

ENTORNO GRÁFICO PARA ALGORITMOS DE ESTIMACIÓN DE TEMPERATURA EN LA CIUDAD DE VILLA MERCEDES, SAN LUIS SOBRE LA BASE DE PUNTOS DE TEMPERATURAS CONOCIDOS.

J. Demichelis¹, J. Carletto², P. Gimeno³

Laboratorio de Energías Alternativas – Universidad Nacional de San Luis
Facultad de Ingeniería y Ciencias Agropecuarias
Avda. 25 de Mayo 384 - 5730 - Villa Mercedes - San Luis - Argentina
Tel - Fax: 054 2657 531000 - e-mail: jpdemichelis@gmail.com

Recibido 10/08/16, aceptado 08/10/16

RESUMEN: El objetivo de este trabajo es el diseño e implementación de un entorno gráfico para un algoritmo de estimación de temperaturas sobre la ciudad de Villa Mercedes, San Luis desarrollado con anterioridad. Sobre la base de datos conocidos de cinco centrales meteorológicas estratégicamente ubicadas, el algoritmo entrega una superficie tridimensional de temperatura sobre la ciudad que permite observar la variación y además entrega las isotermas sobre el mapa en función de una fecha solicitada. Este nuevo entorno gráfico, constituye una mejora desde el punto de vista de la usabilidad, no solo porque transforma el algoritmo dotándolo de una interfaz intuitiva y simple, sino también porque lo hace independiente del software utilizado para el desarrollo (Matlab), mejorando la portabilidad de la aplicación. Se presenta, el principio de funcionamiento del algoritmo, el entorno desarrollado, las distintas posibilidades que brinda, como así también los pasos para la distribución e instalación del software desarrollado.

Palabras clave: algoritmo, estimación de temperaturas, diferencia de temperaturas.

INTRODUCCION

En un trabajo realizado con anterioridad y publicado por Demichelis et. al 2015, se intentó solucionar mediante un algoritmo desarrollado en Matlab, las dudas que surgen al observar los valores de temperatura en la ciudad de Villa Mercedes, San Luis, publicados por distintos servicios de clima on line, donde se observan siempre diferencias entre ellos.

MATLAB® es el lenguaje de alto nivel y el entorno interactivo utilizado por millones de ingenieros y científicos en todo el mundo (MathWorks 2015). En nuestro caso en particular esta herramienta es utilizada para la sistematización de los datos y para lograr una rápida y fácil visualización de la curva de temperatura sobre la ciudad de Villa Mercedes, ahora, con un entorno operativo moderno, e independiente del lenguaje de programación.

Si bien se logró el objetivo planteado realizando una estimación interesante para los puntos no conocidos y una forma de ver la variación de temperatura sobre la ciudad como una superficie en un sistema de coordenadas XYZ, era requerida la utilización de Matlab para correr el algoritmo. Esto potenciaba enormemente la aplicación permitiendo al usuario hacer uso de otras herramientas del software, pero bajaba la usabilidad de la aplicación, debido a que el usuario necesitaba tener conocimientos específicos de MATLAB y además y requería tener instalado en la computadora que se utilizar el mencionado software.

¹ Investigador FICA UNSL

² Investigador CyT UNSL

³ Investigador CyT UNSL

La usabilidad es una característica que mide qué tan intuitiva y fácil de usar es una página web o una aplicación informática para el usuario común. (Nielsen 2000)

Nielsen también aclara en su escrito, los tres grandes aspectos que miden la usabilidad, Eficacia, Eficiencia y Satisfacción. Si bien el algoritmo desarrollado cumplía los dos primeros aspectos, la satisfacción del usuario podía mejorarse si se mejoraba el entorno gráfico, y más aún, si se transformaban los scripts escritos en Matlab, en una aplicación independiente.

La plataforma de MATLAB GUIDE permite realizar tareas de forma interactiva a través de controles, como botones y barras de desplazamiento entre otros; los cuales ayudan a una mejor interacción entre el entorno gráfico desarrollado (GUI) y el usuario. Para obtener más control sobre el diseño y el desarrollo de una GUI, se crea un código de MATLAB que define todas las propiedades de los componentes y comportamientos; que finalmente registrarán los resultados de los cálculos del software. (Quintero Castillo et. al 2013). El nuevo algoritmo, encapsulado como una aplicación independiente y con un entorno gráfico propio, se logró utilizando esta plataforma de Matlab.

La aplicación final, logra determinar la superficie de temperatura sobre la ciudad de villa mercedes, sobre la base de 5 centrales meteorológicas conocidas, y dispone de una base de datos de temperaturas desde 01-01-2010 hasta 30-06-2016

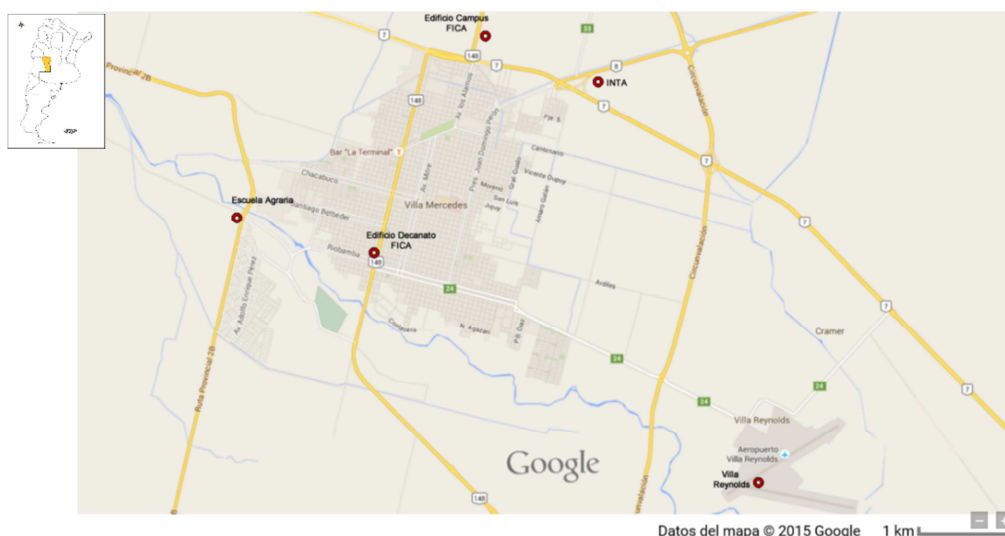


Figura 1. Ubicación geográfica de las centrales meteorológicas.

Las estaciones conocidas son:

La estación perteneciente al LEA - Laboratorio de Energías Alternativas de la FICA - Facultad de Ingeniería y Ciencias Agropecuarias de la UNSL - Universidad Nacional de San Luis, está ubicada en el Campus Universitario, localizado sobre Ruta Provincial N° 55, (Ex ruta 148) extremo norte (-33.644208, -65.447200).

Una segunda central de la misma institución situada en edificio del decanato de la facultad, el cual se localiza en pleno centro de la ciudad de Villa Mercedes (-33.684907, -65.471996).

La estación localizada en la Escuela Agraria perteneciente a la REM – Red de estaciones meteorológicas de la ULP – Universidad de la Punta (-33.678512, -65.504252).

La estación agrometeorológica perteneciente al INTA – Instituto Nacional de Tecnología Agropecuaria - San Luis (-33.652623, -65.419220).

La estación meteorológica instalada en Villa Reynolds, perteneciente al CIM - Centro de Investigación Meteorológica del Servicio Meteorológico Nacional (-33.728713, -65.381104).

La estrategia algorítmica planteada por Demichelis et. al. 2015 permitió estimar la temperatura en todos los puntos de la ciudad, y este trabajo permite mejorar dicha aplicación.

RESUMEN DEL ALGORITMO DE ESTIMACIÓN DE TEMPERATURAS

El algoritmo fue explicado en detalle por Demichelis et. al. 2015, y si bien no es objeto de este trabajo ahondar en el algoritmo en sí, su funcionamiento se resume en este apartado. Se traza una grilla sobre la ciudad de Villa Mercedes.

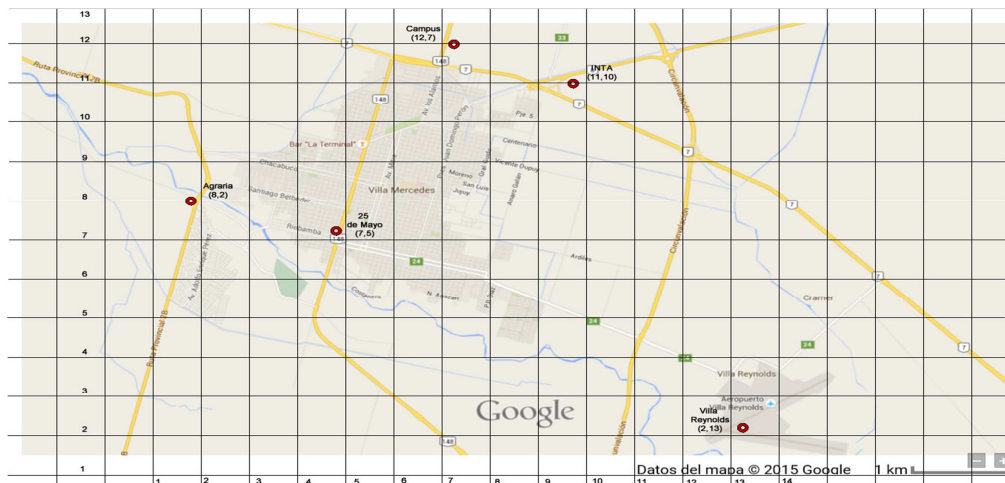


Figura 2. Grilla sobre la ciudad de Villa Mercedes para la estimación de temperaturas.

Considerando las temperaturas conocidas (centrales) en las intersecciones de las grillas, se tiene que las coordenadas de las centrales (Tabla 1) son:

Identificación	Ubicación	Coordenada X	Coordenada Y
T1	Escuela Agraria	8	2
T2	Edificio Decanato FICA	7	5
T3	Villa Reynolds	2	13
T4	INTA	11	10
T5	Edificio Campus FICA	12	7

Tabla 1: Coordenadas de referencia de las centrales meteorológicas.

Cabe destacar que T1 a T5 serán las temperaturas promedios diarias para el día seleccionado. Lo que puede graficarse mediante la (Figura 3) que representa la siguiente matriz:

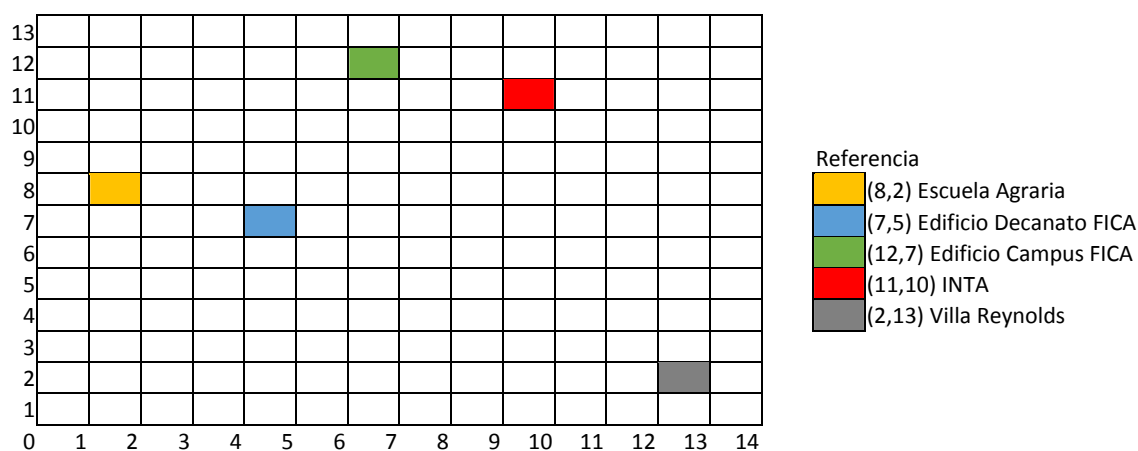


Figura 3. Matriz de temperatura, con puntos de referencia.

Una vez establecida la matriz, se diseña un algoritmo con el fin de estimar las temperaturas en los puntos no conocidos (cuadros vacíos de la Figura 3). Este algoritmo calcula una primera aproximación en función del promedio ponderado según la distancia de las distintas centrales al punto de interés, y

luego se utiliza una segunda estrategia que consiste en calcular la temperatura de un punto de la grilla (i,j) como el promedio de todos los puntos que lo rodean y como una pequeña heurística, se itera calculando este promedio de promedios repetidas veces hasta que se llega al equilibrio. Los detalles de las fórmulas utilizadas, pueden verse en Demichelis et. al. 2015, las cuales no se repiten por no ser el objetivo de este trabajo. Las temperaturas obtenidas de esta manera dan una curva suave dónde se aproximan satisfactoriamente las temperaturas reales, según análisis realizados para distintos valores de temperaturas conocidos.

METODOLOGÍA Y ESTRATEGIAS PARA LA DEFINICIÓN DEL ENTORNO GRAFICO

Para el desarrollo del entorno gráfico se utilizó la herramienta para generación de interfaces de usuario (GUI) que MATLAB ofrece. Esta consta de dos archivos uno “.m” (ejecutable) y otra fig. (parte gráfica). Las dos partes están unidas a través de las subrutinas callback. Una vez que se graban los archivos desde la consola de emisión (si salvamos la .fig automáticamente lo hace el .m asociado) podemos ejecutar el programa en la ventana de comando de Matlab escribiendo el nombre del archivo solamente.

Para ello se ejecutó el comando GUIDE desde el entorno de desarrollo de MATLAB. Tal como se puede ver en la Figura 4.

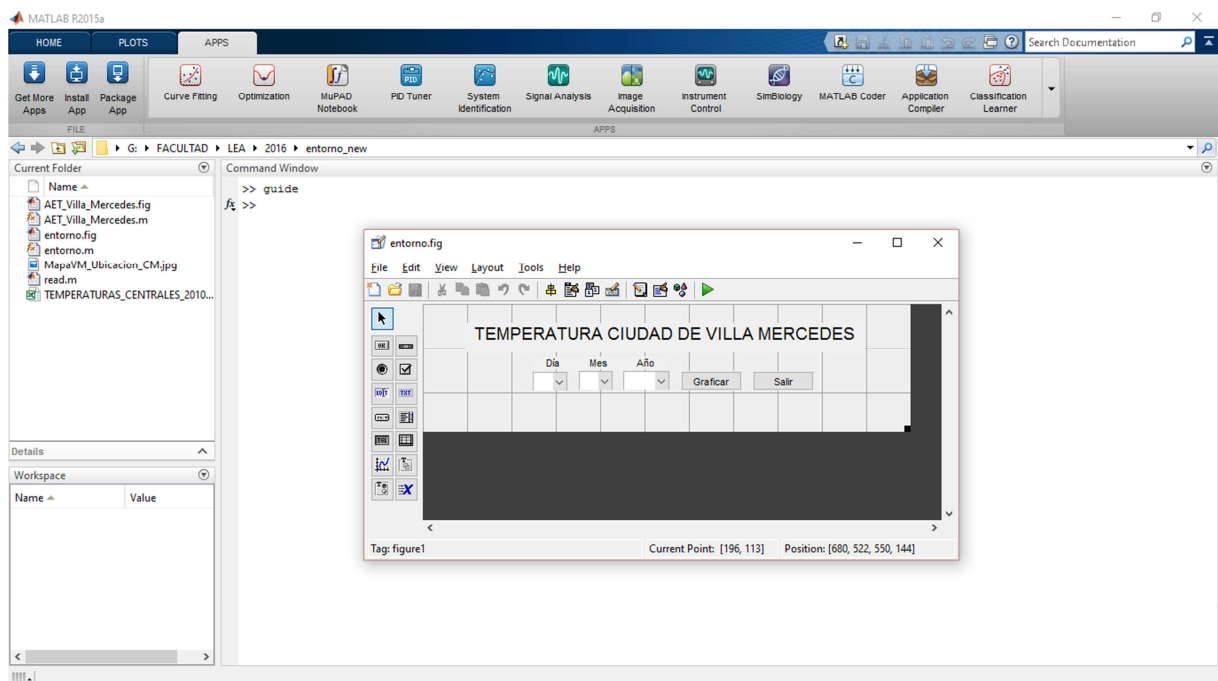


Figura 4. Entorno de generación de interface de usuario (GUI) de MATLAB.

En la misma se puede observar que se hizo uso de los componentes habituales que la GUI ofrece, tales como:

- PopupMenu** (para nuestro caso se utilizan para la selección del Día, Mes y Año)
- Pushbuttons** (se usó para establecer el botón **Graficar**; que permite realizar la gráfica del día seleccionado y el botón **Salir**; que cierra el programa)
- Text** (textos que permiten una ayuda visual al usuario)

Una vez terminado el desarrollo del entorno visual, se procede a la configuración de las funciones que cada uno de los mismos desarrollará. Para ello se procede a hacer uso de los “**Callbacks**”, que son las funciones que realizan las acciones requeridas cuando un componente se activa.

El archivo .m que se crea tiene una estructura predeterminada. Consta de un encabezado y a continuación el código correspondiente a las siguientes subrutinas.

Se detallan a continuación las secciones más relevantes: El encabezado tiene un formato estándar y se puede observar a continuación:

-

```

function varargout = entorno(varargin)
% ENTORNO MATLAB code for entorno.fig
% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                  'gui_Singleton',  gui_Singleton, ...
                  'gui_OpeningFcn', @entorno_OpeningFcn, ...
                  'gui_OutputFcn',  @entorno_OutputFcn, ...
                  'gui_LayoutFcn',  [] , ...
                  'gui_Callback',    []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

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

```

- Además, dentro del encabezado, se definen variables generales para el correcto funcionamiento del programa.

handles: puntero a todas variables y elementos de la aplicación.

guidata(): comando para guardar las variables de la aplicación.

```

% --- Executes just before entorno is made visible.
function entorno_OpeningFcn(hObject, eventdata, handles, varargin)

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

% Update handles structure
guidata(hObject, handles);

```

- La primera subrutina, es la que permite la selección de la fecha a graficar. La misma está dividida en tres subrutinas selección, del **Día**, del **Mes** y del **Año**. Por simplicidad se muestra el código de la subrutina del **Día**, siendo las otras dos similares.

La subrutina `function popupmenu1` hace referencia al nombre del desplegable. Dentro de la misma, a través de la función **get**, en primera instancia se selecciona toda la cadena de valores del menú, para luego con la misma función determinar la posición del valor seleccionado; de esta manera cruzando las variables se obtiene el valor seleccionado; representado por la variable “**d**” (número de día a graficar).

```

% --- Executes on selection change in popupmenu1.
function popupmenu1_Callback(hObject, eventdata, handles)

items = get(hObject, 'String');
index_selected = get(hObject, 'Value');
d = items{index_selected};

```

- Luego están las subrutinas de los botones, en primer lugar mencionamos el botón que realiza la acción de procesar y mostrar la gráfica de la superficie de la ciudad en la fecha seleccionada anteriormente. Para ello se genera la función `function pushbutton2`. Como puede observarse el código es similar a la de la sección anterior con la diferencia que cuando se usa la función **get**, el apuntador se direcciona al nombre de la función que contiene el valor a extraer; particularmente para el día se hace referencia a `handles.popupmenu1`.

Por último se convierte dicho valor a numérico, ya que en un principio el mismo es un “string” (texto); a través de la función `str2num` de MATLAB; esto queda representado por la variable “`d`” (número de día a graficar).

Lo mismo se hace para los valores del Mes y Año.

```
% --- Executes on button press in pushbutton2.
function pushbutton2_Callback(hObject, eventdata, handles)

%TOMO EL VALOR DEL DIA SELECCIONADO
items_d=get(handles.popupmenu1,'String');
index_selected_d=get(handles.popupmenu1,'Value');
str_d=items_d{index_selected_d};
d=str2num(str_d);
```

Por último se hacen uso de las variables creadas para introducirlas al código que permite el procesamiento y posterior gráfica de la fecha seleccionada.

Una vez depurado el código se procede a la generación de la aplicación, para ello MATLAB nos ofrece la opción “**Application Compiler**”, tal como se observa en la Figura 5.

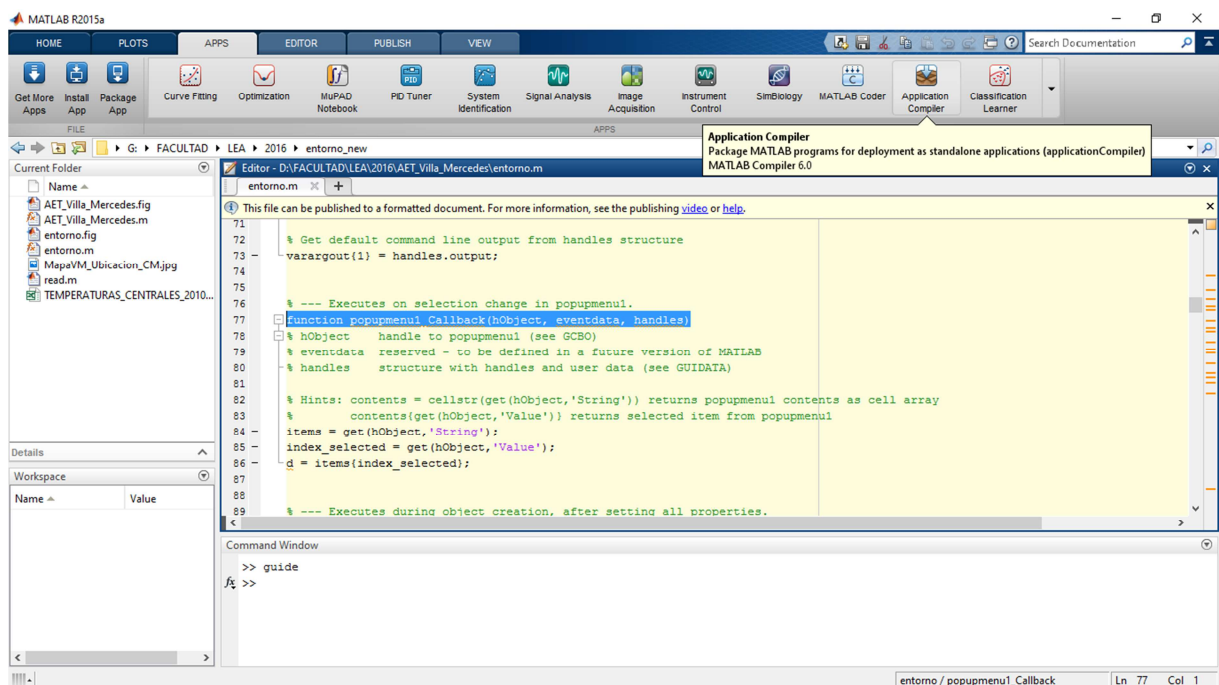


Figura 5. Selección de la opción “**Application Compiler**” de MATLAB.

Cuando la misma se ejecuta, automáticamente nos abre una nueva ventana Figura 6 que nos permite armar nuestro programa empaquetado.

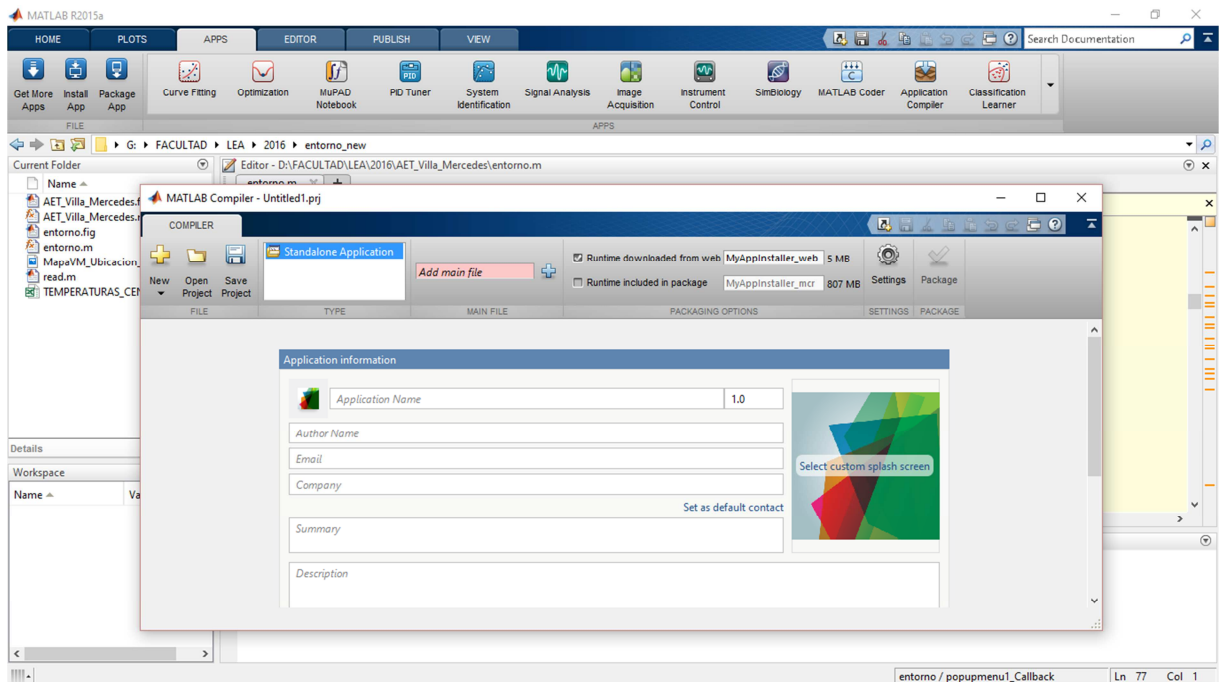


Figura 6. Entorno “Application Compiler” de MATLAB.

Para nuestro caso particular se seleccionaron los archivos previamente desarrollados y se generó una imagen de presentación para el programa, tal como se ve en la Figura 7.

Una vez que se completaron todos los datos solicitados se procede a hacer clic en el botón **Package**, el cual realiza el empaquetado del programa brindándonos un archivo ejecutable.

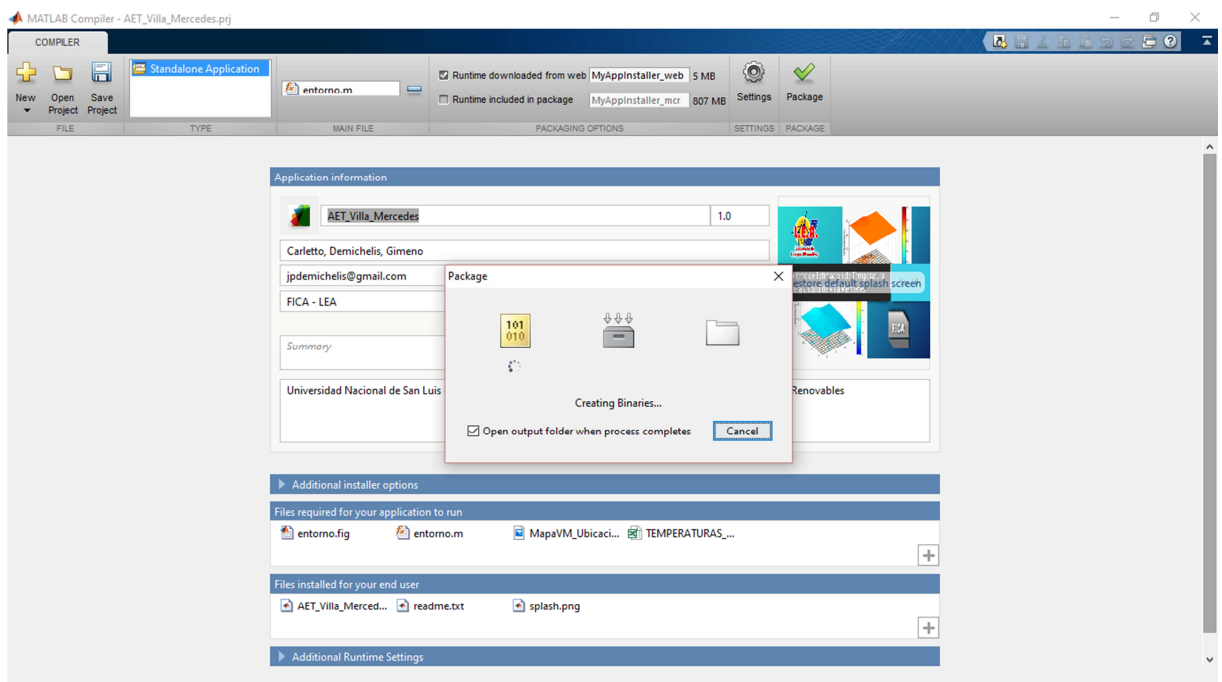


Figura 7. Creación del archivo ejecutable del programa desarrollado.

Terminado dicho proceso, podemos ver que MATLAB generará en la carpeta seleccionada anteriormente un archivo ejecutable de nuestro programa. Esto queda representado en la Figura 8, donde se observa el fichero empaquetado (AET_Villa_Mercedes.exe)

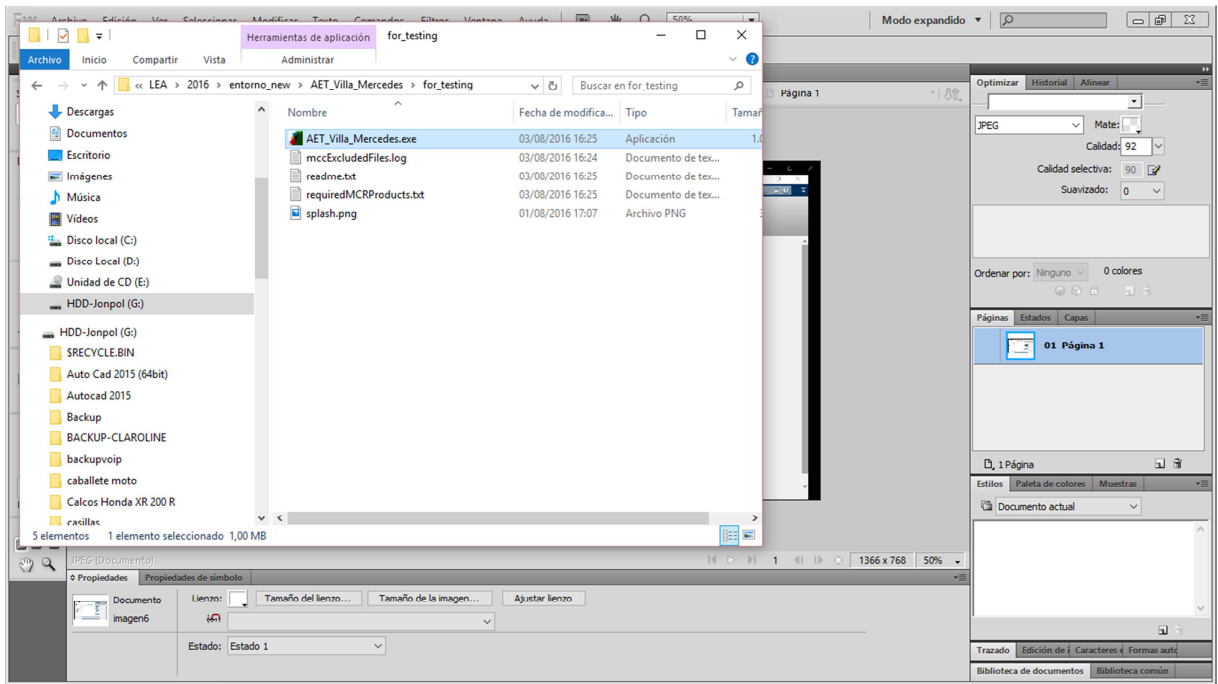


Figura 8. Archivo ejecutable del programa desarrollado.

La Figura 9 muestra la pantalla de presentación del Programa, donde se visualiza el nombre que se le dio a la aplicación “Algoritmo de Estimación de Temperatura sobre la ciudad de Villa Mercedes”, además se detalla la versión del mismo (ver 1.0).

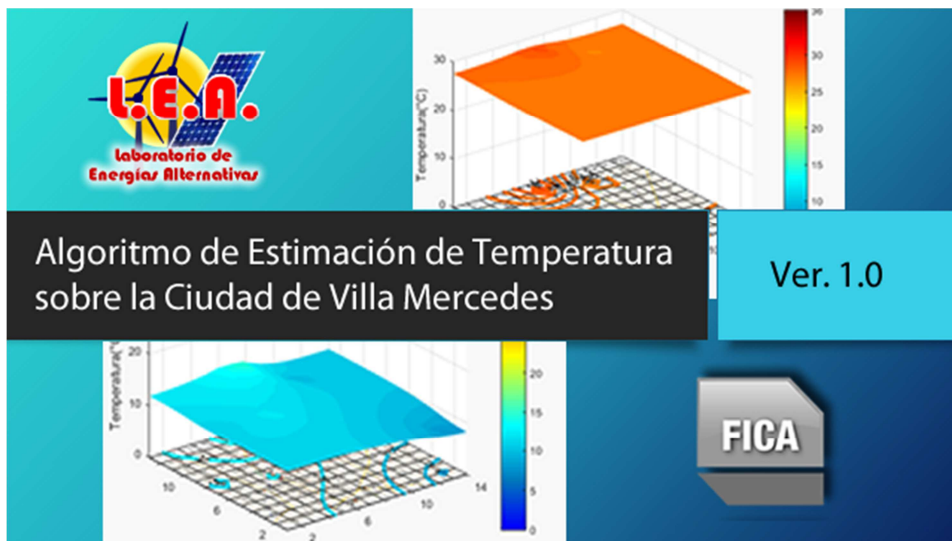


Figura 9. Imagen de presentación del programa AET_Villa_Mercedes.

El entorno del programa es muy simple e intuitivo y se puede observar en la Figura 10. La ventaja principal que ofrece esto es que nos independizamos de MATLAB, permitiéndonos correr el programa de manera independiente.



Figura 10. Entorno del programa desarrollado.

A continuación en la Figura 11 se muestran varias pantallas del programa ejecutándose. Cabe destacar una capacidad muy importante que este permite, pudiendo graficar simultáneamente varias gráficas de la temperatura sobre la ciudad de Villa Mercedes en distintos días.

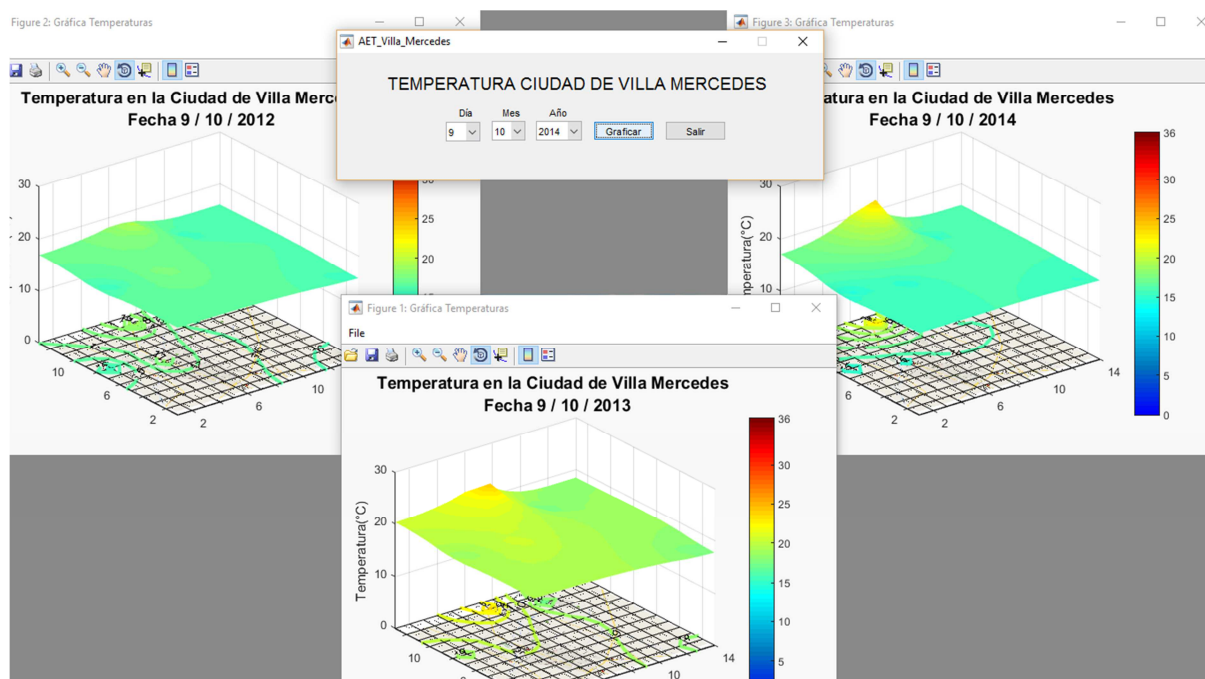


Figura 11. Representación de la curva de temperatura para distintos días seleccionados.

CONCLUSIONES

Se logró el objetivo planteado realizando un entorno gráfico que mejora la usabilidad de la aplicación, y a su vez la independiza de MATLAB.

Se puede realizar una distribución de la aplicación en forma sencilla, y cualquier persona independientemente de sus conocimientos sobre MATLAB, podrá utilizar la aplicación.

Frente a cualquier actualización de la aplicación, será necesario la recompilación de la aplicación para su actualización y posterior distribución, esto constituye en cierta forma una debilidad para aquellos usuarios que tienen conocimientos sobre MATLAB. Pero una simplificación para aquellos que no lo tienen.

En términos generales, se puede concluir que ahora se dispone de una herramienta, que puede utilizar tanto el usuario con conocimientos de MATLAB, como aquellos que no lo tienen.

Si bien el error cometido no es importante desde el punto de vista que se pretende conocer una aproximación de temperatura en los puntos no conocidos, este desarrollo facilita la manipulación del algoritmo, lo que permitirá aplicar la herramienta con facilidad de forma tal de poder mejorarlo e incluso poder cuantificar y mejorar el error que se comete en los puntos no conocidos, lo que será objeto de futuros trabajos.

REFERENCIAS

Quintero Castillo A., Acevedo Pérez J., Uribe Pérez I., Aguirre Rodríguez L. (2013) - Diseño de un software para del riesgo basado en las el análisis y evaluación normas api 580 y 581 - Revista: Fuentes El Reventón Energético-Vol. 11 No 2 de 2013 - Jul/Dic - pp 71/77 - (<http://revistas.uis.edu.co/index.php/revistafuentes/article/view/3833/4192>)

Nielsen J. (2000) - Usabilidad. Diseño de páginas Web. Prentice Hall .

Demichelis J., Carletto J., Gimeno P. - (2015) - Algoritmos de estimación de temperatura en la ciudad de Villa Mercedes, San Luis sobre la base de puntos de temperaturas conocidos. Estrategias algorítmicas y posibilidades de análisis. - ASADES 2015.

Demichelis J., Carletto J., Gimeno P. - (2014) - Diferencia de temperatura en los extremos de la ciudad de Villa Mercedes - San Luis - ASADES 2014.

ABSTRACT

The objective of this work is to design and implement a graphical algorithm to estimate the temperatures over the city of Villa Mercedes. Previously we developed an algorithm, the challenge here is the graphical environment to simplify the use of it. We used the data from five weather stations strategically located; the algorithm delivers a three-dimensional surface temperature over the city which allows the observation of the variation and also delivers the isotherms on the map based on a requested date. This new graphical environment, is an improvement from the point of view of usability, not only because it transforms the algorithm giving it an intuitive and simple interface, but also because it makes it independent of the software used for the development (Matlab), improving portability of the application. The operating principle of the algorithm, the environment developed, the various possibilities offered are presented as well as the steps for the distribution and installation of software developed.

Keywords: algorithm, temperatures estimation, temperature difference.