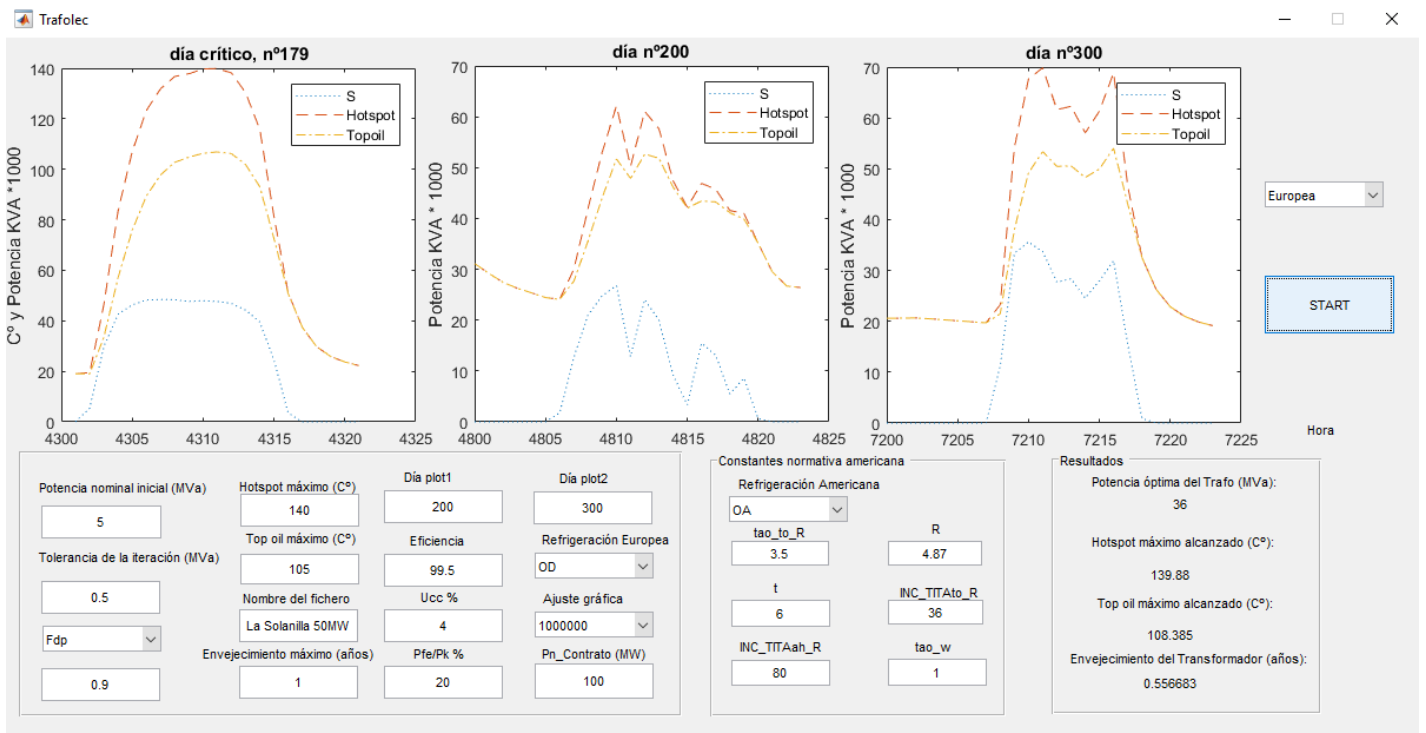


Figura 36. La Solanilla 50MW.1

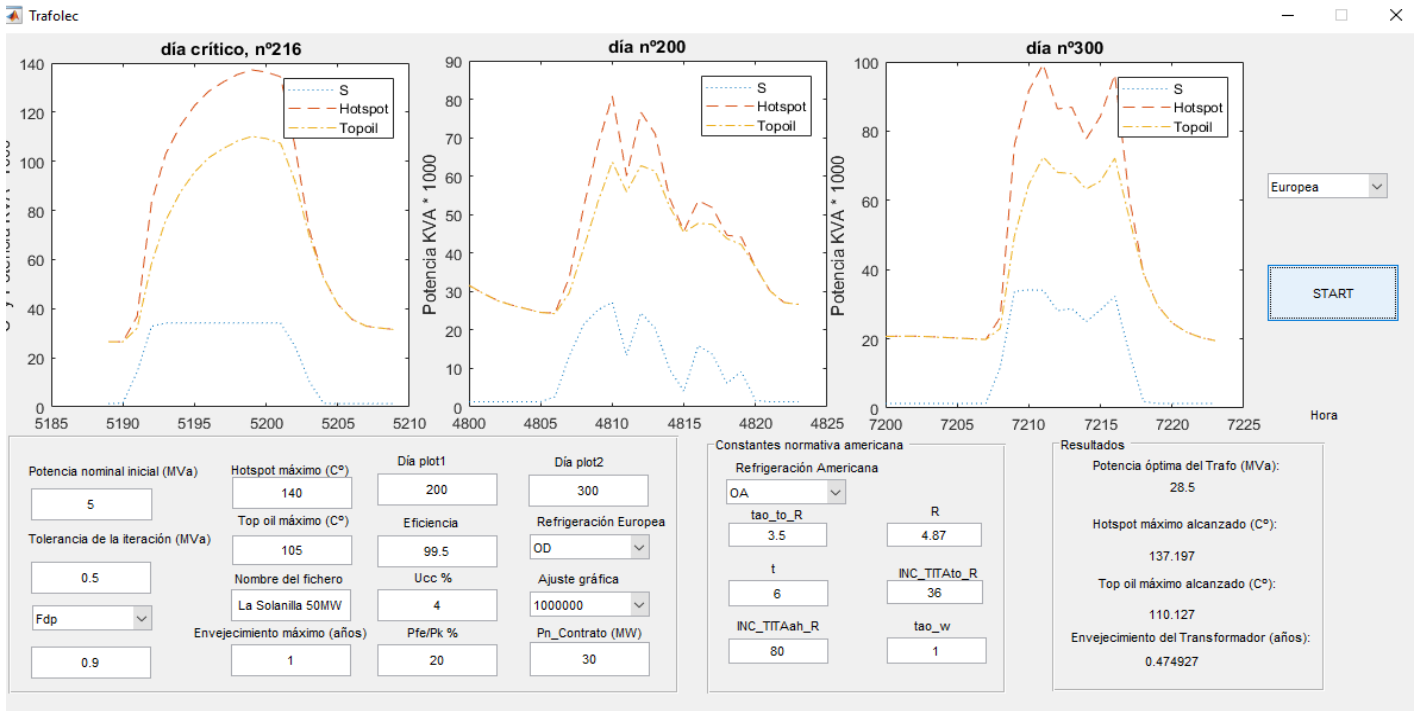


Figura 37. La Solanilla 50MW.2

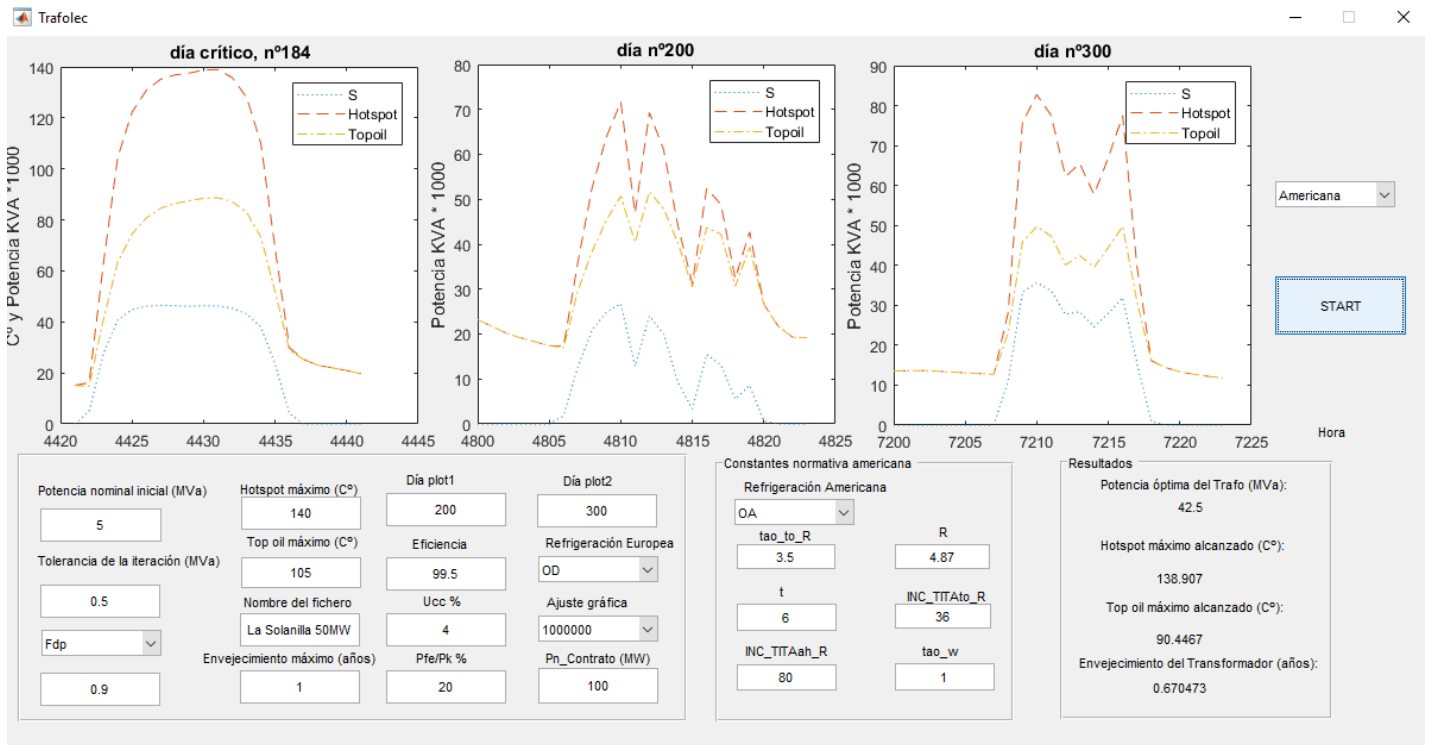


Figura 38. La Solanilla 50MW.

- El Pedregal 9.9 MW

Al igual que el anterior ejemplo se muestran tres casos, el caso de la norma europea, el caso de la norma europea con los inversores limitados a una potencia activa determinada y el caso de la norma americana capando también los inversores.

El análisis sería parecido en estos casos también, se puede comprobar como al capar los inversores y sobrecargar menos el transformador la temperatura disminuye considerablemente.

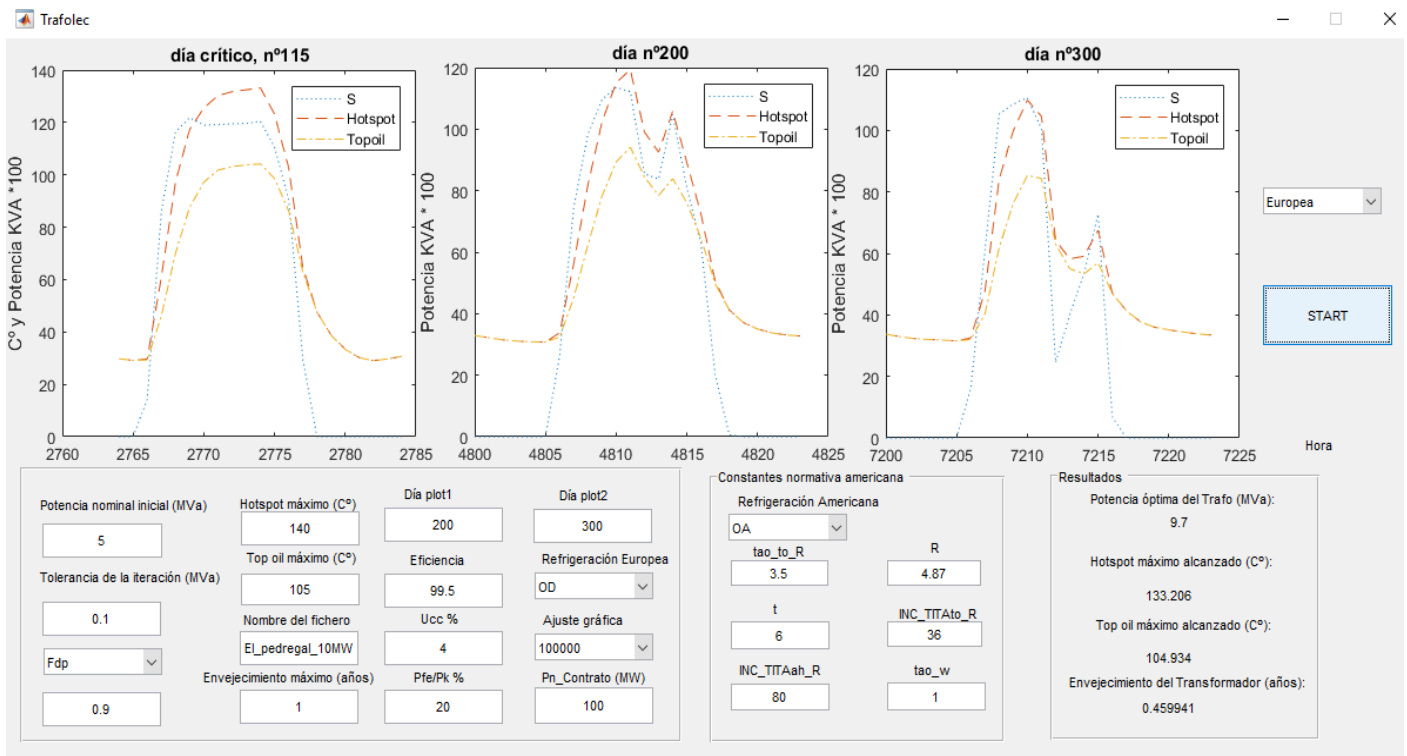


Figura 39. El_pedregal_10MW.3

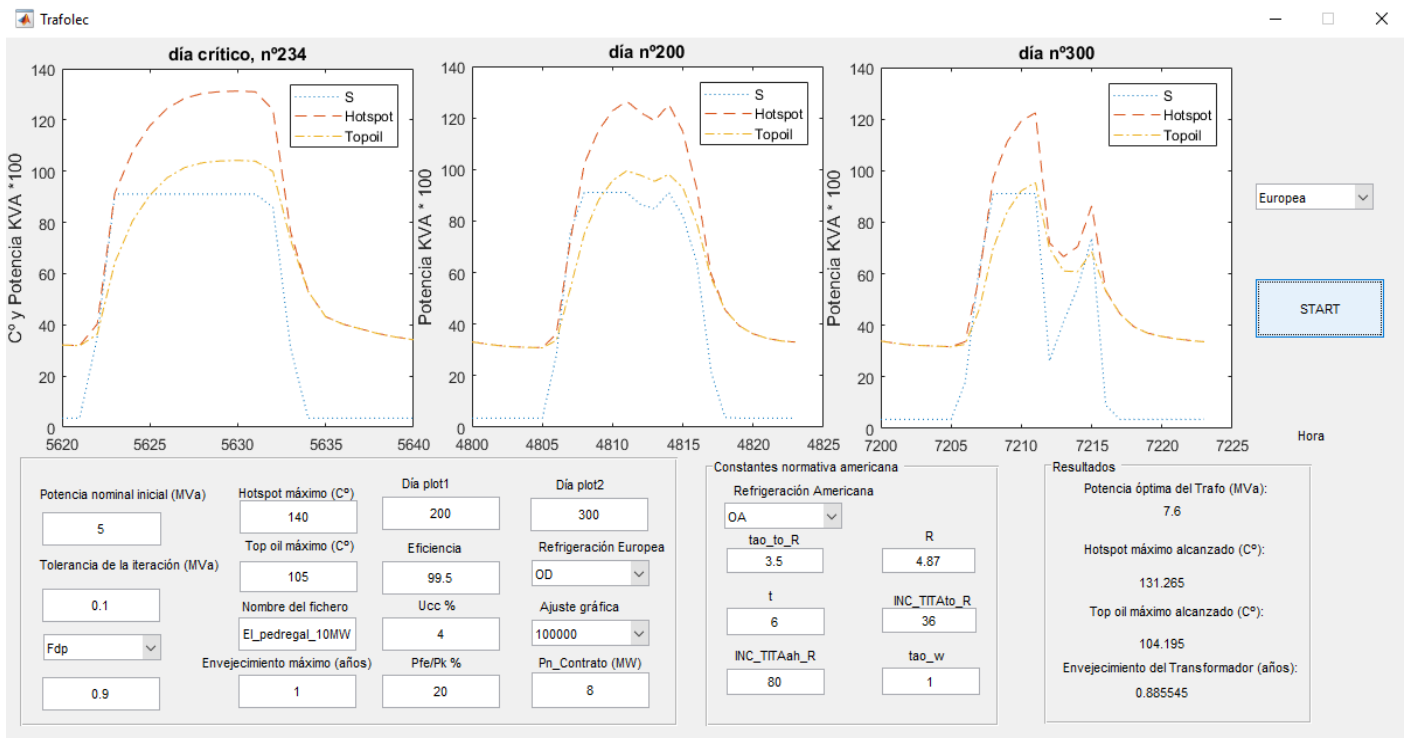


Figura 40. El_pedregal_10MW.2

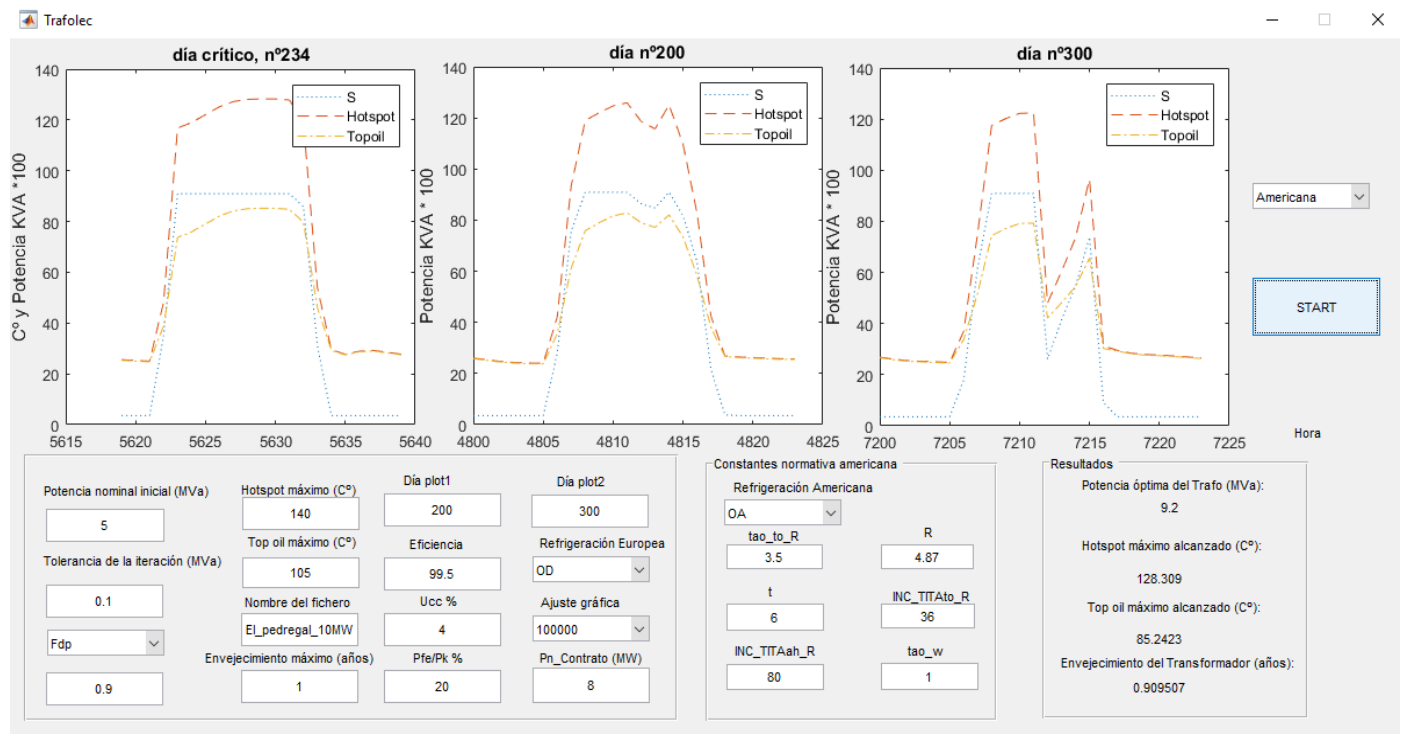


Figura 41. El_pedregal_10MW.3

7. CONCLUSIONES

Tras el desarrollo del programa y ser probado con diferentes plantas fotovoltaicas hemos llegado a las siguientes conclusiones.

El objetivo principal del proyecto es abaratar los costes de la planta fotovoltaica calculando el transformador óptimo para la misma, y como se puede comprobar en el estudio de las plantas fotovoltaicas, el cálculo final de transformadores suele encontrarse por debajo de la potencia instalada de la planta, lo que quiere decir que esta reducción del transformador es óptima para el proyecto de la planta fotovoltaica y abarataría costes de la misma.

El programa desarrollado servirá para hacer proyectos de plantas fotovoltaicas complementando a la herramienta PVsyst y llevando los transformadores a sus límites, aprovechando al máximo la potencia de los transformadores seleccionados, con una pérdida lo más reducida posible de vida útil.

Existe diferencia entre la norma americana y la norma europea, las cuales, teniendo cálculos similares, obtienen resultados algo dispares. La norma americana es más desfavorable, dando resultados de transformadores con mayor potencia que los obtenidos con la normativa europea. Se puede suponer que la norma europea es más precisa debido a que ella diferencia entre el aumento y disminución de carga del transformador utilizando diferentes cálculos entre un caso y otro, mientras que la norma americana no los diferencia.

Otra conclusión obtenida de este proyecto tiene que ver con el envejecimiento del transformador. Como se ha explicado antes, la potencia óptima del transformador se encontrará por debajo de la potencia instalada de la planta, esto provocará una sobrecarga casi diaria en el transformador. La sobrecarga disminuirá la vida útil del transformador, conociendo gracias a este dato la vida útil real del transformador. Esto puede ser una forma de mantenimiento predictivo, pudiendo predecir cuándo cambiar el transformador por otro nuevo y así no perder producción de energía.

8. BIBLIOGRAFÍA

- [1] C57.91 -1995/Cor 1-2002 IEEE Guide for Loading Mineral-Oil-Immersed Transformers.
- [2] IEC 60076-7 Power Transformers - Part7: Loading guide for oil immersed power Transformers.
- [3] MÁQUINAS ELÉCTRICAS JESÚS FRAILE MORA quinta edición, edición Mc Graw Hill.
- [4] <http://www.energetica21.com/descargar.php?seccion=articulos&archivo=wtMlmba9wPVRz5F401AH0w6VBCYgFR22Ccme3kQH7h7zmyycz6qk1A.pdf>
(visitada el 09/05/2019)
- [5] <https://fornieles.es/energia-reactiva/compensacion-reactiva-transformador/>
(visitada el 12/05/2019)
- [6] Transformadores de potencia, de medida y de protección / Enrique Ras Oliva, edición Marcombo.
- [7] Cálculo óptimo de transformadores / Juan Corrales Martín, edición Marcombo.
- [8] MATLAB guide / Desmond J. Higham, Nicholas J. Higham, edición Siam.
- [9] Estimación energía producida por una planta solar fotovoltaica (1,2 MW) conectada a la red / Saray San Martín Santos; director, Andrés Melgar Bachiller
- [10] <https://new.abb.com/docs/librariesprovider78/eventos/jjts-2017/presentaciones-chile/como-especificar-transformadores-de-potencia-luiz-yamazaki.pdf?sfvrsn=2>
(visitada el 25/05/2019)
- [11] <https://files.sma.de/dl/18858/Q-at-Night-TI-es-11.pdf>
(visitada el 03/06/2019)

8.1 FUENTES DE LAS IMÁGENES

Figura 1. Producción diaria planta fotovoltaica.

<https://www.meteoforenergy.com/es/servicios/energias-renovables/fotovoltaica/>
(visitada el 16/04/2019)

Figura 3: Imagen Asignatura energías renovables. Energía solar fotovoltaica, pp. 3.

Figura 4:

https://www.google.com/url?sa=i&source=images&cd=&cad=rja&uact=8&ved=2ahUKEwiJ1Jm340XhAhVnxYUKHQe0CGwQjRx6BAGBEAU&url=https%3A%2F%2Flistado.mercadolibre.com.mx%2Fpaneles-solares-costco&psig=A0vVaw1Et_cAM2v_UbW16n77QyQl&ust=1556092978640201
(visitada el 23/04/2019)

Figura 5:

https://www.google.com/imgres?imgurl=https%3A%2F%2Fmadridcamper.com%2Fc%2F46-medium_default_2x%2Fonda-pura-inversores-de-corriente.jpg&imgrefurl=https%3A%2F%2Fmadridcamper.com%2F54-cargadores-inversores-booster&docid=NNJBKTf5xIWHhM&tbnid=FnbRMTiRSgCO1M%3A&vet=10ahUKEwjrneOm4eXhAhWzAGMBHY7DCPYQMwjnAShHMEc..i&w=240&h=276&bih=657&biw=1366&q=inversores&ved=0ahUKEwjrneOm4eXhAhWzAGMBHY7DCPYQMwjnAShHMEc&iact=mrc&uact=8
(visitada el 23/04/2019)

Figura 6: Imagen obtenida de fuente MÁQUINAS ELÉCTRICAS JESÚS FRAILE MORA quinta edición pp. 167.

Figura 7: Imagen obtenida de fuente MÁQUINAS ELÉCTRICAS JESÚS FRAILE MORA quinta edición pp. 185.

Figura 8: Imagen obtenida de la IEC 60076-7 parte 7 pp. 28

ANEXO 1: CÓDIGO DEL PROGRAMA

```
function varargout = Trafolec(varargin)
% TRAFOLEC MATLAB code for Trafolec.fig
%     TRAFOLEC, by itself, creates a new TRAFOLEC or raises the
existing
%     singleton*.
%
%     H = TRAFOLEC returns the handle to a new TRAFOLEC or the handle
to
%     the existing singleton*.
%
%     TRAFOLEC('CALLBACK',hObject,eventData,handles,...) calls the
local
%     function named CALLBACK in TRAFOLEC.M with the given input
arguments.
%
%     TRAFOLEC('Property','Value',...) creates a new TRAFOLEC or
raises the
%     existing singleton*. Starting from the left, property value
pairs are
%     applied to the GUI before Trafolec_OpeningFcn gets called. An
%     unrecognized property name or invalid value makes property
application
%     stop. All inputs are passed to Trafolec_OpeningFcn via
varargin.
%
%     *See GUI Options on GUIDE's Tools menu. Choose "GUI allows
only one
%     instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Edit the above text to modify the response to help Trafolec

% Last Modified by GUIDE v2.5 02-Apr-2019 20:28:21

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                  'gui_Singleton',   gui_Singleton, ...
                  'gui_OpeningFcn', @Trafolec_OpeningFcn, ...
                  'gui_OutputFcn',  @Trafolec_OutputFcn, ...
                  'gui_LayoutFcn',  [], ...
                  'gui_Callback',    []);
if nargin && ischar(varargin{1})
```



```

        gui_State.gui_Callback = str2func(varargin{1});
    end

    if nargin
        [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
    else
        gui_mainfcn(gui_State, varargin{:});
    end
% End initialization code - DO NOT EDIT

% --- Executes just before Trafolec is made visible.
function Trafolec_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to Trafolec (see VARARGIN)

% Choose default command line output for Trafolec
handles.output = hObject;

set(handles.Hotspot_M, 'String', '140');
set(handles.TopOil_M, 'String', '105');
set(handles.INC_TITAto_R, 'String', '36');           %Hottest-spot
conductor rise over top-oil temperature, at rated load.
set(handles.R, 'String', '4.87');                   %Ratio of load loss
at rated load to no-load loss
set(handles.tao_to_R, 'String', '3.5');
set(handles.INC_TITAah_R, 'String', '80');          %80C° for 65C°
average winding rise and 65C° for 55C° average winding rise.
set(handles.t, 'String', '6');                      %is the duration of
load hours.
set(handles.tao_w, 'String', '1');
set(handles.Eficiencia, 'String', '99.5');
set(handles.Zc, 'String', '4');
set(handles.fichero, 'String', 'dato_meteo');
set(handles.Perdidas, 'String', '20');
% Update handles structure
guidata(hObject, handles);

% UIWAIT makes Trafolec wait for user response (see UIRESUME)
% uiwait(handles.figure1);

% --- Outputs from this function are returned to the command line.
function varargout = Trafolec_OutputFcn(hObject, eventdata, handles)
% varargout  cell array for returning output args (see VARARGOUT);
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;

function Pn_dia_Callback(hObject, eventdata, handles)

```

```

% hObject    handle to Pn_dia (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of Pn_dia as text
%         str2double(get(hObject,'String')) returns contents of Pn_dia
as a double

% --- Executes during object creation, after setting all properties.
function Pn_dia_CreateFcn(hObject, eventdata, handles)
% hObject    handle to Pn_dia (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%         See ISPC and COMPUTER.

if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUiControlBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function fichero_Callback(hObject, eventdata, handles)
% hObject    handle to fichero (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of fichero as text
%         str2double(get(hObject,'String')) returns contents of fichero
as a double

% --- Executes during object creation, after setting all properties.
function fichero_CreateFcn(hObject, eventdata, handles)
% hObject    handle to fichero (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%         See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUiControlBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function ndias_Callback(hObject, eventdata, handles)
% hObject    handle to ndias (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

```

```

% Hints: get(hObject,'String') returns contents of ndias as text
%         str2double(get(hObject,'String')) returns contents of ndias
as a double

% --- Executes during object creation, after setting all properties.
function ndias_CreateFcn(hObject, eventdata, handles)
% hObject    handle to ndias (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%         See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function Fdp_Callback(hObject, eventdata, handles)
% hObject    handle to Fdp (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of Fdp as text
%         str2double(get(hObject,'String')) returns contents of Fdp as
a double

% --- Executes during object creation, after setting all properties.
function Fdp_CreateFcn(hObject, eventdata, handles)
% hObject    handle to Fdp (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%         See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function Perd_Fe_Callback(hObject, eventdata, handles)
% hObject    handle to Perd_Fe (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of Perd_Fe as text
%         str2double(get(hObject,'String')) returns contents of Perd_Fe
as a double

```

```

% --- Executes during object creation, after setting all properties.
function Perd_Fe_CreateFcn(hObject, eventdata, handles)
% hObject    handle to Perd_Fe (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function Perd_Cu_Callback(hObject, eventdata, handles)
% hObject    handle to Perd_Cu (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of Perd_Cu as text
%       str2double(get(hObject,'String')) returns contents of Perd_Cu
as a double

% --- Executes during object creation, after setting all properties.
function Perd_Cu_CreateFcn(hObject, eventdata, handles)
% hObject    handle to Perd_Cu (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function Tol_Callback(hObject, eventdata, handles)
% hObject    handle to Tol (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of Tol as text
%       str2double(get(hObject,'String')) returns contents of Tol as
a double

% --- Executes during object creation, after setting all properties.
function Tol_CreateFcn(hObject, eventdata, handles)
% hObject    handle to Tol (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

```

```

% Hint: edit controls usually have a white background on Windows.
%     See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function Hotspot_M_Callback(hObject, eventdata, handles)
% hObject     handle to Hotspot_M (see GCBO)
% eventdata   reserved - to be defined in a future version of MATLAB
% handles     structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of Hotspot_M as text
%         str2double(get(hObject,'String')) returns contents of
Hotspot_M as a double

% --- Executes during object creation, after setting all properties.
function Hotspot_M_CreateFcn(hObject, eventdata, handles)
% hObject     handle to Hotspot_M (see GCBO)
% eventdata   reserved - to be defined in a future version of MATLAB
% handles     empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%     See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function TopOil_M_Callback(hObject, eventdata, handles)
% hObject     handle to TopOil_M (see GCBO)
% eventdata   reserved - to be defined in a future version of MATLAB
% handles     structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of TopOil_M as text
%         str2double(get(hObject,'String')) returns contents of
TopOil_M as a double

% --- Executes during object creation, after setting all properties.
function TopOil_M_CreateFcn(hObject, eventdata, handles)
% hObject     handle to TopOil_M (see GCBO)
% eventdata   reserved - to be defined in a future version of MATLAB
% handles     empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%     See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

```

end

```
function ageing_M_Callback(hObject, eventdata, handles)
% hObject      handle to ageing_M (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of ageing_M as text
%         str2double(get(hObject,'String')) returns contents of
ageing_M as a double
```

```
% --- Executes during object creation, after setting all properties.
function ageing_M_CreateFcn(hObject, eventdata, handles)
% hObject      handle to ageing_M (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```

```
function ajuste_Callback(hObject, eventdata, handles)
% hObject      handle to ajuste (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of ajuste as text
%         str2double(get(hObject,'String')) returns contents of ajuste
as a double
```

```
% --- Executes during object creation, after setting all properties.
function ajuste_CreateFcn(hObject, eventdata, handles)
% hObject      handle to ajuste (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```

```
function Dia_plot1_Callback(hObject, eventdata, handles)
```

```

% hObject    handle to Dia_plot1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of Dia_plot1 as text
%         str2double(get(hObject,'String')) returns contents of
Dia_plot1 as a double

% --- Executes during object creation, after setting all properties.
function Dia_plot1_CreateFcn(hObject, eventdata, handles)
% hObject    handle to Dia_plot1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUiControlBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function Dia_plot2_Callback(hObject, eventdata, handles)
% hObject    handle to Dia_plot2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of Dia_plot2 as text
%         str2double(get(hObject,'String')) returns contents of
Dia_plot2 as a double

% --- Executes during object creation, after setting all properties.
function Dia_plot2_CreateFcn(hObject, eventdata, handles)
% hObject    handle to Dia_plot2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUiControlBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

% --- Executes on selection change in Normativa.
function Normativa_Callback(hObject, eventdata, handles)
% hObject    handle to Normativa (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns Normativa
contents as cell array

```

```

%         contents{get(hObject,'Value')} returns selected item from
Normativa

% --- Executes during object creation, after setting all properties.
function Normativa_CreateFcn(hObject, eventdata, handles)
% hObject    handle to Normativa (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: popupmenu controls usually have a white background on Windows.
%         See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

% --- Executes on selection change in Tipo_Trafo.
function Tipo_Trafo_Callback(hObject, eventdata, handles)
% hObject    handle to Tipo_Trafo (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

guidata(hObject, handles);

% Hints: contents = cellstr(get(hObject,'String')) returns Tipo_Trafo
contents as cell array
%         contents{get(hObject,'Value')} returns selected item from
Tipo_Trafo

% --- Executes during object creation, after setting all properties.
function Tipo_Trafo_CreateFcn(hObject, eventdata, handles)
% hObject    handle to Tipo_Trafo (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: popupmenu controls usually have a white background on Windows.
%         See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

% --- Executes on slider movement.
function ajusta_grafica_Callback(hObject, eventdata, handles)
% hObject    handle to ajusta_grafica (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'Value') returns position of slider
%         get(hObject,'Min') and get(hObject,'Max') to determine range
of slider

```



```

% --- Executes during object creation, after setting all properties.
function ajusta_grafica_CreateFcn(hObject, eventdata, handles)
% hObject    handle to ajusta_grafica (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: slider controls usually have a light gray background.
if isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUiControlBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9 .9]);
end

% --- Executes on selection change in Ajuste.
function Ajuste_Callback(hObject, eventdata, handles)
% hObject    handle to Ajuste (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns Ajuste
contents as cell array
%         contents{get(hObject,'Value')} returns selected item from
Ajuste

% --- Executes during object creation, after setting all properties.
function Ajuste_CreateFcn(hObject, eventdata, handles)
% hObject    handle to Ajuste (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: popupmenu controls usually have a white background on Windows.
%         See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUiControlBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit15_Callback(hObject, eventdata, handles)
% hObject    handle to edit15 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit15 as text
%         str2double(get(hObject,'String')) returns contents of edit15
as a double

% --- Executes during object creation, after setting all properties.
function edit15_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit15 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

```

```

% Hint: edit controls usually have a white background on Windows.
%     See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function Eficiencia_Callback(hObject, eventdata, handles)
% hObject     handle to Eficiencia (see GCBO)
% eventdata   reserved - to be defined in a future version of MATLAB
% handles     structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of Eficiencia as text
%     str2double(get(hObject,'String')) returns contents of
Eficiencia as a double

% --- Executes during object creation, after setting all properties.
function Eficiencia_CreateFcn(hObject, eventdata, handles)
% hObject     handle to Eficiencia (see GCBO)
% eventdata   reserved - to be defined in a future version of MATLAB
% handles     empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%     See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function Perdidas_Callback(hObject, eventdata, handles)
% hObject     handle to Perdidas (see GCBO)
% eventdata   reserved - to be defined in a future version of MATLAB
% handles     structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of Perdidas as text
%     str2double(get(hObject,'String')) returns contents of
Perdidas as a double

% --- Executes during object creation, after setting all properties.
function Perdidas_CreateFcn(hObject, eventdata, handles)
% hObject     handle to Perdidas (see GCBO)
% eventdata   reserved - to be defined in a future version of MATLAB
% handles     empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%     See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

```

```
end
```

```
function Zc_Callback(hObject, eventdata, handles)
% hObject    handle to Zc (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of Zc as text
%        str2double(get(hObject,'String')) returns contents of Zc as a
double
```

```
% --- Executes during object creation, after setting all properties.
function Zc_CreateFcn(hObject, eventdata, handles)
% hObject    handle to Zc (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called
```

```
% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
```

```
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```

```
% --- Executes on button press in START.
```

```
function START_Callback(hObject, eventdata, handles)
% hObject    handle to START (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
ndias=365; %numero de horas de la planta a analizar.
Fichero=get(handles.fichero,'String');
%nombre del excell que contiene los datos de la planta.
PnTrafo= str2double(get(handles.Pn_dia,'String'));%Potencia inicial de
iteración del Transformador en KVA.
Trafo=get(handles.Tipo_Trafo,'Value');
Normativa=get(handles.Normativa,'Value');
Trafo_americano=get(handles.Trafo_americano,'Value');
React=get(handles.Reactiva,'Value');
Reactiva_fdp=str2double(get(handles.Fdp,'String'));
%Factor de potencia.
Efi=str2double(get(handles.Eficiencia,'String'));
Perd=str2double(get(handles.Perdidas,'String'));
Zc=str2double(get(handles.Zc,'String'));
tol=str2double(get(handles.Tol,'String')); %Tolerancia
de iteración.
Hotspot_max=str2double(get(handles.Hotspot_M,'String'));
%140c° según la norma.
Topoil_max=str2double(get(handles.TopOil_M,'String'));
%105c° según la norma.
Ageing_max=str2double(get(handles.ageing_M,'String'));
%ageing máx que permitido en días.
Ajuste=get(handles.Ajuste,'Value'); %Ajusta la potencia para verse
con el top oil y el hotspot.
Day_Cr=3; %El día crítico muestra el día con mayor
hotspot.
```

```

Day_plot1=str2double(get(handles.Dia_plot1,'String'));
%Los day plot te muestran el día que quieras de todos los estudiados.
Day_plot2=str2double(get(handles.Dia_plot2,'String'));
%Si se pone Day_plot1=inf no aparecerá la gráfica.
INC_TITAto_R=str2double(get(handles.INC_TITAto_R,'String'));
%Hottest-spot conductor rise over top-oil temperature, at rated load.
R=str2double(get(handles.R,'String')); %Ratio of
load loss at rated load to no-load loss
tao_to_R=str2double(get(handles.tao_to_R,'String'));
INC_TITAah_R=str2double(get(handles.INC_TITAah_R,'String'));
%80C° for 65C° average winding rise and 65C° for 55C° average winding
rise.
t=str2double(get(handles.t,'String')); %is the
duration of load hours.
tao_w=str2double(get(handles.tao_w,'String')); %tao_w
is te winding time constant at hot spot location hours.
Pn_Contrato=(str2double(get(handles.Pn_Contrato,'String')))*1000000;
n=ndias*24;

```

%TRATAMIENTO DE DATOS

```

num =xlsread(Fichero);
X=ones(1,n);
T=ones(1,n);
HTrafo=1:n-1;

%Bucle para la energía inyectada en ndías
for i=1:n
    X(1,i)=num(i,2);
    T(1,i)=num(i,1);
    if X(1,i)<0
        X(1,i)=0;
    end
end

Xvar=ones(n-1,1);
Xvar_react=ones(n-1,1);
Xvar_react_perd=ones(n-1,1);
Svar=ones(n-1,1);
for i=1:n-1
    Xvar(i,1)=(X(1,i+1)-X(1,i));
    T(1,i)=(T(1,i)+T(1,i+1))/2;
end

for i=1:n-1
    if i==1
        Xvar(i,1)= Xvar(i,1)+ Xvar(i,1);
    else
        Xvar(i,1)= Xvar(i,1)+ Xvar(i-1,1);
    end
end

L=0;
for i=1:n-1
    if Xvar(i,1)>Pn_Contrato;
        L=1;
    end
end

```

```
end
end
```

```
%FIN TRATAMIENTO DE DATOS
```

```
switch Ajuste
  case 1
    Ajuste_grafica=100000000;
  case 2
    Ajuste_grafica=10000000;
  case 3
    Ajuste_grafica=1000000;
  case 4
    Ajuste_grafica=100000;
  case 5
    Ajuste_grafica=10000;
  case 6
    Ajuste_grafica=1000;
  case 7
    Ajuste_grafica=100;
  case 8
    Ajuste_grafica=10;
  case 9
    Ajuste_grafica=1;
end
```

```
switch Normativa
  case 1 %Europea
switch Trafo %características térmicas
  case 1
    x=1;
    y=2;
    k11=1;
    k21=1;
    k22=1;
    tao0=90;
    taoW=7;
    R=6;
    H=1.3;
    INC_TITAomr=46;
    INC_TITAor=49;
    INC_TITAh=29;
    INC_TITAbr=43;
    TITAh=98;
    Gr=14.5;
  case 2
    x=1;
    y=1.3;
    k11=1;
    k21=1.3;
    k22=1;
    tao0=90;
    taoW=7;
    R=6;
    H=1.3;
    INC_TITAomr=46;
    INC_TITAor=56;
    INC_TITAh=22;
```

```

INC_TITAbr=36;
TITAh=98;
Gr=14.5;
case 3
x=0.8;
y=1.3;
k11=0.5;
k21=2;
k22=2;
tao0=210;
taoW=10;
R=6;
H=1.3;
INC_TITAomr=43;
INC_TITAor=52;
INC_TITAh=26;
INC_TITAbr=34;
TITAh=98;
Gr=14.5;
case 4
x=0.8;
y=1.3;
k11=0.5;
k21=2;
k22=2;
tao0=150;
taoW=7;
R=6;
H=1.3;
INC_TITAomr=43;
INC_TITAor=52;
INC_TITAh=26;
INC_TITAbr=34;
TITAh=98;
Gr=14.5;
end

%Ta=Te(Tyear,Tmonth_max); % Temperatura media

%valores iniciales
K=0;
Ta=T(1,1);
INC_TITAoi= INC_TITAoi_initial(INC_TITAor,R,K,Ta,x);
INC_TITAhi= INC_TITAhi_initial(k21,INC_TITAh,R,K,y);

ft1=f1(60,k11,tao0);
ft2=f2(60,k22,tao0,taoW,k21);
ft3=f3(60,tao0,k11);

X=X';
Hotspot=ones(n-1,1);
Topoil=ones(n-1,1);
Pn=PnTrafo*1000000;
err1=1;
err2=1;
iteraciones=0;

while err2==1

```

```

while err1==1
    %Cálculo Pérdidas potencia activa
    Pcu=((1-(Efi/100))*Pn_Contrato)/(1+(Perd/100));
    PFe=Pcu*(Perd/100);
    %Cálculo la S sin perdidas
    if React==1
        Sn_Contrato=Pn_Contrato/Reactiva_fdp;
    end
    if React==2
        Sn_Contrato=sqrt((Reactiva_fdp)^2+(Pn_Contrato)^2);
    end
    if L==0
        if React==1
            for i=1:n-1
                Xvar_react(i,1)=(Xvar(i,1)*tan(acos(Reactiva_fdp)));
                Xvar_react_perd(i,1)=Xvar_react(i,1)+(Zc/100)*Xvar_react(i,1);
            end
        end
        if React==2
            for i=1:n-1
                Xvar_react(i,1)=Reactiva_fdp*1000000;
                Xvar_react_perd(i,1)=Xvar_react(i,1)+(Zc/100)*Xvar_react(i,1);
            end
        end

    end

    if L==1
        Pn_Contrato_Perd=Pn_Contrato+(PFe+Pcu*(Sn_Contrato/Pn)^2);
        for i=1:n-1
            if Xvar(i,1)>Pn_Contrato_Perd
                Xvar(i,1)=Pn_Contrato_Perd;
            end
        end
    end
    if React==1
        for i=1:n-1

            Xvar_react_perd(i,1)=(Xvar(i,1)*tan(acos(Reactiva_fdp)))+(Zc/100)*Sn_Contrato;
        end
    end
    if React==2
        for i=1:n-1
            Xvar_react_perd(i,1)=Reactiva_fdp*1000000+(Zc/100)*Sn_Contrato;
        end
    end

    end

    %Xvar se encuentra en MW ya que el Pvsyst te da la potencia activa.
    %Obtenemos la S sin pérdidas;

    for i=1:n-1
        Svar(i,1)=sqrt((Xvar_react_perd(i,1))^2+(Xvar(i,1))^2);
    end
    %Ahora Xvar(S) es la potencia en MVA

    Ks=ones(n-1,1);
    Ks=Svar./(Pn);    %load factor

```

```

err1=0;
err2=0;
for i=1:n-1
    K=Ks(i,1);
    Ta=T(1,i);
    if i==1
        Hotspot(i,1) = Tincr(
Ta,INC_TITAOi,INC_TITAor,R,K,x,ft1,INC_TITAhI,H,Gr,y,ft2);
        INC_TITAhI= INC_TITAhI_increase(INC_TITAhI,H,Gr,K,y,ft2);
        INC_TITAOi= INC_TITAOi_increase(INC_TITAOi,INC_TITAor,R,K,x,ft1);
        Topoil(i,1)= INC_TITAOi+Ta;
        if Hotspot(i,1)>Hotspot_max | Topoil>Topoil_max
            err1=1;
        end

        else

        if Ks(i-1,1)<Ks(i,1)
            Hotspot(i,1) = Tincr(
Ta,INC_TITAOi,INC_TITAor,R,K,x,ft1,INC_TITAhI,H,Gr,y,ft2);
            INC_TITAhI= INC_TITAhI_increase(INC_TITAhI,H,Gr,K,y,ft2);
            INC_TITAOi= INC_TITAOi_increase(INC_TITAOi,INC_TITAor,R,K,x,ft1);
            Topoil(i,1)= INC_TITAOi+Ta;
            if Hotspot(i,1)>Hotspot_max | Topoil>Topoil_max
                err1=1;
            end
        elseif Ks(i-1,1)>Ks(i,1)
            Hotspot(i,1) = Tdecrease(
Ta,INC_TITAOi,INC_TITAor,R,K,x,H,Gr,y,ft3);
            INC_TITAhI= INC_TITAhI_decrease(H,Gr,K,y);
            INC_TITAOi= INC_TITAOi_decrease(INC_TITAOi,INC_TITAor,R,K,x,ft3);
            Topoil(i,1)= INC_TITAOi+Ta;
            if Hotspot(i,1)>Hotspot_max | Topoil>Topoil_max
                err1=1;
            end
        elseif Ks(i-1,1)==Ks(i,1) && Ks(i,1)~=(PFe/Pn)
            Hotspot(i,1) = Tdecrease(
Ta,INC_TITAOi,INC_TITAor,R,K,x,H,Gr,y,ft3);
            INC_TITAhI= INC_TITAhI_decrease(H,Gr,K,y);
            INC_TITAOi= INC_TITAOi_decrease(INC_TITAOi,INC_TITAor,R,K,x,ft3);
            Topoil(i,1)= INC_TITAOi+Ta;
            if Hotspot(i,1)>Hotspot_max | Topoil>Topoil_max
                err1=1;
            end
        elseif Ks(i,1)==(PFe/Pn);
            Hotspot(i,1)=T(1,i);
            Topoil(i,1)=T(1,i);
            if Hotspot(i,1)>Hotspot_max | Topoil>Topoil_max
                err1=1;
            end
        end

        end

end
if err1==1
Pn=Pn+(tol*1000000);
end
iteraciones=iteraciones+1;

```



```

end

Loss_life=0;
ageing_day=ones(n-1,1);
for i=1:n-1
    Hs=Hotspot(i,1);
    ageing_day(i,1)=relative_ageing(Hs);
    ageing_step=relative_ageing(Hs);
    Loss_life=Loss_life+ageing_step;
end
Loss_life_days=Loss_life/(24*365);

if Loss_life_days>Ageing_max
    err2=1;
    err1=1;
    Pn=Pn+(tol*1000000);
end
end
case 2
switch Trafo_americano %características térmicas
case 1
    nc=0.8;
    m=0.8;
case 2
    nc=0.9;
    m=0.8;
case 3
    nc=0.9;
    m=0.8;
case 4
    m=1;
    nc=1;
end

%constantes americanas
C=5.07934;
X=X';
Hotspot=ones(n-1,1);
Topoil=ones(n-1,1);
Pn=PnTrafo*1000000;
err1=1;
err2=1;
iteraciones=0;

while err2==1
while err1==1
    %Cálculo Pérdidas potencia activa
    Pcu=((1-(Efi/100))*Pn_Contrato)/(1+(Perd/100));
    PFe=Pcu*(Perd/100);
    %Calculo la S sin perdidas
    if React==1
        Sn_Contrato=Pn_Contrato/Reactiva_fdp;
    end
    if React==2
        Sn_Contrato=sqrt((Reactiva_fdp)^2+(Pn_Contrato)^2);
    end
    if L==0
        if React==1
            for i=1:n-1
                Xvar_react(i,1)=(Xvar(i,1)*tan(acos(Reactiva_fdp)));
            end
        end
    end
end
end

```

```

        Xvar_react_perd(i,1)=Xvar_react(i,1)+(Zc/100)*Xvar_react(i,1);
    end
end
if React==2
    for i=1:n-1
        Xvar_react(i,1)=Reactiva_fdp*1000000;
        Xvar_react_perd(i,1)=Xvar_react(i,1)+(Zc/100)*Xvar_react(i,1);
    end
end

end

if L==1
    Pn_Contrato_Perd=Pn_Contrato+(PFe+Pcu*(Sn_Contrato/Pn)^2);
    for i=1:n-1
        if Xvar(i,1)>Pn_Contrato_Perd
            Xvar(i,1)=Pn_Contrato_Perd;
        end
    end
    if React==1
        for i=1:n-1

            Xvar_react_perd(i,1)=(Xvar(i,1)*tan(acos(Reactiva_fdp)))+(Zc/100)*Sn_Contrato;
        end
    end
    if React==2
        for i=1:n-1
            Xvar_react_perd(i,1)=Reactiva_fdp*1000000+(Zc/100)*Sn_Contrato;
        end
    end

end
%Xvar se encuentra en MW ya que el Pvsyst te da la potencia activa.
%Obtenemos la S sin pérdidas;

for i=1:n-1
    Svar(i,1)=sqrt((Xvar_react_perd(i,1))^2+(Xvar(i,1))^2);
end
%Ahora Xvar(S) es la potencia en MVA

Ks=ones(n-1,1);
Xvar_perd=ones(n-1,1);
Xvar_perd=Xvar;
Ks=Svar./(Pn);    %load factor
err1=0;
err2=0;
K_i=Ks(1,1);
INC_TITAto_i_p=INC_TITAto_i(INC_TITAto_R,K_i,nc,R);
INC_TITAh_i_p=INC_TITAh_i(INC_TITAh_R,K_i,m);
for i=1:n-1
    K=Ks(i,1);
    Ta=T(1,i);
    INC_TITAto_u_p=INC_TITAto_u(INC_TITAto_R,K,nc,R);

tao_to_p=tao_to(INC_TITAto_u_p,INC_TITAto_i_p,INC_TITAto_R,nc,tao_to_R);
INC_TITAo_p=INC_TITAo(INC_TITAto_u_p,INC_TITAto_i_p,tao_to_p);

```

```

        if tao_to_p <=0
            INC_TITAO_p=0;
        end
INC_TITAh_R_p=INC_TITAh_R(INC_TITAAh_R,INC_TITAtO_R);
INC_TITAh_u_p=INC_TITAh_u(INC_TITAh_R_p,K,m);
INC_TITAh_p=INC_TITAh(INC_TITAh_u_p,INC_TITAh_i_p,tao_w,t);
Hotspot(i,1)=Ta+INC_TITAO_p+INC_TITAh_p;
Topoil(i,1)=Ta+INC_TITAO_p;
if Hotspot(i,1)>Hotspot_max | Topoil>Topoil_max
    err1=1;
end
end
if err1==1
Pn=Pn+(tol*1000000);
end
iteraciones=iteraciones+1;
end

Loss_life=0;
ageing_day=ones(n-1,1);
for i=1:n-1
    Hs=Hotspot(i,1);
    ageing_day(i,1)=relative_ageing(Hs);
    ageing_step=relative_ageing(Hs);
    Loss_life=Loss_life+ageing_step;
end
Loss_life_days=Loss_life/(24*365);

if Loss_life_days>Ageing_max
    err2=1;
    err1=1;
    Pn=Pn+(tol*1000000);
end
end

end

%ANÁLISIS DE DATOS

Resultados=[HTrafo',Svar,Hotspot,Topoil,ageing_day,Xvar];
[Hora,col]=find(Resultados==max(Resultados(:,3)));
Day_Crit=floor(Hora(1,1)/24);
HTrafo_Crit=(Hora(1,1)-10):(Hora(1,1)+10);
HTrafo_Chosen1=(Day_plot1*24):((Day_plot1*24)+23);
HTrafo_Chosen2=(Day_plot2*24):((Day_plot2*24)+23);
P_Crit=ones(1,20);
Topoil_Crit=ones(1,20);
Hotspot_Crit=ones(1,20);
P_Chosen1=ones(1,24);
Topoil_Chosen1=ones(1,24);
Hotspot_Chosen1=ones(1,24);
P_Chosen2=ones(1,24);
Topoil_Chosen2=ones(1,24);
Hotspot_Chosen2=ones(1,24);
j=1;

for i=(Hora(1,1)-10):(Hora(1,1)+10)

```

```

P_Crit(1,j)=Resultados(i,2)/Ajuste_grafica;
Hotspot_Crit(1,j)=Resultados(i,3);
Topoil_Crit(1,j)=Resultados(i,4);
j=j+1;
end
axes(handles.plot_diac);
plot(HTrafo_Crit,P_Crit,':',HTrafo_Crit,Hotspot_Crit,'--',HTrafo_Crit,Topoil_Crit,'-.');
str=['día crítico, n°',num2str(Day_Crit)];
title(str)
str2=['C° y Potencia KVA *',num2str(Ajuste_grafica/1000)];
ylabel(str2)
legend('S','Hotspot','Topoil')

j=1;

if Day_plot1==0
for i=1:24
P_Chosen1(1,j)=Resultados(i,2)/Ajuste_grafica;
Hotspot_Chosen1(1,j)=Resultados(i,3);
Topoil_Chosen1(1,j)=Resultados(i,4);
j=j+1;
end
else
for i=(24*Day_plot1):((Day_plot1*24)+23)
P_Chosen1(1,j)=Resultados(i,2)/Ajuste_grafica;
Hotspot_Chosen1(1,j)=Resultados(i,3);
Topoil_Chosen1(1,j)=Resultados(i,4);
j=j+1;
end
end
axes(handles.plot1);
plot(HTrafo_Chosen1,P_Chosen1,':',HTrafo_Chosen1,Hotspot_Chosen1,'--',HTrafo_Chosen1,Topoil_Chosen1,'-.');
str=['día n°',num2str(Day_plot1)];
title(str);
str2=['Potencia KVA *',num2str(Ajuste_grafica/1000)];
ylabel(str2)
legend('S','Hotspot','Topoil')

j=1;

if Day_plot2==0
for i=1:24
P_Chosen2(1,j)=Resultados(i,2)/Ajuste_grafica;
Hotspot_Chosen2(1,j)=Resultados(i,3);
Topoil_Chosen2(1,j)=Resultados(i,4);
j=j+1;
end
else
for i=(24*Day_plot2):((Day_plot2*24)+23)
P_Chosen2(1,j)=Resultados(i,2)/Ajuste_grafica;
Hotspot_Chosen2(1,j)=Resultados(i,3);
Topoil_Chosen2(1,j)=Resultados(i,4);
j=j+1;
end
end

```

```

end
axes(handles.plot2);
plot(HTrafo_Chosen2,P_Chosen2,':',HTrafo_Chosen2,Hotspot_Chosen2,'--
',HTrafo_Chosen2,Topoil_Chosen2,'-.');
str=['día n°',num2str(Day_plot2)];
title(str);
str2=['Potencia KVA * ',num2str(Ajuste_grafica/1000)];
ylabel(str2)
legend('S','Hotspot','Topoil')

%FIN ANÁLISIS DE DATOS

HSM=max(Resultados(:,3));
TOM=max(Resultados(:,4));
set(handles.Result_Pn,'String',Pn/1000000)
set(handles.Result_Hotspot,'String',HSM)
set(handles.Result_Topoil,'String',TOM)
set(handles.Result_Ageing,'String',Loss_life_days)

% --- Executes on selection change in Reactiva.
function Reactiva_Callback(hObject, eventdata, handles)
% hObject    handle to Reactiva (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns Reactiva
contents as cell array
%         contents{get(hObject,'Value')} returns selected item from
Reactiva

% --- Executes during object creation, after setting all properties.
function Reactiva_CreateFcn(hObject, eventdata, handles)
% hObject    handle to Reactiva (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: popupmenu controls usually have a white background on Windows.
%         See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUiControlBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function INC_TITAto_R_Callback(hObject, eventdata, handles)
% hObject    handle to INC_TITAto_R (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of INC_TITAto_R as
text
%         str2double(get(hObject,'String')) returns contents of
INC_TITAto_R as a double

```

```

% --- Executes during object creation, after setting all properties.
function INC_TITAto_R_CreateFcn(hObject, eventdata, handles)
% hObject    handle to INC_TITAto_R (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUiControlBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function R_Callback(hObject, eventdata, handles)
% hObject    handle to R (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of R as text
%       str2double(get(hObject,'String')) returns contents of R as a
double

% --- Executes during object creation, after setting all properties.
function R_CreateFcn(hObject, eventdata, handles)
% hObject    handle to R (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUiControlBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function tao_to_R_Callback(hObject, eventdata, handles)
% hObject    handle to tao_to_R (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of tao_to_R as text
%       str2double(get(hObject,'String')) returns contents of
tao_to_R as a double

% --- Executes during object creation, after setting all properties.
function tao_to_R_CreateFcn(hObject, eventdata, handles)
% hObject    handle to tao_to_R (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB

```

```

% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function INC_TITAah_R_Callback(hObject, eventdata, handles)
% hObject    handle to INC_TITAah_R (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of INC_TITAah_R as
text
%       str2double(get(hObject,'String')) returns contents of
INC_TITAah_R as a double

% --- Executes during object creation, after setting all properties.
function INC_TITAah_R_CreateFcn(hObject, eventdata, handles)
% hObject    handle to INC_TITAah_R (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function t_Callback(hObject, eventdata, handles)
% hObject    handle to t (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of t as text
%       str2double(get(hObject,'String')) returns contents of t as a
double

% --- Executes during object creation, after setting all properties.
function t_CreateFcn(hObject, eventdata, handles)
% hObject    handle to t (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.

```

```

if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function tao_w_Callback(hObject, eventdata, handles)
% hObject    handle to tao_w (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of tao_w as text
%        str2double(get(hObject,'String')) returns contents of tao_w
as a double

% --- Executes during object creation, after setting all properties.
function tao_w_CreateFcn(hObject, eventdata, handles)
% hObject    handle to tao_w (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%        See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

% --- Executes on selection change in Trafo_americano.
function Trafo_americano_Callback(hObject, eventdata, handles)
% hObject    handle to Trafo_americano (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns
Trafo_americano contents as cell array
%        contents{get(hObject,'Value')} returns selected item from
Trafo_americano

% --- Executes during object creation, after setting all properties.
function Trafo_americano_CreateFcn(hObject, eventdata, handles)
% hObject    handle to Trafo_americano (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: popupmenu controls usually have a white background on Windows.
%        See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

```



```

function Pn_Contrato_Callback(hObject, eventdata, handles)
% hObject    handle to Pn_Contrato (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of Pn_Contrato as text
%        str2double(get(hObject,'String')) returns contents of
Pn_Contrato as a double

% --- Executes during object creation, after setting all properties.
function Pn_Contrato_CreateFcn(hObject, eventdata, handles)
% hObject    handle to Pn_Contrato (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

```

ANEXO 2: FUNCIONES DEL PROGRAMA

```
function [f1] = f1(t,k11,tao0)
```

```
f1=1-exp((-t)/(k11*tao0));  
end
```

```
function [f2] = f2(t,k22,tao0,taoW,k21)  
f2=k21*(1-exp((-t)/(k22*taoW)))-(k21-1)*(1-exp((-t)/(tao0/k22)));  
end
```

```
function [f3] = f3(t,tao0,k11)  
f3=exp((-t)/(k11*tao0));  
end
```

```
function [INC_TITAh] = INC_TITAh(INC_TITAh_u,INC_TITAh_i,tao_w,t)  
INC_TITAh=(INC_TITAh_u-INC_TITAh_i)*(1-exp(-t/tao_w))+INC_TITAh_i;  
end
```

```
function [INC_TITAh_i] = INC_TITAh_i(INC_TITAh_R,K_i,m)  
INC_TITAh_i=INC_TITAh_R*K_i^(2*m);  
end
```

```
function [INC_TITAh_R] = INC_TITAh_R(INC_TITAh_a_R,INC_TITAh_o_R)  
INC_TITAh_R=INC_TITAh_a_R-INC_TITAh_o_R;  
end
```

```
function [INC_TITAh_u] = INC_TITAh_u(INC_TITAh_R,K_u,m)  
INC_TITAh_u=INC_TITAh_R*K_u^(2*m);  
end
```

```
function [INC_TITAh_i_decrease] = INC_TITAh_i_decrease(H,Gr,K,y)  
INC_TITAh_i_decrease=H*Gr*(K^y);  
end
```

```
function [INC_TITAh_i_increase] =  
INC_TITAh_i_increase(INC_TITAh_i,H,Gr,K,y,f2)  
INC_TITAh_i_increase=INC_TITAh_i+(H*Gr*(K^y)-INC_TITAh_i)*f2;  
end
```

```
function [INC_TITAh_i_initial] = INC_TITAh_i_initial(k21,INC_TITAh_r,K,y)  
INC_TITAh_i_initial=(k21*INC_TITAh_r*K^y)-((k21-1)*INC_TITAh_r*K^y);  
end
```

```

function [INC_TITAO] = INC_TITAO(INC_TITAto_u,INC_TITAto_i,tao_to)
INC_TITAO=(INC_TITAto_u-INC_TITAto_i)*(1-exp(-1/tao_to))+INC_TITAto_i;
end

```

```

function [INC_TITAOi_decrease] =
INC_TITAOi_decrease(INC_TITAOi,INC_TITAOor,R,K,x,f3)
INC_TITAOi_decrease=INC_TITAOor*((1+R*K^2)/(1+R))^x+(INC_TITAOi-
INC_TITAOor*((1+R*K^2)/(1+R))^x)*f3;
end

```

```

function [INC_TITAOi_increase] =
INC_TITAOi_increase(INC_TITAOi,INC_TITAOor,R,K,x,f1)
INC_TITAOi_increase=INC_TITAOi+(INC_TITAOor*((1+R*K^2)/(1+R))^x -
INC_TITAOi)*f1;
end

```

```

function [INC_TITAOi_initial] =
INC_TITAOi_initial(INC_TITAOor,R,K,Ta,x)
INC_TITAOi_initial=Ta+INC_TITAOor*((1+R*K^2)/(1+R))^x;
end

```

```

function [INC_TITAto_i] = INC_TITAto_i(INC_TITAto_R,K_i,n,R)
INC_TITAto_i=INC_TITAto_R*((K_i^2*R+1)/(R+1))^n;
end

```

```

function [INC_TITAto_u] = INC_TITAto_u(INC_TITAto_R,K_u,n,R)
INC_TITAto_u=INC_TITAto_R*(K_u^2*R+1/(R+1))^n;
end

```

```

function [V] = relative_ageing(Hotspot)
V=exp((15000/(110+273))-(15000/(Hotspot+273)));
end

```

```

function [tao_to] =
tao_to(INC_TITAto_u,INC_TITAto_i,INC_TITAto_R,n,tao_to_R)
tao_to=tao_to_R*((INC_TITAto_u/INC_TITAto_R)-
(INC_TITAto_i/INC_TITAto_R))/((INC_TITAto_u/INC_TITAto_R)^(1/n)-
(INC_TITAto_i/INC_TITAto_R)^(1/n));
end

```

```

function [tao_to_R] = tao_to_R(C,INC_TITAto_R,P_T_R)
tao_to_R=(C*INC_TITAto_R)/P_T_R;
end

```

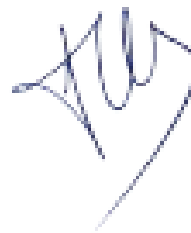
ANEXO 3: VALORACIÓN FINAL DE LA EMPRESA IBÉRICA SOLAR

D. Enrique Puente Lázaro, CTO de la empresa Ibérica Solar,

HACE CONSTAR:

que D. Daniel Caloca Mendo, estudiante del grado de Ingeniería Eléctrica, de la Escuela de Ingenierías Industriales de la Universidad de Valladolid, que realiza el Trabajo Fin de Grado titulado "*Potencia óptima de los transformadores asociados a plantas de energía solar fotovoltaica*", propuesto por nuestra empresa al Departamento de Ingeniería Eléctrica de dicha Universidad, ha realizado esta tarea durante el presente curso con gran dedicación y capacidad de trabajo, de modo muy satisfactorio y acorde con los objetivos planteados.

Y para que pueda acreditarlo donde proceda, se expide el presente testimonio en Valladolid, a veintiocho de junio de dos mil diecinueve.



Fdo.: D. Enrique Puente Lázaro

IBÉRICA SOLAR P.E.R., S
CIF: B-47568142
Avda. Miguel Ángel Blanco, 3, 1-A
47014 VALLADOLID - SPAIN
Tel: +34 983 37 86 73
info@ibericasolar.es