Herramientas de simulación libres y abiertas para el diseño de unidades de procesamiento de potencia para sistemas fotovoltaicos

Free and open source simulation tools for the design of power processing units for photovoltaic systems

Sergio Morales-Hernández[1], Carlos Meza-Benavides[2]

Fecha de recepción: 18 de setiembre del 2014
Fecha de aprobación: 10 de diciembre del 2014

Morales-Hernández, S; Meza-Benavides, C. Herramientas de simulación libres y abiertas para el diseño de unidades de procesamiento de potencia para sistemas fotovoltaicos. *Tecnología en Marcha*. Vol. 28, Nº 2, Abril-Junio. Pág 44-60.

1 Profesor Adjunto, Escuela de Ingeniería en Electrónica, Instituto Tecnológico de Costa Rica
Correo electrónico: smorales@itcr.ac.cr
2 Profesor Adjunto, Escuela de Ingeniería en Electrónica, Instituto Tecnológico de Costa Rica
Correo electrónico: cmeza@itcr.ac.cr

Palabras clave

Sistemas fotovoltaicos; herramientas de simulación; convertidores de potencia.

Resumen

Las fuentes de energía renovable, entre ellas la solar fotovoltaica, requieren de circuitos electrónicos que sirvan de interface entre el dispositivo transductor y el dispositivo o sistema que utilizará la energía. Más aún, la eficiencia energética y el costo del sistema se pueden ver comprometidos si este circuito electrónico no ha sido diseñado de forma apropiada. Dado que, por un lado, las características eléctricas de los dispositivos fotovoltaicos son no lineales y, por otro, los circuitos electrónicos más eficientes son naturalmente discontinuos, se requiere un análisis dinámico detallado que permita optimizar el diseño. Dicho análisis debe ser apoyado por herramientas informáticas de simulación. En este artículo se hace una comparación entre dos herramientas informáticas de simulación de sistemas dinámicos para determinar su utilidad en el proceso de diseño de sistemas fotovoltaicos, principalmente en lo que corresponde a las unidades de procesamiento de potencia. Utilizando como caso de estudio un sistema fotovoltaico para la carga de una batería, se determinó que la herramienta Scicoslab facilita la simulación de sistemas dinámicos complejos, presentes en las unidades de procesamiento de potencia de los sistemas fotovoltaicos.

Keywords

Photovoltaic systems; simulation tools; power converters.

Abstract

Renewable energy sources, including solar photovoltaic, require electronic circuits that serve as interface between the transducer device and the device or system that uses energy. Moreover, the energy efficiency and the cost of the system can be compromised if such electronic circuit is not designed properly. Given that the electrical characteristics of the photovoltaic devices are nonlinear and that the most efficient electronic circuits for power processing are naturally discontinuous, a detailed dynamic analysis to optimize the design is required. This analysis should be supported by computer simulation tools. In this paper a comparison between two software tools for dynamic system simulation is performed to determinate its usefulness in the design process of photovoltaic systems, mainly in what corresponds to the power processing units. Using as a case of study a photovoltaic system for battery charging it was determined that Scicoslab tool was the most suitable.

Introducción

La energía solar es abundante y el proceso de conversión a energía eléctrica puede llegar a tener un impacto ambiental muy bajo. Los sistemas solares fotovoltaicos permiten el aprovechamiento de la radiación electromagnética producida por el sol por medios eléctricos. La tecnología fotovoltaica es hoy en día bastante madura y en muchas regiones del mundo su costo es comparable con el de las fuentes de energía tradicional.

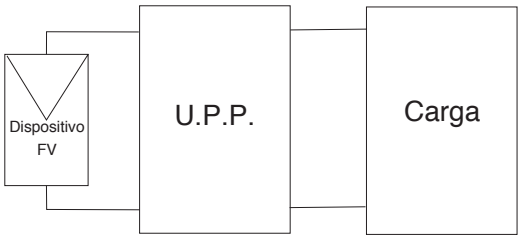

Figura 1. Diagrama de bloques representativo de un sistema fotovoltaico (U.P.P.: Unidad de Procesamiento de Potencia).

Un sistema fotovoltaico está conformado por uno o varios dispositivos fotovoltaicos (celda, panel o conjunto de paneles) y una unidad de procesamiento de potencia (ver figura 1). Las características eléctricas de un dispositivo fotovoltaico hacen necesario el uso de un circuito electrónico, denominado unidad de procesamiento de potencia (UPP), para aprovechar de forma óptima la energía que éste genera. Sin la UPP, el dispositivo fotovoltaico generaría menos energía de la que potencialmente puede producir. Esto se debe a que un dispositivo fotovoltaico presenta un punto de máxima generación de potencia definido por un par de valores de tensión y corriente determinados, tal y como se puede apreciar en la figura 2. Una de las tareas de la UPP es posicionar en todo momento el punto de operación del dispositivo fotovoltaico en su punto de máxima potencia. La otra tarea de la unidad de procesado de potencia es lograr que la carga que está siendo alimentada por el dispositivo fotovoltaico reciba un nivel de tensión y corriente adecuado (ver figura 1). Cargas típicas en sistemas fotovoltaicos son baterías, motores, luminarias o la misma red eléctrica.

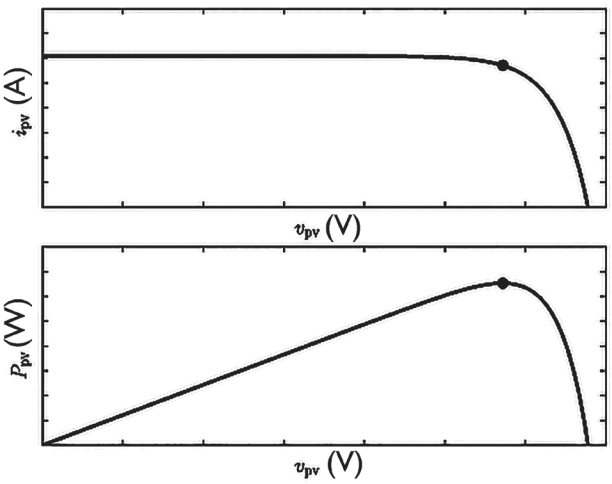

Figura 2. Curvas de voltaje vs. corriente y voltaje vs. potencia de un dispositivo fotovoltaico típico a temperatura e insolación constantes. El círculo rojo indica el punto de máxima generación de potencia.

Como se puede deducir de la figura 2, las características eléctricas de un dispositivo fotovoltaico son no lineales y en algunos casos la unidad de procesamiento de potencia también puede tener características eléctricas no solo no lineales sino también discontinuas. Esta situación requiere de un análisis detallado del sistema que, dada su complejidad, debe ser apoyado

por simulaciones numéricas. El uso de simulaciones numéricas permite realizar experimentos virtuales para probar potenciales soluciones y sistemas, reduciendo significativamente el tiempo de diseño y los errores en la implementación final.

El presente artículo tiene como objetivo identificar un conjunto de herramientas de simulación que pueden apoyar el proceso de diseño de sistemas fotovoltaicos, con un especial enfoque en el diseño de las unidades de procesamiento de potencia. Como se comentará más adelante, enfocarse en el análisis del comportamiento eléctrico de las UPP implica estudiar los transitorios del sistema que normalmente tienen una duración en el orden de los milisegundos. En consecuencia, las herramientas de simulación que se desean identificar no son aquellas que se utilizan para determinar el desempeño de un sistema fotovoltaico en períodos de meses o años, en estos casos ya hay varias herramientas maduras que logran obtener excelentes resultados (Turcotte et al., 2001). Por otro lado, este documento se concentra únicamente en herramientas que sean de acceso libre y abierto, no solo porque representan una opción más económica que las pocas alternativas comerciales existentes, sino porque, dado el carácter particular y reciente del sistema que se quiere estudiar, es necesario contar con herramientas lo suficientemente flexibles que permitan la modificación y alteración de algunas de sus rutinas y modelos.

El presente artículo está estructurado de la siguiente forma: primero se presentan las características eléctricas de los principales sistemas que se quieren simular, esto es, el generador fotovoltaico y la unidad de procesamiento de potencia. A partir de la caracterización de dichos sistemas se obtiene un conjunto de requerimientos que la herramienta de simulación de *software* debe cumplir. Luego, se presenta un conjunto de herramientas de *software* que cumplen de forma total o parcial los requerimientos identificados en la sección anterior. Finalmente, se muestran los resultados de una simulación prueba de sistemas fotovoltaicos con las herramientas seleccionadas y se discutirá cuál es la más apropiada para el enfoque planteado.

Características eléctricas de los principales componentes de un sistema fotovoltaico

Generador fotovoltaico

El efecto fotovoltaico convierte la radiación electromagnética en energía eléctrica. Un dispositivo fotovoltaico es aquel elemento que se diseña y construye con el objetivo de aprovechar al máximo el efecto fotovoltaico. Una de las características más útiles de estos dispositivos es su alta escalabilidad en potencia, esto es, que es posible generar potencias en un amplio rango desde unas decenas de miliwatts hasta gigawatts. Por ejemplo, la celda fotovoltaica más común es una oblea de silicio dopada de forma especial que puede generar entre 100 mW a 1 W. Es posible obtener potencias de entre 10 a 250 W construyendo matrices de celdas eléctricamente conectadas, normalmente en serie, conformando lo que se denomina un módulo o panel fotovoltaico. De igual forma, es posible conformar matrices de paneles fotovoltaicos para llegar a generar hasta decenas de gigawatts.

Existen distintos tipos de tecnologías de celdas fotovoltaicas, las cuales, tal y como se menciona, por ejemplo, en Zinsser, Makrides, Schmitt, Georghiou y Werner (2007), Cereghetti, Realini, Chianese y Rezzonico (2003), Friesen y colaboradores (2007a), Friesen y colaboradores (2007b), Chianese y colaboradores (2007) y Friesen y colaboradores (2010), responden eléctricamente de forma distinta a condiciones ambientales, y por lo tanto su comportamiento eléctrico responde a modelos matemáticos distintos. Por otro lado, el modelo general presentado en Meza y Ortega (2013) es lo suficientemente amplio para abarcar la mayoría de las tecnologías fotovoltaicas disponibles en la actualidad. La expresión matemática de este modelo es la siguiente:

$$i_{pv} = \Lambda(S, T, \rho_{pv}) - \Phi(S, T, v_{PV}, \rho_{PV}) \quad (1)$$

En donde

- i_{pv} es la corriente del generador fotovoltaico.
- v_{pv} es la tensión del generador fotovoltaico.
- S es la intensidad de radiación solar.
- T es la temperatura.
- ρ_{pv} es un conjunto de *n* parámetros que dependen de las características constructivas del dispositivo.

El modelo descrito en la ecuación (1) se puede adecuar a distintas tecnologías por medio de los siguientes parámetros:

- $\rho_{pv} = \{\rho_{pv1}, \rho_{pv2}, \ldots\}$ cada tecnología tiene un valor de ρ_{pvi} y un número de parámetros determinados.
- la función y la función pueden ser distintas para cada tecnología.

Es importante destacar que el anterior modelo pretende representar propiedades comunes que tienen las distintas tecnologías fotovoltaicas en su comportamiento eléctrico, esto es,

- la función $\Phi(S, T, v_{PV}, \rho_{PV})$ es siempre positiva y estrictamente creciente con respecto a v_{pv}.
- $\Lambda(S, T, \rho_{pv})$ es siempre positiva y mayor que $\Phi(S, T, v_{PV}, \rho_{PV})$.

Existen tecnologías fotovoltaicas que tienen asociadas capacitancias parásitas (Friesen et al., 2007). En este caso, es necesario agregar al anterior modelo un componente dinámico de la siguiente forma, tal y como se indica en Meza, Virtuani y Chianese (2010):

$$i_{pv} = \Lambda(S, T, \rho_{pv}) - \Phi(S, T, v_{PV}, \rho_{PV}) - C(v_{PV}, \rho_{PV})\dot{v}_{pv} \quad (2)$$

La función $C(v_{PV}, \rho_{PV})$ es normalmente no lineal y dependiente de v_{pv}.

Con base en los anteriores modelos, podemos afirmar que la simulación del transiente eléctrico de un sistema que utilice un dispositivo fotovoltaico debería ser capaz de:

- Resolver funciones matemáticas no lineales.
- Preferiblemente resolver funciones implícitas. Nótese que según la ecuación (1), existe la posibilidad de que la corriente del dispositivo fotovoltaico aparezca tanto en la parte derecha como en la parte izquierda de la ecuación. Dependiendo de cómo sea la forma de , será posible despejar o no la corriente con funciones matemáticas fundamentales. En algunos casos, utilizando funciones matemáticas como la función de Lambert W (Sharma, 2009) se puede despejar la corriente. En este caso, el sistema de simulación debería ser capaz de resolver esta función o permitir su programación.
- Resolver ecuaciones diferenciales no lineales, en el caso en que se utilice el modelo dinámico de la ecuación (2).

Unidad de procesamiento de potencia

Como se mencionó anteriormente, una unidad de procesamiento de potencia de un sistema fotovoltaico tiene dos objetivos principales:

1. Lograr que el dispositivo fotovoltaico esté siempre posicionado en el punto de operación de máxima potencia.
2. Lograr que la carga tenga un nivel de tensión y corriente determinados.

Dichos objetivos se obtienen por medio de dos elementos: el convertidor de potencia y el subsistema de control. El convertidor de potencia se entiende en este documento como el circuito electrónico conformado por componentes pasivos (inductores y capacitores) y activos (transistores, diodos, etc.), que permiten convertir niveles de tensión y de corriente. Algunos de los dispositivos activos del convertidor de potencia son controlados por medio del subsistema de control, el cual, con las mediciones de las variables eléctricas del convertidor de potencia, los valores de referencia y un algoritmo de control, genera las señales de control. La figura 3 muestra el diagrama de bloques de una unidad de potencia.

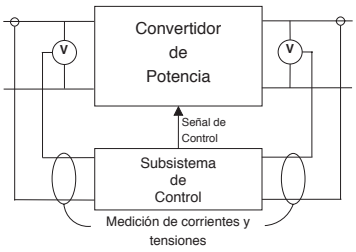

Figura 3. Diagrama de bloques de una unidad de procesamiento de potencia.

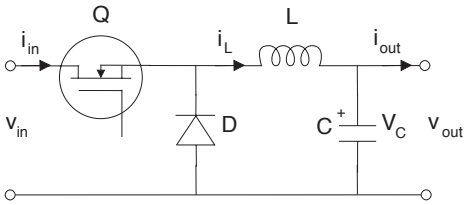

Figura 4. Circuito de un convertidor de potencia tipo reductor.

La gran mayoría de los sistemas fotovoltaicos utiliza convertidores de potencia conmutados. Estos convertidores consisten de capacitores, inductores, diodos y transistores (usualmente MOSFET). Estos últimos operan de forma conmutada con el fin de obtener menores pérdidas en el proceso de conversión. Un convertidor de potencia se puede modelar matemáticamente por medio de un conjunto de ecuaciones diferenciales que describen el proceso de carga y descarga de los elementos que almacenan energía dentro del circuito, esto es, el inductor y el capacitor. Por ejemplo, las ecuaciones que describen el comportamiento dinámico del convertidor reductor que se muestra en la figura 4 son las siguientes:

$$\frac{di_L}{dt} = u\frac{v_{in}}{L} - \frac{v_C}{L}$$

$$\frac{dv_C}{dt} = \frac{i_L}{C} - \frac{i_{out}}{C}$$

En donde

- i_L es la corriente en el inductor L.
- v_c es la tensión en el capacitor C.
- v_{in} es la tensión de entrada al convertidor.
- i_{out} es la corriente de salida del convertidor.
- u es la señal de conmutación del convertidor.

Nótese que en este caso *u* es discontinua, esto es, solo puede tomar los valores 0 o 1. En algunos casos es conveniente utilizar el modelo promediado del convertidor, descrito por las siguientes ecuaciones:

$$x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} \bar{i}_L \\ \bar{v}_C \end{bmatrix}$$

$$\dot{x} = \begin{bmatrix} 0 & -\frac{1}{\mathrm{L}} \\ \frac{1}{\mathrm{C}} & 0 \end{bmatrix} x + \begin{bmatrix} 0 \\ \frac{\bar{i}_{out}}{\mathrm{C}} \end{bmatrix} + \begin{bmatrix} \frac{\bar{v}_{in}}{\mathrm{L}} \\ 0 \end{bmatrix} d$$

En este caso *d* sustituye a *u*, en donde *d* es una señal continua comprendida entre 0 y 1. Es importante destacar que la simulación del modelo promediado no permite ver los efectos del rizado de conmutación en el sistema.

El subsistema de control es el encargado de conmutar los transistores del convertidor de potencia o equivalentemente de generar la señal *u*. La señal *u* también puede obtenerse a la salida de un generador de anchura de pulso en el caso en el cual el subsistema de control genere la señal *d*. Para generar esta señal, el subsistema de control puede utilizar un algoritmo empírico, ecuaciones diferenciales lineales o no lineales o ecuaciones en diferencia (sistemas discretos).

De esta forma, para la UPP, el sistema de simulación debe ser capaz de:

- Resolver numéricamente ecuaciones diferenciales lineales, no lineales, continuas y con discontinuidades.
- Resolver ecuaciones en diferencia.
- Permitir programar algoritmos empíricos (redes neuronales, lógica difusa, entre otros).

Finalmente, sería de mucha utilidad que el algoritmo o las ecuaciones dinámicas del subsistema de control pudieran exportarse de forma sencilla, para ser implementados en un microcontrolador o un sistema de prototipado rápido.

Herramientas de simulación

Scilab/Scicoslab: Scilab es un programa computacional científico para cálculos numéricos, que provee un poderoso entorno de computación abierta para aplicaciones científicas y de ingeniería. Scicos permite realizar modelados y simulaciones en forma gráfica de sistemas dinámicos, lo cual se ejecuta bajo Scilab. Scilab y Scicos han sido desarrollados por el Instituto Nacional Francés para la Investigación en Ciencias de la Computación (INRIA), desde 1990. Además del conjunto de funciones predefinidas de Scilab, un grupo de investigadores ha desarrollado nuevas e importantes funciones con propósitos educativos (por ej., Pendharkar,

2005 y Meza, Andrade-Romero, Alavarez y Coelho, 2010) y para el diseño de sistemas de control avanzados (por ej., Delebecque, 2000). Scilab ha sido utilizado en una amplia variedad de aplicaciones en investigación, tales como la identificación de fuentes de contaminación en ríos, descrita en Chancelier, Cohen, Maldinery y Pacard (1996) y en la mejora de las máquinas de prueba para fricción, mencionado en Dongping, Dongjie y Qiang (2007). Siendo un *software* científico altamente sofisticado y gratuito, ha podido ser utilizado ampliamente para propósitos educativos, tanto en países industrialmente desarrollados como en vías de desarrollo.

Python: es un lenguaje de programación de propósito general orientado a objetos diseñado para ser interpretado, esto es, está compuesto de un conjunto de instrucciones que deben ser convertidas a operaciones ejecutables por el computador. Esto hace que el lenguaje se pueda aprender de forma sencilla ya que el código creado se puede probar y depurar de forma rápida y fácil.

Python se ha extendido ampliamente en los últimos años debido sobre todo a las siguientes ventajas:

- Es libre y abierto y está incluido en la gran mayoría de las distribuciones de Linux.
- Está disponible para la mayoría de los sistemas operativos (Linux, Unix, Windows, Mac OS). Un programa escrito para un sistema se ejecuta en los otros sin necesidad de modificarlo.
- Es más fácil de aprender y produce código más fácilmente entendible que otros lenguajes de programación.
- Python y sus extensiones son fáciles de instalar.

Existen numerosas librerías y extensiones para Python. Una de las más útiles para la simulación de sistemas dinámicos es SciPy, que es una colección de algoritmos para el cálculo numérico, procesamiento de datos y operaciones matriciales, optimización y estadística, entre otros. El usuario puede utilizar estos algoritmos de una forma sencilla e intuitiva y de modo muy similar a otros lenguajes comerciales para uso científico, como Matlab.

Una de las grandes ventajas de usar Python con SciPy es, además de las ventajas de Python anteriormente mencionadas, que funciona como una forma flexible y útil de integrar librerías compiladas en otros programas como C, C++ y Fortran. Por estas razones, Python y SciPy se han usado para el modelado de sistemas dinámicos en múltiples y variadas aplicaciones, como el modelado de sistemas celulares (Olivier, Rohwer y Hofmeyr, 2005) y convertidores de potencia (Meza y Ortega, 2013).

Prueba de simulación y discusión

Con el fin de probar las herramientas de simulación identificadas, se utilizó un caso de estudio de un sistema fotovoltaico para la carga de una batería. Este sistema se muestra en la figura 5 y consiste de un panel fotovoltaico que alimenta a una batería por medio de un convertidor de potencia reductor. Las ecuaciones dinámicas que describen el comportamiento eléctrico de este sistema son las siguientes:

$$\begin{aligned} \dot{x_1} &= \frac{i_{pv}(x_1)}{\mathrm{C}} - \frac{ux_2}{\mathrm{C}} \\ \dot{x_2} &= \frac{ux_1}{\mathrm{L}} - \frac{E}{\mathrm{L}} \\ i_{pv}(x_1) &= \Lambda - \Phi(x_1) \end{aligned} \quad (3)$$

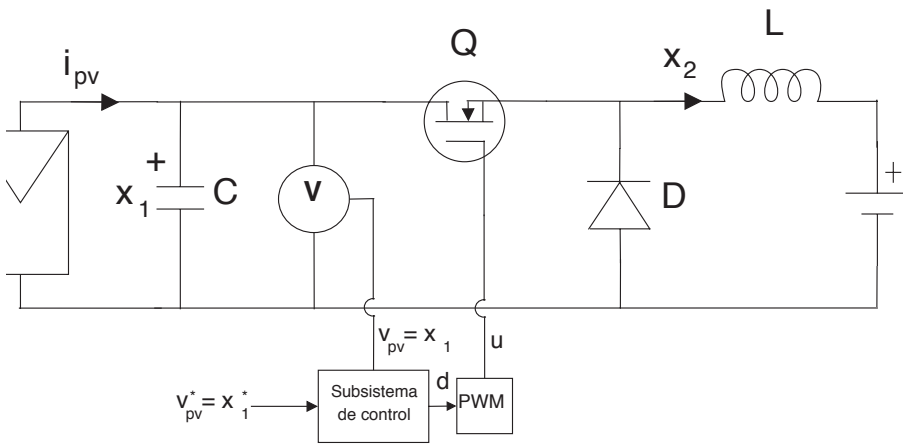

Figura 5. Circuito esquemático del caso de estudio utilizado para evaluar las herramientas de simulación.

en donde $\Phi(\cdot) = \psi e^{\alpha v_{pv}}$ y se considera que la radiación *S* y la temperatura *T* son constantes. El cuadro 1 muestra el valor de los parámetros utilizados para la simulación.

Cuadro 1. Parámetros del sistema descrito por la ecuación (3) para su simulación.

Parámetro	Valor	Unidades
ψ	0.0022	A
Λ	1.2	A
α	0.2	1/V
E	12	V
L	47	mH
C	0.1	mF

Como se puede observar en la figura 5, existe un subsistema de control que genera un ciclo de trabajo *d* que luego es es utilizado por un modulador de anchura de pulsos para generar la señal *u* que controla la activación del transistor. El subsistema de control también está definido por medio de un conjunto de ecuaciones diferenciales que se muestra a continuación.

$$u = K_P \tilde{x}_1 + K_I \omega$$
$$\dot{\omega} = \tilde{x}_1$$

$$\tilde{x}_1 = x_1 - x_1^* \qquad (4)$$

en donde los parámetros K_P y K_I se utilizan para ajustar el comportamiento dinámico del sistema en lazo cerrado.

El subsistema de control debe asegurar que $d \in [0,1]$. De esta forma, el modulador de anchura de pulsos, comparando *d* con una señal moduladora triangular o diente de sierra con una amplitud de 1, puede generar *u*. La expresión matemática que describe la operación del modulador de anchura de pulsos es la siguiente:

$$m = \frac{1}{T_S} - \left[\frac{1}{2} + \frac{t}{T_S}\right] + \frac{1}{2} \quad (5)$$
$$u = \begin{cases} 1 \text{ si } d > m \\ 0 \text{ si } d < m \end{cases}$$

en donde representa la función máximo entero (*floor*) que se define de la siguiente forma:

$$\lfloor x \rfloor = max\{k \in Z \vee k \leq x\}$$

esto es, la función máximo entero aplicado a una variable *x* devuelve el máximo número entero *k* no superior a *x.*

Llegados a este punto, es importante destacar que existen dos sistemas que se pueden analizar y estudiar. Uno que considera únicamente el comportamiento eléctrico del sistema de la ecuación 3 promediado en los periodos de conmutación del transistor y otro que sí toma en cuenta los efectos del sistema debido a la conmutación. Tal y como se mencionó anteriormente, el primer caso se obtiene de la ecuación 3 sustituyendo *u* por *d*. A continuación, se presentan los resultados de simulación para ambos casos.

Las condiciones iniciales y los parámetros para el método numérico utilizados se muestran en los cuadros 2 y 3, respectivamente. Los parámetros del subsistema de control utilizados fueron K_P = 0.1 y K_I = 0.75 con una señal de referencia $x_1^* = v_{pv}^* = 24V$.

Cuadro 2. Condiciones iniciales (cuando *t=0*) del sistema descrito por la ecuación (3) para su simulación.

Variable	Valor
$x_1 = v_{pv}$	31.51
$x_2 = i_{out}$	0
W	0

Cuadro 3. Parámetros para el método numérico utilizado para resolver las ecuaciones diferenciales del sistema simulado.

Parámetro	Valor
Máximo error absoluto permitido	$1x10^{-8}$
Máximo error relativo permitido	$1x10^{-8}$
Paso de integración máximo permitido	0.001
Tiempo de simulación	0.6
Cantidad de datos tomados en el tiempo de simulación	$1x10^{5}$

Simulación con Python

Se utilizó la versión 2.7.3 de Python con la extensión Numpy (ver 1.6.2) y las bibliotecas SciPy (ver 0.10.1) y Matplotlib (ver 1.1.1). El programa que describe el sistema a simular consistió en unas pocas líneas de código que utiliza, principalmente, el solucionador de ecuaciones diferenciales incluido en la biblioteca SciPy. Dicho solucionador está tomado de la colección de métodos numéricos descritos en la librería ODEPACK para Fortran. Este solucionador de ecuaciones diferenciales permite resolver numéricamente ecuaciones diferenciales ordinarias de tipo rígidas o *stiff*, que en varias ocasiones están presentes en sistemas dinámicos de convertidores de potencia. Los resultados de simulación del sistema promediado pueden observarse en la figura 6, los cuales se corresponden con el comportamiento esperado.

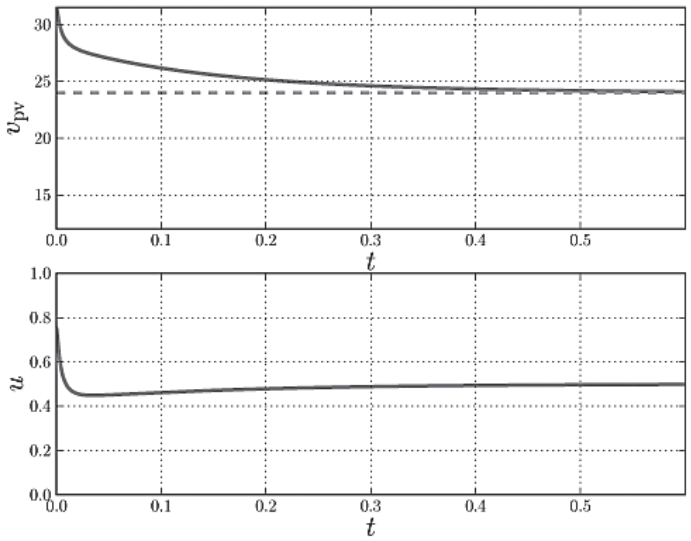

Figura 6. Resultados de simulación del sistema de la Figura 4 descrito por las ecuaciones 4 y 5.

Se pudo comprobar que Python y sus extensiones existentes no permiten de forma sencilla y directa simular sistemas dinámicos que deben realizar funciones guiadas por distintos períodos de tiempo. Este es el caso de la simulación del sistema en estudio tomando en cuenta las conmutaciones del transistor. En este sentido, es importante destacar que, en la mayoría de los casos, el subsistema de control es diseñado a partir del modelo promediado del sistema. Esto ocurre, por ejemplo, cuando se utilizan controles proporcionales e integrales tal y como el que está indicado en la Ecuación 4. Bajo estas condiciones, se parte del supuesto de que la señal generada por el control *d* permanecerá constante durante un ciclo de conmutación del transistor. Más aún, el subsistema de control debería recibir las señales promediadas o muestreadas de tal y como se indica en la figura 7. Si bien es cierto, los mencionados requerimientos no representan mayor problema para el análisis dinámico del sistema o para el diseño del subsistema de control, sí dificultan su implementación en Python. Como se dijo anteriormente, la dificultad radica en la necesidad de definir subsistemas que operan a distintos instantes, a saber, el sistema de ecuaciones diferenciales que es resuelto a un paso de integración dado y el sistema de promediado o muestreo de datos que debe resolver en un período de conmutación del transistor.

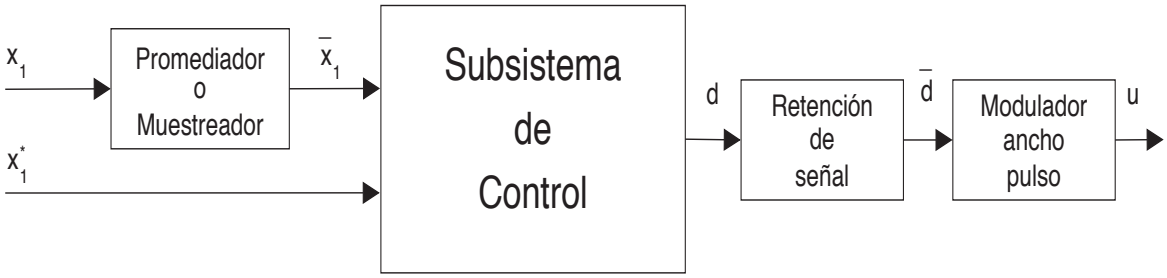

Figura 7. Subsistema de control que genera un ciclo de trabajo *d* para un modulador de anchura de pulsos.

Aunque es posible escribir código en Python para atender el requerimiento anteriormente mencionado, los autores consideran que resulta innecesario dado que, como se indica a continuación, existen otras herramientas que resuelven el anterior problema de forma muy sencilla.

Simulación con Scicos

Para esta simulación se utilizó la versión 4.4.1-1 de Scicoslab y la versión 4.4.1 de Scicos. Al igual que en el caso de Python + SciPy, Scicoslab y Scicos utilizan el paquete ODEPACK para la resolución numérica de ecuaciones diferenciales ordinarias.

Scicos permite definir un sistema dinámico por medio de un diagrama de bloques, con lo cual el caso de prueba se definió tal y como se muestra en las figuras 8 y 9. En la figura 8 se muestra el modelo del sistema promediado que se corresponde con el sistema simulado utilizando Python descrito en la sección precedente. Por otro lado, la figura 9 muestra el diagrama de bloques del sistema con el modulador de anchura de pulsos. Este sistema no se pudo simular en Python por las razones anteriormente expuestas, no obstante, gracias a la flexibilidad que ofrece la definición del sistema con bloques, este sí se pudo realizar en Scicos.

El resultado de la simulación en Scicoslab + Scicos con los parámetros de simulación de el cuadro 3 se muestra en las figuras 10 y 11. Nótese que para el modelo promediado se obtuvo la misma respuesta dinámica en Python y en Scicoslab (Figura 6 - Python y Figura 10 - Scicoslab). Así mismo, en la figura 11 se muestra un comportamiento promedio similar al de las figuras 6 y 10, pero en este caso es posible ver el rizado en la tensión del dispositivo fotovoltaico debido a la conmutación del interruptor (ver figura 12).

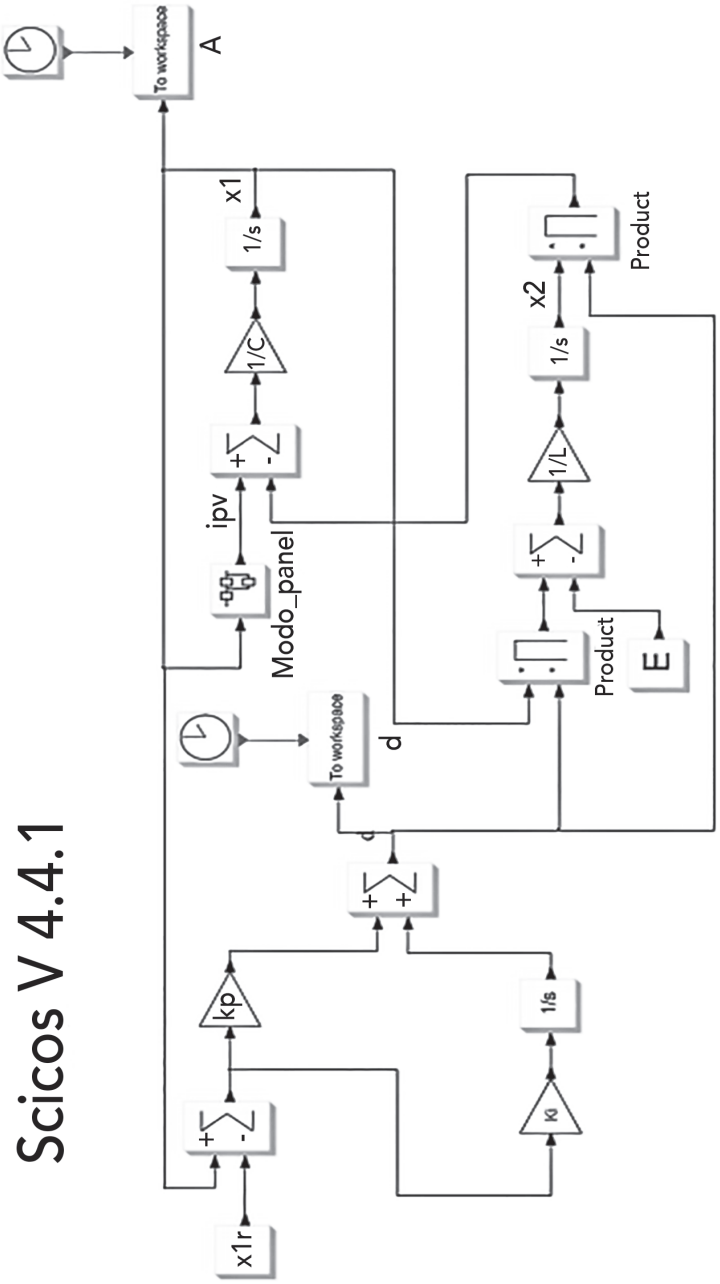

Figura 8. Diagrama de bloques del sistema promediado, equivalente a la simulación realizada con Python.

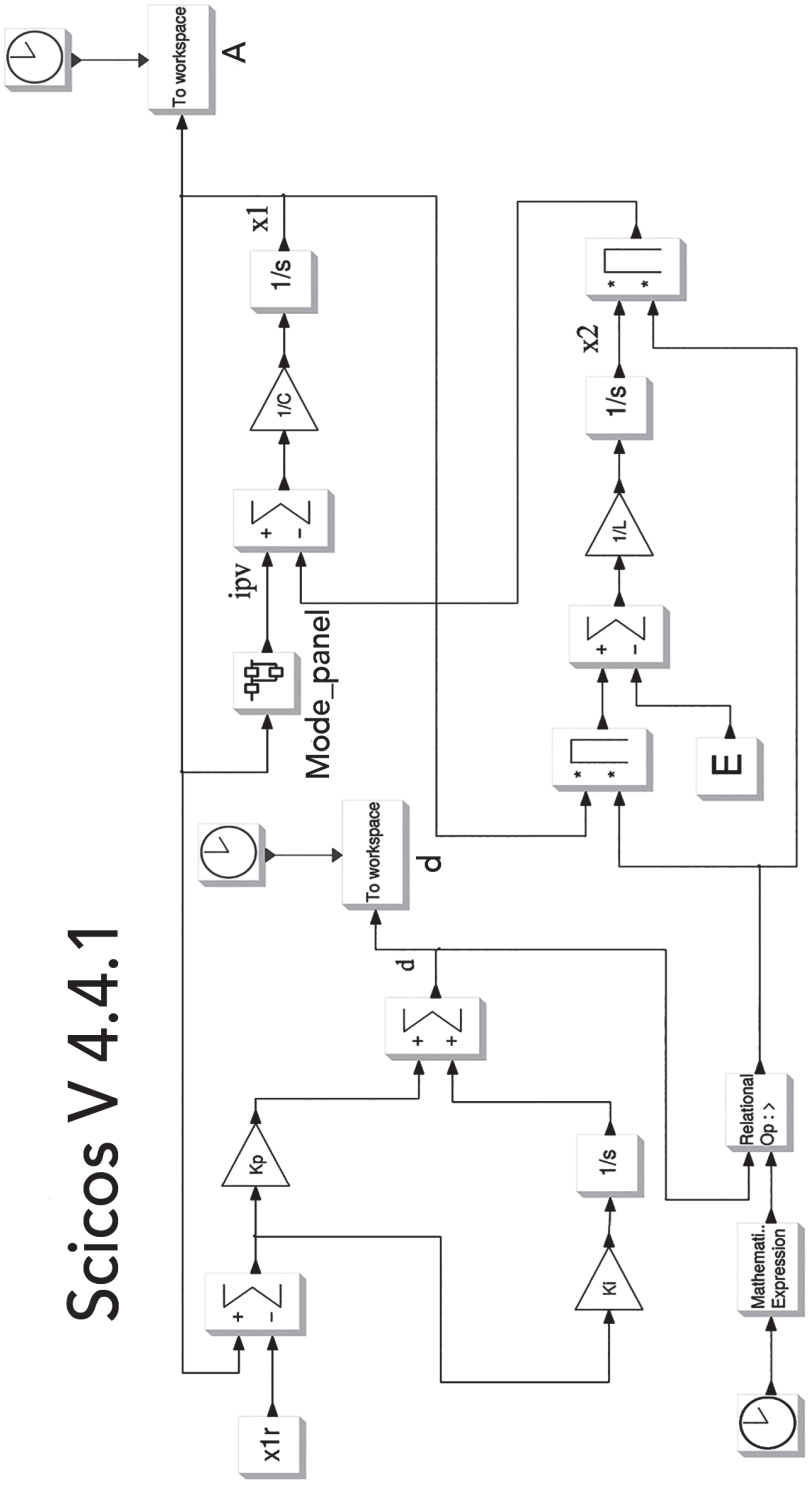

Figura 9. Diagrama de bloques del sistema con el control y el modulador de ancho de pulso.

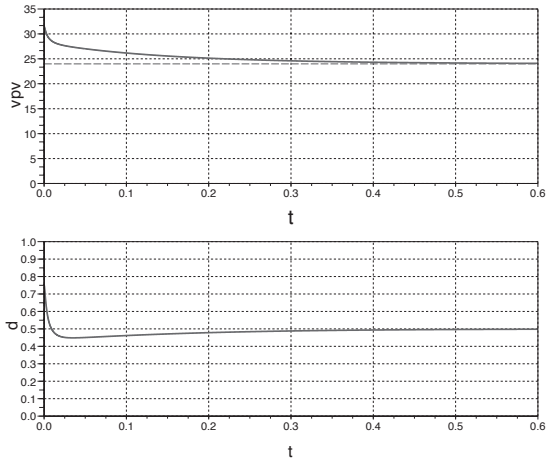

Figura 10. v_{pv} y u obtenidos de la simulación con Scicoslab + Scicos, para el sistema de la Figura 5 con el modelo de la Figura 8.

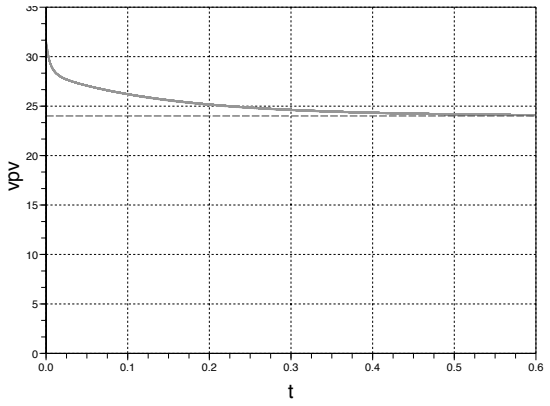

Figura 11. v_{pv} obtenido de la simulación con Scicoslab + Scicos, para el sistema de la Figura 5 con el modelo de la Figura 9.

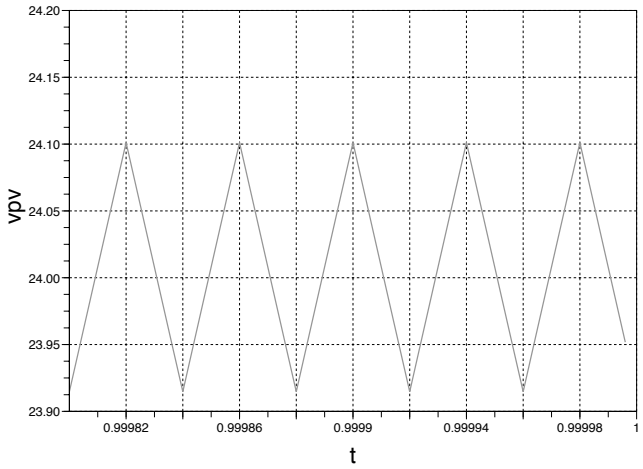

Figura 12. Rizado de la tensión v_{pv} mostrado en la Figura11.

Conclusiones y recomendaciones

En el presente artículo se ha utilizado un sistema fotovoltaico para cargar una batería, como un caso de estudio que permita probar la versatilidad de dos herramientas de simulación de sistemas dinámicos, a saber, Python + SciPy y Scicoslab + Scicos. Dichas herramientas son abiertas y libres y pueden ser utilizadas en las plataformas informáticas más comunes.

Tanto Python como Scicoslab permitieron realizar simulaciones del sistema dinámico promediado, no así del sistema completo conformado por el control y el modulador de ancho de pulso, elementos presentes en los sistemas fotovoltaicos convencionales. En este apartado, el trabajo con Scicoslab se pudo realizar de forma sencilla, debido principalmente a que permite el desarrollo de simulaciones que requieren funciones guiadas por distintos periodos de tiempo. Esto último, si se quisiera desarrollar en Python, exigiría un esfuerzo importante en programación.

Pese a que Python es muy versátil debido a que permite integrar numerosas librerías para el cálculo numérico de manera sencilla, el desarrollo de procesos de simulación con cierto nivel de complejidad exige una inversión importante de tiempo en la etapa de programación. Por otro lado, Scicoslab brinda un ambiente gráfico que facilita el proceso de definición e implementación de sistemas dinámicos complejos, en los cuales es posible manejar diferentes tiempos de procesamiento, como, por ejemplo, para la resolución numérica de una ecuación diferencial ordinaria y el proceso de conmutación del transistor, ambas situaciones presentes en las unidades de procesamiento de potencia de los sistemas fotovoltaicos.

Bibliografía

Boke. U. (septiembre, 2007). *A simple model of photovoltaic module electric characteristics.* European Conference on Power Electronics and Applications, 1-8.

Cereghetti, N., Realini, A., Chianese, D. & Rezzonico, S. (2003). *Power and Energy production of PV modules.* IEEE 3rd World Conference in Photovoltaic Energy Conversion.

Chancelier, J., Cohen, M., Maldinery, M. & Pacard, F. (1996). *Identification of pollution sources in rivers*. IEEE International Symposium on Computer-Aided Control System Design.

Chianese, D., Friesen, G., Pasinelli, P., Pola, I., Realini, A., Cereghetti, N. & Bernasconi, A. (2007). *Direct Performance Comparison of PV Modules*. 22th European Photovoltaic Solar Energy Conference.

Delebecque, F. (2000). *A slicot based control library for Scilab*. IEEE International Symposium on Computer-Aided Control System Design.

Dongping, Q., Dongjie, Z. & Qiang, T. (2007). *Linux/rtai and Scicos in low cost high performance friction testing machine.* International Conference on Electronic Measurement and Instruments.

Friesen, G., Gottschalg, R., Beyer, H. G., Williams, S., Guerin de Montgareuil, A., van der Borg, N., van Sark, W., Huld, T., Müller, B., de Keizer, A. C. & Niu, Y. (2007a). En *Intercomparison of different energy prediction methods within the European project performance – results of the 1nd Round Robin.* 22th European Photovoltaic Solar Energy Conference.

Friesen, G., Chianese, D., Pola, I., Realini, A. & Bernasconi, A. (2007b). En *Energy Rating Measurements and Predicitions at ISAAC.* 22th European Photovoltaic Solar Energy Conference.

Friesen, G., Chianese, D., Dittmann, S., Dominé, D., Burà, E., Strepparava, D., Margna, B., Denicolà, M., Meoli, R. & Pola, I. (2010). En *Performance intercomparison of 13 different PV modules based on indoor and outdoor test.* 25th European Photovoltaic Solar Energy Conference.

Gow, J. A. & Manning, C. D. (marzo, 1999). Development of a photovoltaic array model for use in power-electronics simulation studies. Electric Power Applications. *IEE Proceedings, 146*(2), 193-200.

Kiusalaas, J. (2013). *Numerical Methods in Engineering with Python 3*. Cambridge University Press.

Lasnier, F. (1990). *Photovoltaic Engineering Handbook*. Taylor & Francis.

Liu, S. & Dougal, R. (2002). Dynamic multiphysics model for solar array. *IEEE Transactions on Energy Conversion, 17*, 285-294.

Meza, C., Andrade-Romero, J. A., Alavarez, M. A. & Coelho, A. A. A. (2010). *Improving Control Engineering Education Using Foss Tools.* XVIII Congresso Brasileiro de Automatica, Bonito, Brasil.

Meza, C., Virtuani, V. & Chianese, D. (septiembre, 2010). *Evaluation of models for the internal capacitance of a pvmodule for the design and simulation of power converters.* 25th European Photovoltaic Solar Energy Conference and Exhibition.

Meza, C. & Ortega, R. (2013). *Control and estimation scheme for PV central inverters*. XXIV IEEE International Conference on Information, Communication and Automation Technologies (ICAT).

Olivier, B. G., Rohwer, J. M. & Hofmeyr, J. H. (2005). Modelling cellular systems with PySCeS. *Bioinformatics* 21.4 (2005), 560-561.

Pendharkar, I. (2005) Rltool for Scilab: A public domain tool for SISO system design. *IEEE Control System Magazine, 25*, 23-25.

Sharma, S. & Kapoor, A. (2009). PV Generator Driven First Order Circuit – Transient Analysis using LambertW Function. *The Open Renewable Energy Journal*, *2*, 111-115.

Turcotte, D., Ross, M. & Sheriff, F. (septiembre, 2001). Photovoltaic hybrid system sizing and simulation tools: status and needs. En *PV Horizon: Workshop on Photovoltaic hybrid systems,* Montreal, Canada.

Zinsser, B., Makrides, G., Schmitt, W., Georghiou, G. E. & Werner, J. H. (2007). *Annual energy yield of 13 photovoltaic technologies in Germany and in Cyprus.* 22th European Photovoltaic Solar Energy Conference.