Desarrollo de un algoritmo de cálculo simbólico para la resolución de las ecuaciones de decaimiento y transmutación de Bateman

Carlos Antonio Cruz López, Juan Luis François Lacouture
Universidad Nacional Autónoma de México
cacl@ier.unam.mx

RESUMEN

El modelo matemático más adecuado para describir los cambios en la composición isotópica de un reactor nuclear, consiste en un plantear un conjunto de ecuaciones diferenciales lineales acopladas. En 1910 el matemático Harry Bateman propuso una solución para dicho sistema de ecuaciones, la cual se conocido como la solución de Bateman. En el presente trabajo, se ha desarrollado un algoritmo que puede construir las soluciones a cadenas cíclicas de una forma rápida a través del cálculo simbólico. Dicho algoritmo, utiliza un análisis combinatorio de las posibles posiciones que pueden ocupar elementos repetidos, y a través de un conjunto de funciones de variables del tipo "string", construye las soluciones de una forma veloz y dinámica. Así mismo, el presente trabajo incluye un análisis del tiempo de cómputo que consume dicho algoritmo, y propone una forma de realizar una comparación con las soluciones existentes.

PALABRAS CLAVE

Algoritmo, cálculo simbólico, ecuación de Bateman, cálculo numérico.

ABSTRACT

The most appropriate mathematical model to describe the changes in the isotopic composition of a nuclear reactor is to propose a set of coupled linear differential equations. In 1910 the mathematician Harry Bateman proposed a solution for this system of equations, which is known as Bateman's solution. In the present work, an algorithm has been developed that can build solutions to cyclic chains in a quick way through symbolic calculation. This algorithm, uses a combinatorial analysis of the possible positions that repeated elements can occupy, and through a set of functions of variables of type "string", builds the solutions in a fast and dynamic way. Likewise, the present work includes an analysis of the computation time consumed by said algorithm, and proposes a way to make a comparison with the existing solutions.

KEYWORDS

Algorithm, symbolic calculation, Bateman´s equation, numerical calculation.

INTRODUCCIÓN

El combustible de un reactor nuclear experimenta cambios en su composición isotópica en el transcurso del tiempo. Dichos cambios se deben principalmente a tres procesos que se presentan en forma de transformaciones sucesivas: el fenómeno de fisión, el de captura y el de decaimiento radiactivo. Estas transformaciones dan lugar a nuevos isótopos, y a la reducción o desaparición de otros. Dado que las propiedades neutrónicas de un reactor dependen de forma considerable de la composición isotópica del combustible, es necesario cuantificar matemáticamente estos cambios.

Si se utiliza el conjunto de funciones $X_i(t), 1 \leq i \leq n$ para denotar la concentración del isótopo i como función del tiempo, es posible representar una sucesión de transformaciones sucesivas mediante la siguiente estructura:

$$X_1 \xrightarrow{r_{1,2}} X_2 \xrightarrow{r_{2,3}} \dots \xrightarrow{r_{n-1,n}} X_n \tag{1}$$

Donde $r_{k,k+1}$ representa la reacción mediante la cual el isótopo k se ha transformado en el isótopo k+1. Estructuras como la que se muestra en (1), donde los isótopos tienen a lo más un sucesor y a lo más un antecesor, reciben el nombre de cadenas lineales.

A pesar de que en la práctica es más probable encontrar estructuras de transformaciones sucesivas más complejas, llamadas redes de decaimiento y transmutación, es posible reducir prácticamente cualquier estructura compleja a un conjunto de cadenas lineales a través del proceso de linearización y superposición.[1] Por lo tanto, desarrollar soluciones para cadenas lineales como la descrita en (1) es fundamental para el análisis de problemas de quemado y activación en ingeniería nuclear.

Gracias a los trabajos de Rutherford[2] y Rubinson[3], es posible modelar las transformaciones sucesivas en (1), a través del siguiente sistema de ecuaciones diferenciales acopladas:

$$\frac{d}{dt}X_i = b_{i-1,i}^{\text{eff}}\lambda_{i-1}^{\text{eff}}X_{i-1} - \lambda_i^{\text{eff}}X_i, \text{ con } b_{0,1}^{\text{eff}} = 0 \tag{2}$$

Donde $b_{i\text{-}1}^{eff},i$ representa el coeficiente de ramificación efectivo, que es el porcentaje de las transformaciones que experimenta el isótopo i-1 y que dan lugar a la creación del isótopo i. Por otro lado, los coeficientes λ_i^{eff} son una generalización de la constante de decaimiento radiactiva, que incluye también la razón de pérdidas por captura y fisión. Si se consideran las siguientes condiciones iniciales:

$$X_1(0) \neq 0, X_i(0) = 0, i = 2, \dots, n \tag{3}$$

Entonces, es posible resolver la ecuación (2) usando la solución que propuso en 1910 el matemático Harry Bateman, quien resolvió un sistema similar que solo consideraba decaimiento radiactivo.[4] Dicha solución es la siguiente[5]:

$$X_n(t) = X_1(0)\prod_{k=1}^{n-1} b_{k,k+1}^{\text{eff}}\lambda_k^{\text{eff}} \sum_{i=1}^{n} e^{-\lambda_i^{\text{eff}}t} \prod_{\substack{j=1 \\ j\neq i}}^{n} \frac{1}{\left(\lambda_j^{\text{eff}} - \lambda_i^{\text{eff}}\right)} \tag{4}$$

La ecuación (4) es útil para resolver un gran número de problemas de quemado y activación. Sin embargo, no puede usarse cuando existen elementos o isótopos repetidos en una cadena lineal, es decir cuando hay dos isótopos iguales en (1),

situación que por lo regular se presenta en estructuras conocidas como cadenas cíclicas. Como se puede observar, en dicho caso sucederá que la resta $\lambda_j^{eff} - \lambda_i^{eff}$ será igual a cero para algún par de *i* y *j*, y por lo tanto el término dentro de la productoria en (4) no estará definido. Dado que las cadenas cíclicas son comunes, sobre todo en estructuras originadas por la transmutación y decaimiento de isótopos pesados, este tópico es de vital importancia e interés para el desarrollo de códigos de quemado y activación.

Para superar este problema se desarrollaron dos metodologías durante el siglo pasado. La primera de ellas consistió en introducir pequeñas modificaciones en las constantes de decaimiento efectivas, de tal forma que la resta que se mencionó antes nunca sea igual a cero. La segunda metodología, consistió en desarrollar soluciones más generales que consideraran la posibilidad de que uno o más elementos en una cadena lineal pudieran repetirse.

El desarrollo de soluciones generales comenzó prácticamente desde 1927 [6] y ha continuado a lo largo de los años, siendo la más popular la desarrollada por Cetnar en el 2006 [7], y siendo la desarrollado por Dreher en el 2013 [8] la que se ha publicado más recientemente. Sin embargo, como algunos autores han señalado[9], el uso de las soluciones generales involucra un mayor tiempo de cómputo, debido a que prácticamente se construye una solución para cada cadena lineal con elementos repetidos. Además, que algunas soluciones generales no pueden implementarse de forma directa en un algoritmo debido a su complejidad, y por lo tanto tienen que ser aproximadas.

En el presente trabajo se propone una metodología alternativa para resolver cadenas cíclicas, que consume una menor cantidad de tiempo de cómputo, y que no necesita ninguna aproximación. Esencialmente, esta metodología consiste en utilizar una formulación integral y por método de Laplace para construir la solución de cadenas cíclicas, acelerando dicho proceso mediante la implementación de un algoritmo de cálculo simbólico.

Para dicha formulación, fue necesario llevar a cabo un análisis combinatorio que identificara todas las posibles configuraciones de cadenas con elementos repetidos que pueden presentarse, descartando aquellas que físicamente son imposibles. Luego, a través del cálculo simbólico, se construyen estas soluciones usando un algoritmo con variables del tipo "*string*".

Como se mencionó, a diferencia de la implementación computacional de otras soluciones generales, el cálculo simbólico (también conocido como álgebra computacional) puede reducir de forma considerable el tiempo de cómputo sin necesidad de aproximar numéricamente la solución de cadenas cíclicas.

La estructura del presente trabajo es la siguiente: en la Sección 2, se describe de forma breve el desarrollo de soluciones generales, así como los principales métodos matemáticos que se han usado para obtenerlas. La Sección 3 contiene una breve descripción del cálculo simbólico, y en ella se analizan las principales partes del algoritmo desarrollado. La Sección 4 contiene la metodología para realizar comparaciones, y los resultados preliminares, y finalmente en la Sección 5 se presentan las conclusiones.

SOLUCIONES GENERALES

De esta sección en adelante, se entenderá por solución general cualquier solución del sistema descrito en (1) y (2), que tiene las condiciones iniciales dadas en (3), y que admite el caso donde existan elementos repetidos.

Luego de una profunda revisión, se encontraron al menos 17 publicaciones relacionadas con el desarrollo de soluciones generales para la ecuación de Bateman, o que resuelven sistemas similares al descrito en (2). Dichas soluciones pertenecen a un amplio rango de disciplinas, que van desde la astrofísica, hasta la teoría de transporte, incluyendo a la ingeniería nuclear. Así mismo, la metodología usada para el desarrollo de dichas soluciones no siempre fue la misma. En la tabla I se listan de forma cronológica las publicaciones.

Es posible clasificar este conjunto de soluciones en 4 tipos, de acuerdo con las herramientas matemáticas usadas para obtenerlas: solución por transformada de Laplace, solución por formulación integral, solución por expansión en serie de potencias y la solución por límites en incrementos.

En el presente trabajo se seleccionaron dos soluciones para ser programadas en cálculo simbólico. La primera de ellas fue la solución basada en una formulación integral. Como se verá más adelante, esta metodología involucra esencialmente el cálculo de integrales. Después se consideró la solución obtenida por Transformada de Laplace, porque esta involucra el cálculo de múltiples derivadas. De esta forma se consideró adecuado comparar la implementación del cálculo simbólico, determinando qué era más rápido: calcular derivadas o calcular integrales.

Para el resto de las soluciones, basta decir que la solución por expansión en series de potencia, usa justamente esta técnica en una parte de la solución de Bateman, para evaluar el caso en que existen constantes de decaimiento idénticas. La solución por límite en incrementos consiste en plantear la ecuación (4) con elementos repetidos, cuya constante efectiva de decaimiento es modificada con un incremento Δ, al que posteriormente se hace tender a cero.

Tabla I. Listado de publicaciones relacionadas con soluciones generales.

Año	Autores	Método	Ref.	Año	Autores	Método	Ref.
1927	Meyer y Schweildler	Integral	6	1994	Vukadin	Serie de potencias	18
1942	Rubinson	Integral	3	1996	Pommé et al.	Integral	19
1961	Clayton	Laplace	11	1997	Mirzadeh y Walsh	Integral	20
1973	Bell	Límite	12	1999	Wilson	Integral	10
1978	Newman	Laplace	13	1999	Popovic'	Laplace	21
1980	Tasaka	Integral	14	2006	Cetnar	Límite	7
1981	Miles	Integral	15	2008	Slodička y Balážová	Límite	22
1989	Raykin y Shlyakhter	Integral	16	2013	Dreher	Límite	8
1993	Blaauw	Laplace	17				

Vale la pena mencionar que, por la forma en la que están expresadas estas soluciones, no resulta evidente comprobar que son equivalentes. Aún más, soluciones como la de Cetnar[7] están escritas en forma de sumatorias anidadas, mientras que otras como las de Wilson[10] o Dreher[8] están expresadas en términos de derivadas múltiples.

Solución por formulación integral

La solución por formulación integral parte de la ecuación (2). Si se arreglan los términos y se multiplica dicha ecuación por el factor integrante $e^{\lambda_i^{\text{eff}} t}$), se tiene que:

$$\frac{d}{dt}\left(X_i e^{\lambda_i^{\text{eff}} t}\right) = b_{i-1,i}^{\text{eff}} \lambda_{i-1}^{\text{eff}} X_{i-1} e^{\lambda_i^{\text{eff}} t} \tag{5}$$

Lo cual se reduce a:

$$X_i(t) = X_i(0)e^{-\lambda_i^{\text{eff}} t} + b_{i-1,i}^{\text{eff}} \lambda_{i-1}^{\text{eff}} e^{-\lambda_i^{\text{eff}} t} \int_0^t X_{i-1}(t_1) e^{\lambda_i^{\text{eff}} t_1} \, dt_1 \tag{6}$$

Como se puede observar, la ecuación por formulación integral es del tipo recursivo, en cuanto a que requiere conocer X_{i-1} para poder obtener X_i. Con el fin de analizar el comportamiento de este método cuando existen elementos repetidos, es necesario desarrollar la ecuación (6) para varios términos, comenzando con i=2. Para ello se considerará que:

$$(t) = X_1(0) e^{-\lambda_i^{\text{eff}} t} \tag{7}$$

Algo que puede obtenerse sin dificultades de la ecuación (2) y las condiciones iniciales (3). Así, considerando i=2 en la ecuación (6), y sustituyendo (7), se tiene que:

$$X_2(t) = X_1(0) b_{1,2}^{\text{eff}} \lambda_1^{\text{eff}} e^{-\lambda_2^{\text{eff}} t} \int_0^t e^{-(\lambda_1^{\text{eff}} - \lambda_2^{\text{eff}})t'} \, dt' \tag{8}$$

En este primer caso, podemos observar que (8) contiene una diferencia de constantes de decaimiento efectivas, entre el isótopo 1 y el isótopo 2. Si se diera el caso que el isótopo 1 es igual al isótopo 2, es claro que esa resta sería igual a cero, y el resultado de la integral sería simplemente t. Repitiendo este proceso para el caso i=3, se obtiene:

$$X_3(t) = X_1(0) b_{1,2}^{\text{eff}} \lambda_1^{\text{eff}} b_{2,3}^{\text{eff}} \lambda_2^{\text{eff}} e^{-\lambda_3^{\text{eff}} t} \int_0^t e^{-(\lambda_2^{\text{eff}} - \lambda_3^{\text{eff}})t_2} dt_2 \int_0^{t_2} e^{-(\lambda_1^{\text{eff}} - \lambda_2^{\text{eff}})t_1} dt_1 \tag{9}$$

Nuevamente aparece una diferencia de constantes de decaimiento efectivas, esta vez entre la que corresponde al isótopo 2 y la que corresponde al isótopo 3. Repitiendo este procedimiento para el caso i=n, y utilizando el Teorema de Fubini[23], se tiene que:

$$X_n(t) = X_1(0) \prod_{j=1}^{n-1} b_{j,j+1}^{\text{eff}} \lambda_j^{\text{eff}} e^{-\lambda_n^{\text{eff}} t} I(n) \tag{10}$$

Con $I(n)$ igual a:

$$\int_0^t \int_0^{t_{n-1}} \cdots \int_0^{t_2} e^{-\left[(\lambda_{n-1}^{\text{eff}} - \lambda_n^{\text{eff}})t_{n-1} + (\lambda_{n-2}^{\text{eff}} - \lambda_{n-1}^{\text{eff}})t_{n-2} + \cdots + (\lambda_1^{\text{eff}} - \lambda_2^{\text{eff}})t_1\right]} \, dt_{n-1} \ldots dt_1 \tag{11}$$

Una parte del presente trabajo consistió en desarrollar y analizar la ecuación (11), en la cual se observaron varias propiedades que resultan útiles para las cadenas cíclicas. En primer lugar, dicha ecuación permite reducir todos los casos en los que existan elementos repetidos que a su vez sean consecutivos. De esta forma, es posible simplificar la sumatoria de la exponencial en (11), y disminuir la dificultad de la integral.

Por otro lado, mediante cambios de variable, es posible reordenar algunos términos que resulten similares, aun cuando no sean consecutivos. Sin embargo, debido a que las diferencias en (11) no están multiplicadas por la misma variable t_k, con $1 \leq k \leq n-1$, esta tarea no es del todo sencilla. Sin embargo, es posible realizar dicho procedimiento a través de un análisis combinatorio sobre las distintas configuraciones en las que pueden presentarse los elementos repetidos en las cadenas cíclicas.

Solución por transformada de Laplace

Como se mencionó anteriormente, existen varias formulaciones de la solución general, algunas de las cuales resultan más fáciles de programar que otras. Por ello, como una primera aproximación se tomó la decisión de utilizar la solución general obtenida por la transformada de Laplace para el presente estudio. Sin embargo, en el futuro, es posible que esta investigación se extienda para realizar una comparación con todas las soluciones generales.

El método de solución por Laplace fue utilizado por primera vez por Harry Bateman en 1910 De hecho, fue la primera vez que dicha técnica se utilizó para resolver ecuaciones diferenciales.[24] Esencialmente, se deben utilizar las siguientes relaciones:

$$\begin{cases} \mathcal{L}\{X_j\} = \tilde{x}_j = \int_0^{\infty} X_j(t)\, e^{-st} dt, \quad 1 \leq j \leq n \\ \mathcal{L}\left\{\frac{dX_j}{dt}\right\} = s\mathcal{L}\{X_j\} - X_j(0) = s\tilde{x}_j - X_j(0) \end{cases} \tag{12}$$

Luego, es necesario reescribir (2) de forma matricial, aplicar (5) y las condiciones iniciales dadas en (3):

$$\underbrace{\begin{bmatrix} \frac{dX_1}{dt} = & -\lambda_1^{\text{eff}} X_1 \\ \frac{dX_2}{dt} = & b_{1,2}^{\text{eff}} \lambda_1^{\text{eff}} X_1 - \lambda_2^{\text{eff}} X_2 \\ \cdots & \cdots \\ \frac{dX_n}{dt} = & b_{n-1,n}^{\text{eff}} \lambda_1^{\text{eff}} X_{n-1} - \lambda_n^{\text{eff}} X_n \end{bmatrix}}_{\text{Sistema (2) escrito de forma matricial}} \xrightarrow[\text{usando (5)}]{} \underbrace{\begin{bmatrix} s\tilde{x}_1 = & X_1(0) & -\lambda_1^{\text{eff}} \tilde{x}_1 \\ s\tilde{x}_2 = & b_{1,2}^{\text{eff}} \lambda_1^{\text{eff}} \tilde{x}_1 & -\lambda_2^{\text{eff}} \tilde{x}_2 \\ \cdots & \cdots & \\ s\tilde{x}_n = & b_{n-1,n}^{\text{eff}} \lambda_{n-1}^{\text{eff}} \tilde{x}_{n-1} & -\lambda_n^{\text{eff}} \tilde{x}_n \end{bmatrix}}_{\text{Usando las condiciones iniciales dadas en (3)}} \tag{13}$$

Después de múltiples sustituciones, es posible obtener de (13) la siguiente expresión:

$$\tilde{x}_n = X_1(0)\prod_{k=1}^{n-1} b_{k,k+1}^{\text{eff}}\lambda_k^{\text{eff}}\,\frac{1}{\left(s+\lambda_1^{\text{eff}}\right)\left(s+\lambda_2^{\text{eff}}\right)\ldots\left(s+\lambda_n^{\text{eff}}\right)} \tag{14}$$

Luego, para resolver el sistema solo es necesario encontrar la transformada inversa de (14). En este punto es donde Harry Bateman supuso que todas las constantes de decaimiento efectivas eran distintas. Es decir, que el conjunto de $-\lambda_i^{\text{eff}}, 1\leq i\leq n$ eran raíces simples del polinomio de variable s que está en el denominador de (14). En tal caso, es posible usar fracciones parciales y escribir:

$$\frac{1}{\left(s+\lambda_1^{\text{eff}}\right)\left(s+\lambda_2^{\text{eff}}\right)\ldots\left(s+\lambda_n^{\text{eff}}\right)} = \sum_{i=1}^{n}\frac{a_i}{s+\lambda_i^{\text{eff}}},\ \text{con}\ a_i = \prod_{\substack{j=1\\ j\neq i}}^{n}\frac{1}{\lambda_j^{\text{eff}}-\lambda_i^{\text{eff}}} \tag{15}$$

De esta forma se tiene que la transformada inversa de (14) es igual a:

$$X_n = \mathcal{L}^{-1}\{\tilde{x}_n\} = X_1(0)\prod_{k=1}^{n-1} b_{k,k+1}^{\text{eff}}\lambda_k^{\text{eff}}\sum_{i=1}^{n}\prod_{\substack{j=1\\ j\neq i}}^{n}\frac{1}{\lambda_j^{\text{eff}}-\lambda_i^{\text{eff}}}\mathcal{L}^{-1}\left\{\frac{1}{s+\lambda_i^{\text{eff}}}\right\} \tag{16}$$

Considerando que $L^{-1}\ \{1/(s+\lambda_i^{\text{eff}})\}=e^{-\lambda_i^{\text{eff}}t}$, es posible verificar que (16) es igual a (4). Sin embargo, para el caso general el paso (15) es más complejo, puesto que el polinomio no necesariamente tiene raíces simples, ya que, si hay elementos repetidos, algunos de los binomios estarán elevados a una potencia igual al número de veces en que se presenta dicha repetición. En este caso, se considerará que de los n isótopos que componen (1), hay un total de m distintos, donde claramente $m\leq n$.

Luego se considerará el conjunto de números $k_1,k_2,\ldots,k_m$ para denotar el número de veces que los isótopos aparecen repetidos. En este sentido el isótopo h, con $1\leq h\leq m$, aparecerá k_h veces repetido en (1). Es posible verificar sin dificultad que $k_1+k_2+\ldots+k_m=n$. Con estas nuevas condiciones la ecuación (14) se transforma en:

$$\tilde{x}_n = \frac{X_1(0)\prod_{l=1}^{m}\left(b_{l,l+1}^{\text{eff}}\lambda_l^{\text{eff}}\right)^{k_l}}{{}_{n-1,n}b^{\text{eff}}\,{}_{n-1}\lambda_{n-1}^{\text{eff}}}\,\frac{1}{\left(s+\lambda_1^{\text{eff}}\right)^{k_1}\left(s+\lambda_2^{\text{eff}}\right)^{k_2}\ldots\left(s+\lambda_m^{\text{eff}}\right)^{k_m}} \tag{17}$$

Donde ${}_{n-1,n}b^{\text{eff}}$ y ${}_{n-1}\lambda^{\text{eff}}$ representan el coeficiente de ramificación y la constante de decaimiento del isótopo que se encuentra en la posición n en la sucesión dada en (1). La parte derecha de (17) también puede reescribirse usando fracciones parciales[25]:

$$\frac{1}{\left(s+\lambda_1^{\text{eff}}\right)^{k_1}\left(s+\lambda_2^{\text{eff}}\right)^{k_2}\ldots\left(s+\lambda_m^{\text{eff}}\right)^{k_m}} = A(s) = \sum_{j=1}^{n}\sum_{i=1}^{k_j}\frac{c_{j,i}}{\left(s+\lambda_j^{\text{eff}}\right)^{i}}, \tag{18}$$

Donde:

$$c_{j,i} = \lim_{s\to-\lambda_j^{\text{eff}}}\begin{cases} A(s)\left(s+\lambda_j^{\text{eff}}\right)^{k_j}, & i=k_j \\ \dfrac{1}{j!}\dfrac{d^j}{ds^j}\left[A(s)\left(s+\lambda_j^{\text{eff}}\right)^{k_j}\right], & 1\leq i\leq k_j-1 \end{cases} \tag{19}$$

Con estas definiciones la transformada inversa de (17) se reduce a:

$$X_n(t) = \frac{X_1(0)\prod_{l=1}^{m}\left(b_{l,l+1}^{\text{eff}}\lambda_l^{\text{eff}}\right)^{k_l}}{{}_{n-1,n}b^{\text{eff}}\,{}_{n-1}\lambda_{n-1}^{\text{eff}}}\sum_{j=1}^{n}\sum_{i=1}^{k_j}c_{j,i}\mathcal{L}^{-1}\left\{\frac{1}{\left(s+\lambda_j^{\text{eff}}\right)^{i}}\right\} \tag{20}$$

Finalmente, se considerará la siguiente igualdad:

$$\mathcal{L}^{-1}\left\{\frac{1}{(s+\lambda_j^{\text{eff}})^i}\right\} = \frac{e^{-\lambda_j^{\text{eff}} t} t^{i-1}}{\Gamma(i)} \tag{21}$$

Donde Γ representa a la función gamma. Usando (21), la solución general por el método de Laplace es igual a:

$$X_n(t) = \frac{X_1(0)\prod_{l=1}^{m}\left(b_{l,l+1}^{\text{eff}}\lambda_l^{\text{eff}}\right)^{k_l}}{{}_{n-1,n}b^{\text{eff}}{}_{n-1}\lambda_{n-1}^{\text{eff}}}\sum_{j=1}^{n}\sum_{i=1}^{k_j} c_{j,i}\frac{e^{-\lambda_j^{\text{eff}} t} t^{i-1}}{\Gamma(i)} \tag{22}$$

Análisis algorítmico de las soluciones

A pesar de que no resulta evidente, las ecuaciones (10) y (22) son equivalentes. Gracias a la forma en cómo fueron obtenidas y por la manera en que fueron expresadas, cada una de ellas tiene ventajas y desventajas que es necesario analizar.

Las ventajas de la ecuación (10), están relacionadas con el tipo de integral que se debe resolver en cada paso del proceso recursivo. Cuando dos constantes de decaimiento son iguales, entonces el proceso de integración se reduce a encontrar integrales de la forma $t^k e^{-\lambda_i^{\text{eff}} t}), 1 \le i \le n, 0 \le k \le n$. Estas integrales han sido estudiadas por años, y existen tablas con fórmulas que permiten calcularlas. Su principal desventaja, por otro lado, es que el proceso de comparación entre las constantes de decaimiento efectivas es lento, porque se hace por pares.

Por su parte, la ecuación (22) realiza la comparación entre las constantes de decaimiento una sola vez y de manera rápida. Para llevar a cabo esta tarea, se determina cuántos elementos repetidos hay, y por lo tanto se calculan los números k_m, eliminando la necesidad de analizar por pares, como sucede en la ecuación (10).

Sin embargo, la principal desventaja se relaciona con la determinación de los coeficientes $c_{j,i}$, que involucran el cálculo de derivadas de orden superior de la función racional que está en la parte derecha de la ecuación (17). Esta dificultad en cuanto al cálculo de derivadas no solo se presenta en la solución por transformada de Laplace, sino también en la solución propuesta por Dreher y [8] por Slodička y Balážová.[22]

Si bien es posible aproximar dichas derivadas, como lo propone Dreher[8], también resulta conveniente buscar alternativas para construir las funciones de forma exacta, puesto que, a diferencia de muchos otros problemas en ingeniería nuclear, en este caso contamos con soluciones analíticas exactas, como puede observarse en (10) y en (22).

En este punto es donde interviene el cálculo simbólico, una herramienta computacional que permite construir funciones matemáticas que luego pueden ser incorporadas a un código, para su evaluación. A diferencia de los métodos numéricos, que aproximan valores de funciones, el cálculo simbólico permite aproximar funciones con otras funciones, o calcularlas de forma exacta. Esta herramienta computacional es sumamente útil cuando se utilizan fórmulas recursivas, como la dada en (10), porque permite la construcción de funciones a través de bucles.

Esta construcción por bucles es igualmente útil cuando se calculan derivadas de orden superior, puesto que en esencia se trata de derivar una función, y luego definir esta "derivada" como una nueva función, y volver a derivarla, lo que esencialmente también es un bucle. Así, la comparación de la presente investigación se reduce a determinar qué es más rápido, a nivel de cálculo simbólico, calcular integrales o calcular derivadas.

CÁLCULO SIMBÓLICO

Es necesario distinguir entre dos tipos de cálculos: los basados en métodos numéricos, y los que se basan en métodos analíticos, o por manipulación de fórmulas. Existen diferencias esenciales entre estas dos metodologías. En primer lugar, para el cálculo simbólico es necesario conocer la solución exacta, y realizar ciertas manipulaciones mientras que en el cálculo numérico no. Así mismo, este último cálculo involucra un mayor número de evaluaciones, que en ocasiones se traducen a pasos recursivos, mientras que el simbólico suele necesitar de la sola evaluación de una fórmula analítica.

Así mismo el cálculo numérico permite aproximar con un alto grado de precisión una operación o ecuación, a través de una serie de operaciones numéricas de tipo algebraico o funcional, sin necesidad de conocer la solución exacta del problema. Mientras que el cálculo simbólico, requiere esencialmente la manipulación de expresiones y el análisis de las relaciones entre ellas.

Cuando el cálculo simbólico se incorpora a un programa de cómputo, es que se habla de computación algebraica[26]. Si bien el cálculo numérico es quien dominó desde el principio la computación científica, el cálculo simbólico también se fue desarrollando a través varios sistemas algebraico computacionales (denotados por las siglas CAS por su acrónimo en inglés Computer Algebra System), encontrándose entre los más destacables Mathematica (donde se incluye a Wolfram Alpha)[27], Maple[28], Maxima[29] y SymPy[30] entre otros.

Operaciones comunes en la formulación integral

A través de un estudio de la ecuación (11) se encontraron las siguientes operaciones recurrentes:

$$\int_0^t e^{-\left(\lambda_i^{\text{eff}}-\lambda_j^{\text{eff}}\right)t_1}dt_1 = \begin{cases} -\dfrac{1}{\lambda_i^{\text{eff}}-\lambda_j^{\text{eff}}}\left[e^{-\left(\lambda_i^{\text{eff}}-\lambda_j^{\text{eff}}\right)t}-1\right], & \lambda_i^{\text{eff}} \neq \lambda_j^{\text{eff}} \\ t & \lambda_i^{\text{eff}} = \lambda_j^{\text{eff}} \end{cases} \tag{23}$$

La operación descrita en (23) puede abreviarse a través de una función binaria dada por $\omega(\lambda_i^{\text{eff}},\lambda_j^{\text{eff.}})$La siguiente operación común, consiste en la multiplicación de $\omega(\lambda_i^{\text{eff}},\lambda_j^{\text{eff}})$ por una función exponencial:

$$\varphi\left(\lambda_i^{\text{eff}},\lambda_j^{\text{eff}}\right) = e^{-\lambda_j^{\text{eff}}t}\omega\left(\lambda_i^{\text{eff}},\lambda_j^{\text{eff}}\right) = \begin{cases} -\dfrac{1}{\lambda_i^{\text{eff}}-\lambda_j^{\text{eff}}}\left[e^{-\lambda_i^{\text{eff}}t}-e^{-\lambda_j^{\text{eff}}t}\right], \\ e^{-\lambda_j^{\text{eff}}t}t \end{cases} \tag{24}$$

Si se avanza en una integración adicional, se tiene que la siguiente expresión

$$e^{-\lambda_k^{\text{eff}}} \int_0^t e^{\lambda_k^{\text{eff}} t_1} \varphi(\lambda_i^{\text{eff}}, \lambda_j^{\text{eff}})\, dt_1 \tag{25}$$

Es igual a:

$$\begin{cases} -\dfrac{1}{\lambda_i^{\text{eff}} - \lambda_j^{\text{eff}}} \left[\varphi(\lambda_i^{\text{eff}}, \lambda_k^{\text{eff}}) - \varphi(\lambda_j^{\text{eff}}, \lambda_k^{\text{eff}})\right], & \lambda_i^{\text{eff}} \neq \lambda_j^{\text{eff}} \\ e^{-\lambda_k^{\text{eff}} t} \displaystyle\int_0^t e^{-\left(\lambda_j^{\text{eff}} - \lambda_k^{\text{eff}}\right) t_1} t_1 dt_1 & \lambda_i^{\text{eff}} = \lambda_j^{\text{ef}} \end{cases} \tag{26}$$

Finalmente, la última expresión identificada y de la que se obtuvo una fórmula general está relacionada con el cálculo del siguiente producto:

$$e^{-\lambda_y^{\text{eff}} t} \int_0^t t_1^n e^{-(\lambda_x^{\text{eff}} - \lambda_y^{\text{eff}}) t_1}\, dt_1 \tag{27}$$

El cual es igual a:

$$= -\frac{t^n e^{-\lambda_x^{\text{eff}} t}}{\left(\lambda_x^{\text{eff}} - \lambda_y^{\text{eff}}\right)} + \frac{n!\left(e^{-\lambda_x^{\text{eff}} t} - e^{-\lambda_y^{\text{eff}} t}\right)}{\left(\lambda_x^{\text{eff}} - \lambda_y^{\text{eff}}\right)^{n+1}} + \sum_{k=1}^{n-1}\left[-\frac{n!\, t^{n-k} e^{-\lambda_x^{\text{eff}} t}}{(n-k)!\left(\lambda_x^{\text{eff}} - \lambda_y^{\text{eff}}\right)^{k+1}}\right] \tag{28}$$

Construcción de una librería de cálculo simbólico

Como se mencionó al principio de esta sección, existen varios CAS con los que se puede llevar a cabo la resolución de (11) y (22). Al principio de esta investigación se seleccionó la paquetería SymPy, que es una biblioteca de licencia libre para el lenguaje Python[30].

Sin embargo, durante el desarrollo del algoritmo surgieron algunas dificultades con el uso de SymPy, relacionadas con la forma en cómo este sistema CAS realizaba algunas factorizaciones y simplificaciones. Si bien era posible superar estos problemas con un estudio más profundo de la forma en cómo se definieron algunas funciones y clases en dicha librería, se consideró más apropiado desarrollar un sistema CAS propio, y que solo dependiese de variables del tipo "string".

Esto último, permitiría que el sistema de cálculo simbólico desarrollado, pudiera implementarse en otros códigos sin muchas dificultades. La discusión que se presentará en la siguiente sección, proporciona los aspectos más relevantes de la librería desarrollada para la resolución de la ecuación (11), siendo este procedimiento similar al utilizado para resolver la ecuación (22).

Funciones y paréntesis

Para la resolución de la ecuación (11) se necesitan 7 funciones. Cuatro de ellas son operaciones binarias, y tres aceptan un solo argumento. Estas funciones se representarán por una letra, o un par de estas, así como un par de paréntesis para denotar o "encerrar" su argumento. En la tabla II se listan las principales funciones, con la operación matemática a la que corresponden.

La mayoría de los lenguajes de programación pueden manipular variables del tipo "string" (también llamadas "character"), así que, aunque la siguiente descripción corresponde a la manipulación de este tipo de variables que se hace en el lenguaje de programación Python 3.5, puede extenderse a otros lenguajes sin muchas dificultades.

Para representar las constantes de decaimiento efectivas se utilizará la letra "l" acompañada de un número. Así, por ejemplo, la constante λ_4^{eff} quedará representada por "l4". No se utilizará ninguna variable para representar los coeficientes de ramificación efectivos, puesto que la ecuación (11) no las contiene.

Tabla II. Representación de las principales funciones con cálculo simbólico y sus propiedades.

Función	Representación Simbólica	Argumentos	Propiedades
a-b	$d(a,b)$	2	$d(a,b)$=-$d(b,a)$
a+b	$S(a,b)$	2	$S(a,b)=S(b,a)$ $S(a,-b)=d(a,b)$
$a \cdot b$	$M(a,b)$	2	$M(a,b)=M(b,a)$ $M(d(a,b),c)=d(M(a,c),M(b,c))$ M(S(a,b),c)=S(M(a,c),M(b,c) M(a,1)=a
e^{at}	$ex(a)$	1	M(ex(a),exp(b))=expn(S(a,b)) ex(d(lx,ly))=1,"si" lx=ly
t^a	$t(a)$	1	M(t(a),t(b))=t(S(a,b))
1/(a-b)	d_i (a,b)	2	d_i (a,b)=-d_i (b,a)
$\int^1_0 a dt$	$G(a)$	1	G(M(a,b))=M(a,G(b)) si a es constante G(S(a,b))=S(G(a),G(b)) G(d(a,b)=d(G(a),G(b))

Es importante dotar a las funciones de ciertas propiedades, por ejemplo, la de conmutación M(a,b)=M(b,a), puesto que esto permite al programa simplificar algunas operaciones para resolver las ecuaciones. El algoritmo comienza con la siguiente ecuación:

$$e^{-\lambda_2^{eff} t} \int_0^t e^{-(\lambda_1^{eff}-\lambda_2^{eff})t'} dt' \tag{29}$$

Que en cálculo simbólico se escribe como:

$$M\left(ex(-l2), G\left(ex(-d(l2,l1))\right)\right) \tag{30}$$

A continuación, se deben buscar todas las integrales que existen en la expresión (30). Dado que esta expresión se almacenará en una variable del tipo "string", para realizar dicha búsqueda solo hace falta identificar en dicha variable cuántas veces aparece la letra "G".

Este procedimiento se puede realizar en Python 3.5 a través del método "count". Una vez localizadas las integrales, deben resolverse una a una mediante iteraciones. En primer lugar, se debe identificar el argumento de estas, lo cual se realiza mediante una función que analiza la cantidad de paréntesis izquierdos y derechos que hay. Nuevamente, esta tarea puede realizarse mediante el método "count". Luego, se deben simplificar los argumentos de las integrales, lo que es necesario prácticamente en un solo caso: cuando las integrales contienen exponenciales de diferencias. Justamente, esta operación se relaciona con el análisis de las constantes de decaimiento efectivas de las cadenas cíclicas. Así, la función integral tiene las siguientes posibilidades en base a su argumento:

$$\begin{cases} a\int_0^1 g(t)dt & \text{si } f(t) = ag(t) \\ t & \text{si } f(t) = e^{-(\lambda_x^{\text{eff}}-\lambda_y^{\text{eff}})t},\ \text{y } \lambda_x^{\text{eff}} = \lambda_y^{\text{eff}} \\ -\dfrac{1}{\lambda_x^{\text{eff}}-\lambda_y^{\text{eff}}}\left[e^{-(\lambda_x^{\text{eff}}-\lambda_y^{\text{eff}})t}-1\right] & \text{si } f(t) = e^{-(\lambda_x^{\text{eff}}-\lambda_y^{\text{eff}})t} \text{ y } \lambda_x^{\text{eff}} \neq \lambda_y^{\text{eff}} \end{cases} \quad (31)$$

Que en términos de cálculo simbólico pueden escribirse:

$$\begin{cases} M\left(a, G(b)\right) & \text{con } G(M(a,b)) \text{ y } a \text{ constante} \\ t(1) & \text{con } G\left(ex\left(-d(lx,ly)\right)\right) \text{ y } lx = ly \\ M\left(-d_i(lx,ly), d\left(ex\left(-d(lx,ly)\right),1\right)\right) & \text{con } G(ex\left(-d(lx,ly)\right) \text{ y } lx \neq ly \end{cases} \quad (32)$$

Existe una posibilidad adicional que no se ha incluido, debido a que requiere de la definición de productoria y sumatoria, que no se incluyeron en la Tabla II por simplicidad. Sabemos que $\lambda_1^{\text{eff}} \neq \lambda_2^{\text{eff}}$, de esta forma (30) se transforma en:

$$M\left(ex(-l2), M\left(-d_i(l1,l2), d(ex(-d(l1,l2)),1)\right)\right) \quad (33)$$

Si se utiliza la propiedad distributiva con respecto a la resta en el segundo producto se tendría

$$d\left(M\left(-d_i(l1,l2), ex(-d(l1,l2))\right), -d_i(l1,l2)\right) \quad (34)$$

Sustituyendo esto en (33), se tiene finalmente que:

$$\left(M(-d_i(l1,l2), ex(-l1)), M(-d_i(l1,l2), ex(-l2))\right) \quad (35)$$

Es importante notar que la última expresión se obtuvo solamente aplicando las propiedades de la Tabla II a la ecuación (30), sin ninguna manipulación matemática normal. Si transformamos la ecuación (35) a su equivalente matemática, tenemos:

$$-\frac{e^{-\lambda_1^{\text{eff}}t}}{\lambda_1^{\text{eff}}-\lambda_2^{\text{eff}}} + \frac{e^{-\lambda_2^{\text{eff}}t}}{\lambda_1^{\text{eff}}-\lambda_2^{\text{eff}}} \quad (36)$$

Que es justamente el resultado de la ecuación matemática en (29). Este ejemplo

muestra cómo puede resolverse uno de los pasos de la ecuación (11), usando cálculo simbólico. Es claro que se necesitaron pasos de identificación, sustitución y la aplicación de varias propiedades, que podrían volver este procedimiento difícil si se realizara manualmente, pero por fortuna dichas tareas pueden realizarse sin dificultad en un algoritmo computacional.

Si la expresión en (35) se almacena en una variable del tipo "*string*" llamada Ψ, para calcular el siguiente paso en la solución de (11), solo hace falta resolver la siguiente expresión de cálculo simbólico:

$$M\left(ex(-l3), G\left(M(\Psi, ex(l3))\right)\right) \tag{37}$$

La cual es el equivalente de la ecuación (6). En este punto, el algoritmo para la resolución de las integrales que se describió en líneas anteriores vuelve a efectuarse. Un proceso similar se llevó a cabo para la construcción y resolución de las derivadas definidas en la ecuación (19), y por lo tanto para la resolución de la ecuación (22). Una vez construidas las soluciones, se desarrolló un algoritmo de "traducción", que permite que expresiones de cálculo simbólico como la mostrada en (53) puedan incorporarse a un código y evaluarse.

Comparación

Existen dos formas de comprobar la utilidad del cálculo simbólico en la resolución de la ecuación general de Bateman. La primera de ellas consiste en resolver un conjunto de cadenas cíclicas con un algoritmo que contenga cálculo simbólico, y con otro que no lo haga, analizando el tiempo de cómputo que ambos códigos requieren para obtener resultados similares. Sin embargo, esta comparación es compleja, porque la mayor parte de los algoritmos que implementan soluciones generales, ocupan aproximaciones de derivadas o de integrales. Por lo tanto, la comparación deberá incluir un análisis de qué tan precisa puede ser una solución que no use cálculo simbólico en comparación con una que sí lo use.

Dicho proceso requiere de un análisis más profundo que compare las soluciones generales con las que usan aproximaciones, un tópico que formará parte de un futuro trabajo. Por ahora, el análisis que puede realizarse en términos computacionales se reduce a verificar que las soluciones obtenidas para un conjunto de cadenas cíclicas sean correctas, y analizar cuál de ellas es más eficiente en cuanto al tiempo de cómputo.

Resulta claro que en el futuro se construirán con cálculo simbólico el resto de las soluciones, para determinar cuál de ellas resulta más conveniente de ser incluida en un algoritmo computacional. Por lo anterior, el presente análisis consistió en tres pasos. Primero, usando cálculo simbólico, se resuelve la ecuación general de Bateman usando la solución de Laplace y la solución integral. La estructura que se resolvió es similar a la mostrada en (25). Luego, se verificaba que dichas soluciones fueran idénticas. Finalmente, se consideró el tiempo que le tomó a cada solución construir la ecuación. La Figura 1 contiene una gráfica comparativa en términos del tiempo de cómputo que necesitó cada una de las soluciones para construir la ecuación general de Bateman, variando la longitud de la cadena cíclica.

Para esta comparación se utilizó el lenguaje de programación Python 3.5, y se utilizó un equipo de cómputo con un procesador Intel i-7 de sexta generación, con 2.6GHz de velocidad de reloj base.

Nótese que no se proponen tiempos de ejecución, porque esto está fuertemente condicionado por el hardware y arquitectura de un equipo, optándose mejor por ciclos (definidos fácilmente como una base normalizada). Como se puede observar en la figura 1, el método de formulación integral es más veloz que el método basado en la transformada de Laplace.

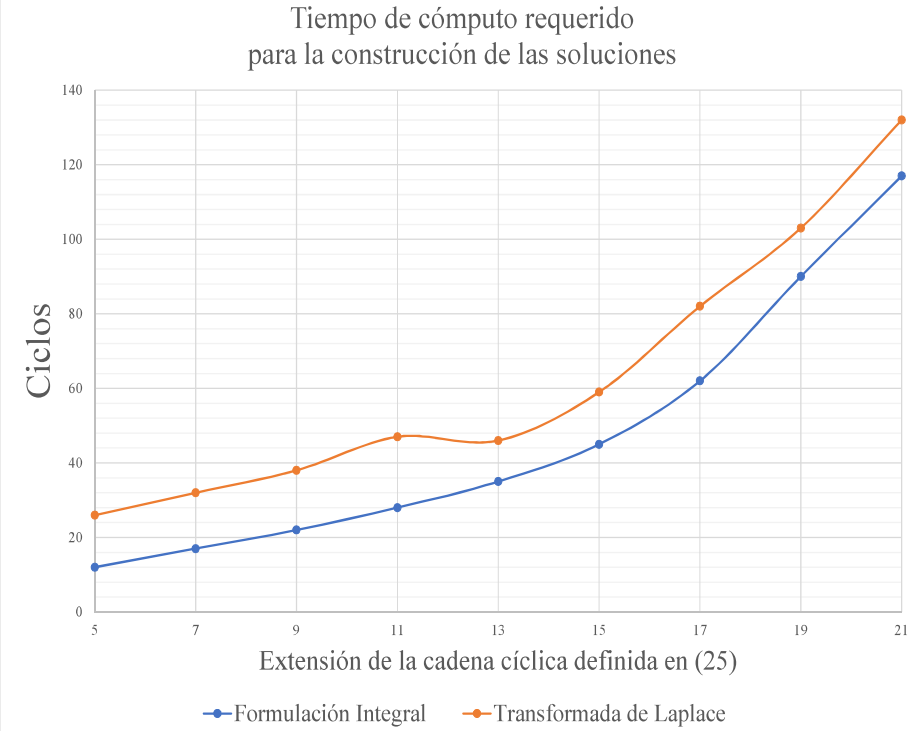

Fig. 1. Ciclos de cómputo para el método de Formulación Integral, y para el método de Transformada de Laplace, en función de la extensión de la cadena cíclica con la estructura dada en (25).

Esto puede explicarse en cuanto a que es más rápido calcular integrales del tipo exponencial, que el cálculo de derivadas de funciones racionales. Sin embargo, es posible observar que la diferencia entre el tiempo se mantiene aproximadamente constante.

Por otro lado, es posible observar que el tiempo de ambos métodos aumenta de forma de forma considerable para los últimos puntos de la gráfica. Luego de una inspección se observó que esto se debe a la presencia de una variable t^k, cuyo exponente incrementa. Esto a su vez incrementa el número de términos, como puede apreciarse en la ecuación (27), y se traduce a una derivada de orden mayor en el caso de la ecuación (19).

CONCLUSIONES

En el presente trabajo se construyeron dos soluciones generales de la ecuación de Bateman usando cálculo simbólico. Para ello se identificaron las principales propiedades de la formulación integral, y se realizó un análisis del cálculo de derivadas de alto orden relacionadas con el método por transformada de Laplace.

Usando variables del tipo "string" fue posible construir una librería CAS especializada en la resolución de integrales y derivadas relacionadas con la ecuación de Bateman. Al únicamente utilizar funciones de este tipo de variables, la metodología desarrollada puede extenderse fácilmente a otros lenguajes de programación.

Al realizarse una comparación entre la construcción de la solución basada en el tiempo real, y la solución basada en la transformada de Laplace, se concluyó que el primer proceso es más rápido, y que dicha diferencia entre los tiempos de cómputo permanece aproximadamente constante. En ambos casos, el tiempo incrementa de forma notoria cuando aumenta la longitud de la cadena cíclica.

Es posible concluir del presente trabajo, que a través del cálculo simbólico pueden aprovecharse las soluciones generales que diferentes autores han publicado, sin necesidad de que algunas operaciones relacionadas con ellas sean aproximadas, y por lo tanto pierdan exactitud. Adicionalmente, el cálculo simbólico permite construir las soluciones en tiempo real, lo que permitirá que los códigos tengan flexibilidad para simular cadenas cíclicas más complejas.

AGRADECIMIENTOS

Al Consejo Nacional de Ciencia y Tecnología por brindar el apoyo económico a C.A. Cruz para la realización de este trabajo que forma parte de su investigación doctoral. Los autores agradecen el apoyo financiero recibido del proyecto estratégico No. 212602: "AZTLAN Platform: Desarrollo de una plataforma mexicana para el análisis y diseño de reactores nucleares", del Fondo Sectorial de Sustentabilidad Energética CONACYT - SENER.

A la UNAM, por el apoyo otorgado a través del proyecto PAPIIT-IN115517.

REFERENCIAS

1. Stammler, R. J. J., Abbate, M. J., Methods of Steady-State Reactor Physics in Nuclear Design, Academic Press Inc, Estados Unidos (1983).
2. Rutherford, E., Radio-Activity, Cambridge Physical Series, Cambridge, Inglaterra (1904).
3. Rubinson, W., "The Equations of Radioactive Transformation in a Neutron Flux", The Journal of Chemical Physics, Vol. 17, (6), p. 542-547 (1949).
4. Bateman, H., "Solution of a System of Differential Equations Occurring in the Theory of Radioactive Transformations", Proceedings of the Cambridge Philosophical Society, Vol. 15, p. 423–427 (1910).

5. Isotalo, A., Computational Methods for Burnup Calculation with Monte Carlo Neutronics, Doctoral Dissertation. Aalto University Publication Series. Helsinki, Finland (2013).
6. Meyer, S., Schweildler, E., Radioaktivität, Springer Fachmedein Wiesbaden GmbH. p. 56-57 (1927).
7. Cetnar, J., "General Solution of Bateman Equations for Nuclear Transmutations", Annals of Nuclear Energy, Vol. 33, p. 640–645 (2006).
8. Dreher, R., "Modified Bateman Solution for Identical Eigenvalues", Annals of Nuclear Energy, Vol. 53, p. 427-438 (2013).
9. Isotalo, A. E., Aarnio, P. A., "Higher order methods for burnup calculations with Bateman Solutions", Annals of Nuclear Energy, Vol. 38, p. 1987-1995 (2011).
10. Wilson, P. P. H., ALARA: Analytic and Laplacian Adaptive Radioactivity Analysis, Doctoral dissertation, Fusion Technology Institute, University of Wisconsin, Madison Wisconsin, USA (1999).
11. Clayton, D. D., Fowler, W. A., Hull, T. E., Zimmerman, B. A., "Neutron Capture Chains in Heavy Element Synthesis", Annals of Physics, Vol. 12(3), p.331-408 (1961).
12. Bell, M. J., "ORIGEN- The ORNL Isotope Generation and Depletion Code", ORNL-4628, Oak Ridge National Laboratory (1973).
13. Newman, M. J., "S-Process Studies: The Exact Solution", The Astrophysical Journal, Vol. 219, p. 676-689 (1978).
14. Tasaka, K., "DCHAIN 2: A Computer Code for Calculation of Transmutation of Nuclides", Japan Atomic Energy Research Institute. Tokyo, Japan (1980).
15. Miles, R. E., "An Improved Method for Treating Problems Involving Simultaneous Radioactive Decay, Buildup, and Mass Transfer", Nuclear Science and Engineering, Vol. 79 (2), p. 239-245 (1981).
16. Raykin, M. S, Shlyakhter, A.I., "Solution of Nuclide Burnup Equations Using Transition Probabilities", Nuclear Science and Engineering, Vol. 102, p. 54-63 (1989).
17. Blaauw, M., "A Versatile Computer Algorithm for Linear First-order Equations Describing Compartmental Models with Backward Branching", Applied Radiation and Isotopes, Vol. 44 (9), p. 1225-1229 (1993).
18. Vukadin, Z., "Solution of Depletion Chain Equations Using Nonsingular Bateman Coefficients", Nuclear Science and Engineering, Vol. 117 (2), p. 121-125 (1994).
19. Pommé, S. G., Hardeman, F. E. M. C., Robouch, P. B., Etxebarria, N., De Corte, F. A., De Wispelaere, A. H. M. J., Sluijs, R. v., Simonits, A. P., "General Activation and Decay Formulas and Their Application in Neutron Activation Analysis with "k" _0 Standardization. Analytical Chemistry", Vol. 68(24), p. 4326-4334 (1996).

20. Mirzadeh, S., Walsh, P., "Numerical Evaluation of the Production of Radionuclides in a Nuclear Reactor (Part I)", Applied Radiation and Isotopes, Vol. 49(4), p. 379-382 (1998).
21. Popovic', J., "Derivation of Laplace Transform for the General Disposition Deconvolution Equation in Drug Metabolism Kinetics", Experimental and Toxicology Pathology, Vol. 51, p. 409-411 (1999).
22. Slodička, M., Balážová, A., "Singular Value Decomposition Method for Multi-species First-order Reactive Transport with Identical Decay Rates", Transport in Porous Media, Vol. 73(2), p. 161-172 (2008).
23. Marsden, J. E., Elementary Classical Analysis, W. H. Freeman and Company, University of California, Berkeley, Estados Unidos, p. 302-306 (1976).
24. Deakin, M. A., "The Development of the Laplace Transform, 1737-1937 II. Poincaré to Doetsch, 1880-1937", Archive for History of Exact Sciences, Vol. 26(4), p. 351-381, (1982).
25. Boyd, S., Introduction to Signal & Systems, Stanford University, Course Notes. Lecture 5: Rational Functions and Partial Fractions Expansion, p. 25-29 (2003).
26. Davenport, J. H., Siret, Y., Tournier, E., Computer Algebra, Academic Press (1993).
27. David, C., Interactive Documents and Computer Algebra System: JOBAD and Wolfram|Alpha, Jacobs University Bremen, Bremen, Alemania (2010).
28. Char, B. W., Geddes, K., O., Gentleman, M., Gonnet, G. H., "The design of Maple: A compact, portable, and powerful computer algebra system", Conferencia Europea en Álgebra Computacional. EUROCAL, Berlin, Heidelberg (1983).
29. Rand, H. R., Introduction to Maxima, Departamento de Mecánica Teórica y Aplicada, Universidad de Cornell, Nueva York, Estados Unidos (2010).
30. Meurer, A., Smith, C. P., Paprocki, M., Čertík, O., Kirpichev, S., B., Rocklin, M., Kumar, A.,…, Scopatz, A., "SymPy, symbolic computing in Python", PeerJ Computer Science, Vol. 3(3), 103.