Procesos de Poisson y desintegración radiactiva

Andrea Real Blanco

Resumen. La desintegración radiactiva es un proceso esencialmente aleatorio que ocurre continuamente en la naturaleza; como cumple ciertas tres condiciones es posible utilizar variables aleatorias con distribución de Poisson para modelizar el proceso.
El objetivo de la PEC será programar funciones que simulen computacionalmente estos procesos de Poisson para la desintegración radiactiva y comparar los resultados obtenidos con aquellos dados por los modelos teóricos.
Para ello se trabajará con tres modelos o tipos de desintegración:

(1) la desintegración radiactiva de un conjunto de radionúclidos de la misma especie,

(2) una cadena de desintegración, y

(3) un esquema de desintegración ramificada.

Índice

1. Desintegración radiactiva de un conjunto de radionúclidos del mismo elemento

1.1. Ejercicio 1

1.1.1. Introducción

El ejercicio tiene como objetivo crear una simulación que reproduzca computacionalmente los resultados predichos por la ley de desintegración de las especies radiactivas y su decaimiento exponencial,

$$N(t) = N_0\, e^{-\lambda_0 t}.$$

Esta ley del decaimiento exponencial predice que partiendo de un conjunto inicial de N_0 núcleos, y con el paso del tiempo, estos se irán desintegrando de forma exponencial siguiendo una probabilidad por unidad de tiempo o constante de desintegración λ_0.

Se programará en lenguaje C el modelo descrito para la simulación y se compararán los resultados obtenidos con los predichos por el modelo teórico.

1.1.2. Metodología

El método propuesto y la parte del código correspondiente a su aplicación se explican a continuación:

1. Crear un array con N_0 celdas e inicializarlas de forma que $N_i = 1$ para todo i.

   Cada celda corresponderá a un núcleo y su estado se indica mediante el valor asociado, si es 1 el núcleo está activo y si es 0 ya se ha desintegrado. El bucle `while` permitirá repetir la simulación M veces.

```
while (k < M) {
    for (i = 0; i < N_0; i++) {
        N[i] = 1;
    }

    while (contador < N_0) {

    ...

    }
    k++;
    pasos = 0;
    j = 0;
    contador = 0;
}
```

2. En cada paso de la simulación se generará un número aleatorio que corresponderá a la posición de un núcleo en el array.
   - Si $N_i = 1$ el estado del núcleo pasará a ser 0 y avanzará un paso en la simulación.
   - Si $N_i = 0$ no se ejecuta ninguna acción y se avanza un paso en la simulación.

     El tiempo asociado a cada paso j de la simulación se obtiene a partir de $t_j = j(\lambda_0 N_0)^{-1}$.

Se utiliza otro bucle `while` para llevar cuenta de los núcleos desintegrados en cada simulación. Dentro de este bucle se solicita el número aleatorio `random` para el paso de la simulación, y podemos distinguir tres condicionales:

- El primero guarda los valores iniciales de la simulación en los array correspondientes, `Nj[0] = N_0` y `tiempo[0] = 0`.
- El segundo es el encargado de desintegrar el núcleo si la celda seleccionada con el número aleatorio contiene un núcleo activo.
- El tercer condicional añade los datos del número de núcleos sin desintegrar y el tiempo de simulación a los arrays correspondientes cada `deltaj` pasos.

```
while (contador < N_0) {
    random = rand() % N_0;
    if (pasos == 0) {
        tiempo[j] = pasos * pow(lambda * N_0, -1);
        Nj[j] += (N_0 - contador);
        j++;
    }
    if (N[random] == 1) {
        N[random] = 0;
        contador++;
    }
    pasos++;
    if (pasos % deltaj == 0) {
        tiempo[j] = pasos * pow(lambda * N_0, -1);
        Nj[j] += (N_0 - contador);
        j++;
    }
}
```

3. Cuando no quede ningún núcleo sin desintegrar se dará por finalizado el proceso.

   Con cada simulación los datos del array `Nj[]` se van sobrescribiendo y el promedio del conjunto de simulaciones es lo que se imprime en el archivo de datos.

1.1.3. Resultados y discusión

Los parámetros de la simulación se introducen como argumentos de la función `main` en el orden: N_0, λ_0, Δj y M.

Y los valores a utilizar son $N_0 = 1000$, $\lambda_0 = 0{,}13$ año^{-1}, $\Delta j = 100$ y $M = 1, 100, 10000$. Teniendo esto en cuenta, la solución teórica vendrá dada por la ecuación

$$N(t) = 1000 \cdot e^{-0{,}13t}.$$

Los resultados obtenidos en las simulaciones se recogen en las figuras 1.1 y 1.2.

- En la figura 1.1 podemos ver que al graficar en el intervalo de 0 a 60 años los puntos pertenecientes a los distintos conjuntos de M se solapan entre ellos y concuerdan notablemente con la solución teórica.

  A priori la información que se puede extraer de la gráfica es que las simulaciones parten de $N_0 = 1000$, el periodo de semidesintegración T se halla cerca de los 5 años y la relación variable dependiente-independiente se podría describir con un ajuste exponencial.

También podemos ver que los puntos azules, $M = 1$, no se ajustan tan bien como los rojos y verdes. Para encontrar la causa es necesario observar los datos en un intervalo más pequeño.

- En la figura 1.2 vemos cómo los puntos no coinciden tan bien como parecía inicialmente.

  Los puntos pertenecientes a $M = 1$ siguen un curioso patrón, esto se debe a que con una sola simulación los núcleos que van quedando siempre van a ser números enteros y puede necesitarse una gran cantidad de pasos hasta que sean seleccionados.

  Para los puntos de $M = 100$ y $M = 10000$ se hace un promedio. En consecuencia, cuantas más simulaciones se hagan más precisos serán los resultados; de ahí que los puntos verdes se ajusten mejor a la solución teórica.

  También es importante destacar que cuanto mayor es el número de simulaciones M más tardan en desintegrarse todos los núcleos, la causa de esto se debe a que al hacer la media se tienen en cuenta todas esas simulaciones con más pasos que el promedio y que contribuyen positivamente a la duración del proceso. Es más fácil tener estas anomalías para valores altos de M.

Para saber cuán tan bien se ajustan los datos de los distintos casos a la solución teórica podemos aplicarles una regresión exponencial, así llegamos a que la ecuación para cada uno es

$$N_1(t) = 889{,}4 \cdot e^{-0{,}12t}, \quad N_{100}(t) = 1001{,}2 \cdot e^{-0{,}13t} \quad \text{y} \quad N_{10000}(t) = 1000{,}4 \cdot e^{-0{,}13t}.$$

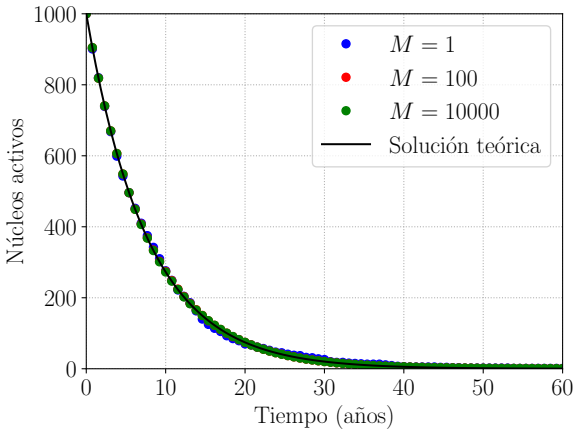

Figura 1.1

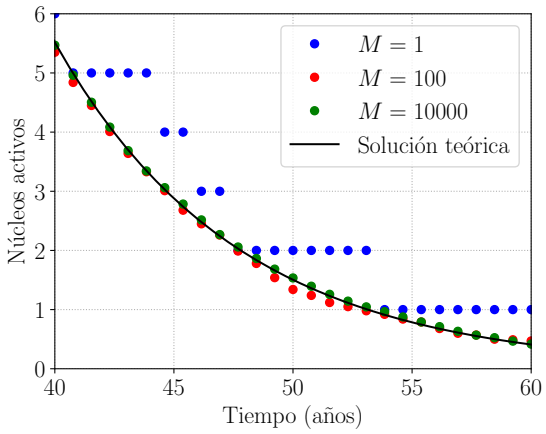

Figura 1.2

2. Cadena de desintegración

2.1. Ejercicio 2.1

Demostrar que cuando $N_0 \to \infty$ la evolución temporal está dada por

$$(\mathbf{a}) \quad \frac{\mathrm{d}N_P}{\mathrm{d}t} = -\lambda_P N_P \quad \text{y} \quad (\mathbf{b}) \quad \frac{\mathrm{d}N_H}{\mathrm{d}t} = \lambda_P N_P - \lambda_H N_H.$$

Datos: $\mathrm{d}t = [(\lambda_P + \lambda_H)N_0]^{-1}$, $P_P = \lambda_P/(\lambda_P + \lambda_H)$ y $P_H = \lambda_H/(\lambda_P + \lambda_H)$.

(**a**) El número de núcleos que se desintegran entre un paso de la simulación y el siguiente viene dado por la media de desintegraciones en ese paso multiplicada por la probabilidad de que se produzca

$$N_P(j) - N_P(j-1) = -\frac{N_P(j-1)}{N_0} \cdot P_P.$$

Como la distinción entre pasos j y tiempo t_j no afecta a las unidades, estos se pueden tratar de forma indistinta, lo que nos permite expresar la ecuación anterior de la forma:

$$N_P(t_j + \mathrm{d}t) - N_P(t_j) = -\frac{N_P(t_j)}{N_0} \cdot \frac{\lambda_P}{\lambda_P + \lambda_H}.$$

Sustituyendo con la relación $\mathrm{d}t = [(\lambda_P + \lambda_H)N_0]^{-1}$ y simplificando tenemos que

$$N_P(t_j + \mathrm{d}t) - N_P(t_j) = -N_P(t_j) \cdot (\lambda_P + \lambda_H)\,\mathrm{d}t \cdot \frac{\lambda_P}{\lambda_P + \lambda_H}$$

y, finalmente, cuando $N_0 \to \infty$ y $\mathrm{d}t \to 0$

$$\frac{N_P(t_j + \mathrm{d}t) - N_P(t_j)}{\mathrm{d}t} = -\lambda_P N_P(t_j) \implies \frac{\mathrm{d}N_P}{\mathrm{d}t} = -\lambda_P N_P.$$

(**b**) Similarmente,

$$N_H(j) - N_H(j-1) = -\frac{N_H(j-1)}{N_0} \cdot P_H + \frac{N_P(j-1)}{N_0} \cdot P_P,$$

donde el segundo término en el lado derecho se debe a que al tratarse de una cadena de desintegración se van añadiendo más núcleos hijos a medida que se desintegran los padres.

Al sustituir la expresión del paso temporal $\mathrm{d}t = [(\lambda_P + \lambda_H)N_0]^{-1}$ y simplificar

$$\begin{aligned} N_H(j) - N_H(j-1) = \\ - N_H(j-1)\frac{\lambda_H}{\lambda_P + \lambda_H} \cdot (\lambda_P + \lambda_H)\,\mathrm{d}t + N_P(j-1)\frac{\lambda_P}{\lambda_P + \lambda_H} \cdot (\lambda_P + \lambda_H)\,\mathrm{d}t \end{aligned}$$

llegamos a

$$\frac{N_H(t_j + \mathrm{d}t) - N_H(t_j)}{\mathrm{d}t} = \lambda_P N_P(t_j) - \lambda_H N_H(t_j).$$

Cuando $N_0 \to \infty$ la expresión se transforma en

$$\frac{\mathrm{d}N_H}{\mathrm{d}t} = \lambda_P N_P - \lambda_H N_H.$$

2.2. Ejercicio 2.2 - 2.3

2.2.1. Introducción

Una cadena de desintegración es un proceso en el que el decaimiento de un radionúclido hacia un núclido estable sucede en varias etapas o estados intermedios.

El caso más simple, el decaimiento de un radionúclido padre P en otro radionúclido hijo H y posteriormente en un elemento estable, tiene como soluciones a la evolución temporal

$$N_P(t) = N_{P,0}\, e^{-\lambda_P t}, \quad \text{y} \quad N_H(t) = N_{P,0}\, \frac{\lambda_P}{\lambda_H - \lambda_P} \left(e^{-\lambda_P t} - e^{-\lambda_H t} \right) + N_{H,0}\, e^{-\lambda_H t}.$$

Para la simulación de la cadena de desintegración se pogramará el modelo descrito en el enunciado y se discutirán los resultados obtenidos.

2.2.2. Metodología

El modelo propuesto consiste en lo siguiente:

1. Crear un array con N_0 celdas y dar valores a cada elemento de este, 2 si es un núcleo padre y 1 si es un núcleo hijo, de modo que $N_0 = N_{P,0} + N_{H,0}$.

   Se crea un bucle `while` para hacer las M simulaciones y se indican los estados de los núcleos del array.

```
while (k < M) {
    for (i = 0; i < N_P0; i++) {
        N[i] = 2;
    }
    for (i = 0; i < N_H0; i++) {
        N[i + N_P0] = 1;
    }
    while (contadorH < N_0) {

    ...

    }
    k++;
    pasos = 0;
    j = 0;
    contadorP = 0;
    contadorH = 0;
}
```

2. En cada paso j de la simulación se selecciona un núcleo aleatoriamente y también se necesitará otro número aleatorio $U \in [0,1)$.
   - Si el núcleo está en estado 2 y $U < P_P$, el valor del elemento del array pasará a ser 1 y se avanza un paso la simulación.
   - Si el núcleo está en estado 1 y $U < P_H$, la celda pasará a valer 0 y se avanza un paso en la simulación.

   Donde P_P es la probabidad de que el núcleo padre se desintegre dando un núcleo hijo y P_H la probabilidad de que un núcleo hijo de lugar a un núcleo estable. El paso temporal viene dado por $\mathrm{d}t = [(\lambda_P + \lambda_H)N_0]^{-1}$.

El segundo bucle `while` contiene el modelo de la simulación en sí, una vez que no quedan más núcleos hijo por desintegrar ha finalizado. En este caso distinguimos tres condicionales principales:

- El primero imprime las condiciones iniciales.
- El segundo desintegra los núcleos si se cumplen las condiciones de estado de la celda y probabilidad asociada.
- El tercero sobrescribe los array con los datos de la simulación cada `deltaj` pasos.

```
while (contadorH < N_0) {
    if (pasos == 0) {
        tiempo[j] = pasos*pow((lambda_P + lambda_H)*N_0,-1);
        N_P[j] += N_P0;
        N_H[j] += N_H0;
        j++;
    }
    pasos++;
    random = (unsigned int)rand() % N_0;
    U = (double)rand() / (RAND_MAX + 1.0);
    if (N[random] == 2 && U < P_P) {
        N[random] = 1;
        contadorP++;
    }else{
        if (N[random] == 1 && U < P_H) {
            N[random] = 0;
            contadorH++;
        }
    }
    if (pasos % deltaj == 0) {
        tiempo[j] = pasos*pow((lambda_P + lambda_H)*N_0,-1);
        N_P[j] += (N_P0 - contadorP);
        N_H[j] += (N_H0 + contadorP - contadorH);
        j++;
    }
```

3. La simulación se da por finalizada cuando no queden núcleos activos.

2.2.3. Resultados y discusión

Los parámetros de la simulación se introducen como argumentos de la función `main` en el orden: N_0, $N_{P,0}$, $N_{H,0}$, λ_P, λ_H, Δj y M.

Ejercicio 2.2

Los valores de los parámetros son $N_0 = 1000$, $N_{P,0} = 1000$, $N_{H,0} = 0$, $\lambda_P = 0{,}1$ año^{-1}, $\lambda_H = 0{,}3$ año^{-1}, $\Delta j = 1000$ y $M = 1000$. Por lo que las ecuaciones de las soluciones teóricas son

$$N_P(t) = 1000 \cdot e^{-0{,}1t} \quad \text{y} \quad N_H(t) = 1000 \cdot \frac{0{,}1}{0{,}3 - 0{,}1}\left(e^{-0{,}1t} - e^{-0{,}3t}\right).$$

La evolución temporal del promedio de los núcleos padre e hijo se representa gráficamente en la figura 2.2.

En ambas figuras podemos afirmar que para $M = 1000$ simulaciones el promedio de los núcleos que van quedando concuerdan con los esperados por la solución teórica.

Los núcleos padre se desintegran de forma exponencial partiendo de $N = 1000$ y con un periodo de semidesintegración de entre 7 u 8 años.

Los núcleos hijo experimentan un crecimiento que dura, aproximadamente, hasta que se han desintegrado la mitad de los núcleos padre. Esto se debe a que en los primeros años hay muchos más núcleos padre y su tasa de desintegración es mayor que la de los núcleos hijo, lo que resulta en un balance positivo de núcleos hijo.

Cuando comienza a disminuir el número de núcleos hijo activos no significa que se hayan terminado de desintegrar los núcleos padre, sino que el número de núcleos padre restantes es menor.

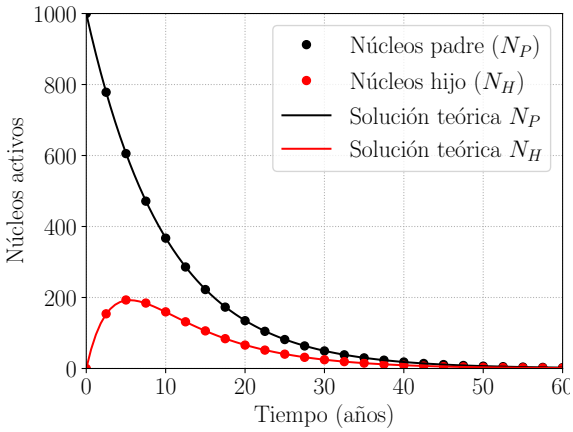

Figura 2.2

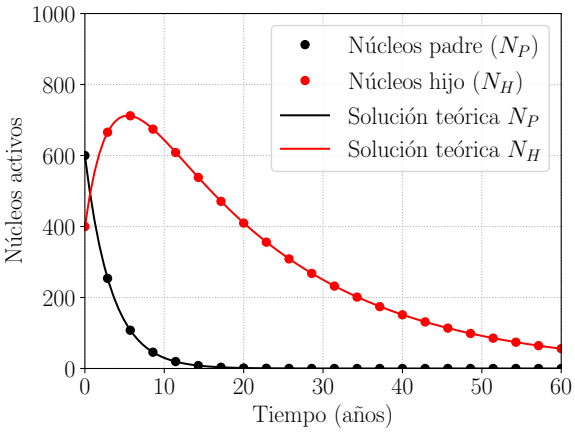

Figura 2.3

Ejercicio 2.3

Los valores de los parámetros son $N_0 = 1000$, $N_{P,0} = 600$, $N_{H,0} = 400$, $\lambda_P = 0{,}3$ año^{-1}, $\lambda_H = 0{,}05$ año^{-1}, $\Delta j = 1000$ y $M = 1000$. Por tanto las ecuaciones de las soluciones teóricas son

$$N_P(t) = 600 \cdot e^{-0{,}3t} \quad \text{y} \quad N_H(t) = 600 \cdot \frac{0{,}3}{0{,}05 - 0{,}3}\left(e^{-0{,}3t} - e^{-0{,}05t}\right) + 400 \cdot e^{-0{,}05t}.$$

La evolución temporal del promedio de los núcleos padre e hijo se representa gráficamente en la figura 2.3.

En los primeros años ocurre los mismo que en la figura 2.2, el ritmo al decaen los núcleos padre en núcleos hijo es superior al de desintegración de los núcleos hijo, resultando en que el balance sea positivo.

Al contar en este caso con un número inicial de núcleos hijo y un periodo de semidesintegración de los núcleos padre menor, el crecimiento inicial es mucho más pronunciado y continúa hasta que se han desintegrado algo más de la mitad de los núcleos padre.

Otro detalle a mencionar es que la curva del decaimiento de los núcleos hijo es menos pronunciada que la de los núcleos padre, es decir, los núcleos tardan más en desintegrarse. Al suponer la mayoría de los núcleos del array no estables, los núcleos hijos deberían ser seleccionados y desintegrarse con mayor frecuencia, sin embargo la probabilidad de que esto ocurra es de un 14,3 %, lo que explica este aplanamiento.

3. Esquema de desintegración ramificada

3.1. Ejercicio 3.1

Demostrar que en el límite $N_0 \to \infty$ la evolución temporal del modelo viene dada por

$$(\mathbf{a}) \quad \frac{\mathrm{d}N_P}{\mathrm{d}t} = -(\lambda_A + \lambda_B)N_P, \quad (\mathbf{b}) \quad \frac{\mathrm{d}N_A}{\mathrm{d}t} = \lambda_A N_P \quad \text{y} \quad (\mathbf{c}) \quad \frac{\mathrm{d}N_B}{\mathrm{d}t} = \lambda_B N_P.$$

Datos: $\mathrm{d}t = [(\lambda_A + \lambda_B)N_0]^{-1}$, $P_A = \lambda_A/(\lambda_A + \lambda_B)$ y $P_B = \lambda_B/(\lambda_A + \lambda_B)$.

(**a**) Los núcleos padre se desintegran en todos los casos, sin seguir ninguna probabilidad, de modo que el promedio de los núcleos padre desintegrados entre un paso y el siguiente viene dado por

$$N_P(j) - N_P(j-1) = -\frac{N_P(j-1)}{N_0}.$$

Sustituyendo el paso temporal $\mathrm{d}t = [(\lambda_A + \lambda_B)N_0]^{-1}$ y la equivalencia entre pasos y tiempo llegamos a

$$\frac{N_P(t_j + \mathrm{d}t) - N_P(t_j)}{\mathrm{d}t} = -N_P(t_j)(\lambda_A + \lambda_B),$$

y cuando $N_0 \to \infty$ obtenemos la expresión deseada:

$$\frac{\mathrm{d}N_P}{\mathrm{d}t} = -(\lambda_A + \lambda_B)N_P.$$

(**b**) Con cada desintegración de los núcleos padre se van añadiendo más núcleos A o B, dependiendo de una probabilidad P_A (o P_B). El número de núcleos A ganados entre pasos consecutivos es

$$N_A(j) - N_A(j-1) = \frac{N_P(j-1)}{N_0} \cdot P_A.$$

Al sustituir el paso temporal $\mathrm{d}t = [(\lambda_A + \lambda_B)N_0]^{-1}$ y simplificar con la probabilidad se obtiene que

$$\frac{N_A(t_j + \mathrm{d}t) - N_A(t_j)}{\mathrm{d}t} = \lambda_A N_P(t_j),$$

tomando el límite $N_0 \to \infty$ se llega a la expresión a demostrar

$$\frac{\mathrm{d}N_A}{\mathrm{d}t} = \lambda_A N_P.$$

(**c**) El procedimiento para obtener la ecuación de la evolución temporal de los núcleos B es análogo al de los núcleos A. Los núcleos B ganados entre pasos consecutivos obedecen

$$N_B(j) - N_B(j-1) = \frac{N_P(j-1)}{N_0} \cdot P_B.$$

Sustituyendo el paso temporal, simplificando con la probabilidad y tomando el límite $N_0 \to \infty$:

$$\frac{N_B(t_j + \mathrm{d}t) - N_B(t_j)}{\mathrm{d}t} = \lambda_B N_P(t_j) \implies \frac{\mathrm{d}N_B}{\mathrm{d}t} = \lambda_B N_P.$$

3.2. Ejercicio 3.2 - 3.3

3.2.1. Introducción

Una desintegración ramificada es aquella en la que para una misma especie radiactiva existen diferentes rutas de desintegración, es decir, dos núclidos idénticos pueden dar lugar a núcleos estables distintos.

En el tipo de desintegración ramificada que se propone para la simulación, un núcleo P se desintegrará en un núcleo A o B dependiendo de una probabilidad asociada P_A (o P_B) donde $P_A + P_B = 1$.

La evolución temporal de estos núclidos se describe con las siguientes ecuaciones:

$$\frac{\mathrm{d}N_P}{\mathrm{d}t} = -(\lambda_A + \lambda_B)N_P, \quad \frac{\mathrm{d}N_A}{\mathrm{d}t} = \lambda_A N_P, \quad \text{y} \quad \frac{\mathrm{d}N_B}{\mathrm{d}t} = \lambda_B N_P.$$

En caso de que $N_{A,0}$ y $N_{B,0}$ no sean nulos las soluciones de las ecuaciones diferenciales son

$$\begin{aligned}
(1)\quad & \frac{\mathrm{d}N_P}{\mathrm{d}t} = -(\lambda_A + \lambda_B)N_P \Rightarrow \frac{\mathrm{d}N_P}{N_P} = -(\lambda_A + \lambda_B)\,\mathrm{d}t \Rightarrow \ln N_P = -(\lambda_A + \lambda_B)t + C_1 \\
& N_P = e^{-(\lambda_A+\lambda_B)t+C_1} \implies N_P(t) = N_{P,0}\, e^{-(\lambda_A+\lambda_B)t}, \\
(2)\quad & \frac{\mathrm{d}N_A}{\mathrm{d}t} = \lambda_A N_P \Rightarrow N_A = \lambda_A \int N_P\,\mathrm{d}t \Rightarrow N_A = \lambda_A \int N_{P,0}\, e^{-(\lambda_A+\lambda_B)t}\,\mathrm{d}t \\
& N_A = N_{P,0}\,\frac{\lambda_A}{\lambda_A + \lambda_B} - e^{-(\lambda_A+\lambda_B)t} + C_2 \\
& \implies N_A(t) = N_{P,0}\,\frac{\lambda_A}{\lambda_A + \lambda_B}\left(1 - e^{-(\lambda_A+\lambda_B)t}\right) + N_{A,0}, \\
(3)\quad & \frac{\mathrm{d}N_B}{\mathrm{d}t} = \lambda_B N_P \Rightarrow N_B = \lambda_B \int N_P\,\mathrm{d}t \Rightarrow N_B = \lambda_B \int N_{P,0}\, e^{-(\lambda_A+\lambda_B)t}\,\mathrm{d}t \\
& N_B = N_{P,0}\,\frac{\lambda_B}{\lambda_A + \lambda_B} - e^{-(\lambda_A+\lambda_B)t} + C_3 \\
& \implies N_B(t) = N_{P,0}\,\frac{\lambda_B}{\lambda_A + \lambda_B}\left(1 - e^{-(\lambda_A+\lambda_B)t}\right) + N_{B,0},
\end{aligned}$$

donde $N_{P,0} = e^{C_1}$, $C_2 = N_{A,0} + N_{P,0} \cdot \lambda_A/(\lambda_A + \lambda_B)$ y $C_3 = N_{B,0} + N_{P,0} \cdot \lambda_B/(\lambda_A + \lambda_B)$.

3.2.2. Metodología

El modelo propuesto consiste en lo siguiente:

1. En un array de N_0 elementos se dan valores a las celdas en función de las condiciones iniciales: 2 si es un núcleo P, 1 si es un núcleo A y 0 si es un núcleo B.

   El bucle `while` permite hacer las M simulaciones, y los `for` llenan las celdas con el estado del núcleo correspondiente.

```
while (k < M) {
    for (i = 0; i < N_P0; i++) {
        N[i] = 2;
    }
    for (i = 0; i < N_A0; i++) {
        N[i + N_P0] = 1;
    }
```

```
    for (i = 0; i < N_B0; i++) {
        N[i + N_P0 + N_A0] = 0;
    }
    while (contadorP < N_P0) {

    ...

    }
    k++;
    pasos = 0;
    j = 0;
    contadorP = 0;
    contadorA = 0;
    contadorB = 0;
}
```

2. En cada paso j de la simulación se selecciona aleatoriamente uno de los núcleos.
   - Si $U < P_A$ la celda pasará de 2 a 1 y avanzaremos un paso en la simulación.
   - Si $U \geq P_A$ la celda pasará de 2 a 0 y se avanza un paso en la simulación.

   Nótese que P_A es la probabilidad de que el núcleo P se desintegre en un núcleo A, y U es un número aleatorio en el intervalo [0,1). El incremento del paso temporal es $\mathrm{d}t = [(\lambda_A + \lambda_B)N_0]^{-1}$.

   Los tres condicionales principales tienen las siguientes funciones: imprime los valores iniciales, desintegra los núcleos en función del estado y la probabilidad asociada, e imprime los datos de la simulación cada `deltaj` pasos.

```
    while (contadorP < N_P0) {
        if (pasos == 0) {
            tiempo[j] = pasos*pow((lambda_A + lambda_B)*N_0,-1);
            N_P[j] += N_P0;
            N_A[j] += N_A0;
            N_B[j] += N_B0;
            aux[j] += 1;
            j++;
        }
        pasos++;
        random = (unsigned int)rand() % N_0;
        U = (double)rand() / (RAND_MAX + 1.0);
        if (N[random] == 2) {
            if (U < P_A) {
                N[random] = 1;
                contadorA++;
            }else {
                N[random] = 0;
                contadorB++;
            }
        }
        contadorP = contadorA + contadorB;
        if (pasos % deltaj == 0) {
            tiempo[j] = pasos*pow((lambda_A + lambda_B)*N_0,-1);
            N_P[j] += N_P0 - contadorP;
            N_A[j] += contadorA + N_A0;
            N_B[j] += contadorB + N_B0;
            aux[j] += 1;
```

```
                j++;
            }
        }
```

3. La simulación termina cuando no queden núcleos P activos.

3.2.3. Resultados y discusión

- **Ejercicio 3.2** los parámetros se introducen como argumentos de la función `main` en el orden: N_0, λ_A, λ_B, Δj y M.

  Los valores asociados son $N_0 = 1000$, $\lambda_A = 0{,}1$ años^{-1}, $\lambda_B = 0{,}3$ año^{-1}, $\Delta j = 500$, $M = 1000$. De modo que las soluciones teóricas vendrán dadas por

$$N_P(t) = 1000 \cdot e^{-(0{,}1+0{,}3)t}, \quad N_A(t) = 1000\frac{0{,}1}{0{,}1+0{,}3}\left(1 - e^{-(0{,}1+0{,}3)t}\right),$$

  y

$$N_B(t) = 1000\frac{0{,}3}{0{,}1+0{,}3}\left(1 - e^{-(0{,}1+0{,}3)t}\right).$$

- **Ejercicio 3.3** Los parámetros se introducen en el orden N_0, $N_{P,0}$, $N_{A,0}$, $N_{B,0}$, λ_A, λ_B, Δj y M.

  Los valores asociados son $N_0 = 1000$, $N_{P,0} = 500$, $N_{A,0} = 300$, $N_{B,0} = 200$, $\lambda_A = 0{,}1$ años^{-1}, $\lambda_B = 0{,}3$ año^{-1}, $\Delta j = 500$, $M = 1000$. Y las soluciones teóricas vienen dadas por

$$N_P(t) = 500 \cdot e^{-(0{,}1+0{,}3)t}, \quad N_A(t) = 500\frac{0{,}1}{0{,}1+0{,}3}\left(1 - e^{-(0{,}1+0{,}3)t}\right) + 300,$$

  y

$$N_B(t) = 500\frac{0{,}3}{0{,}1+0{,}3}\left(1 - e^{-(0{,}1+0{,}3)t}\right) + 200.$$

Los resultados obtenidos en la simulación se presentan en las figuras 3.2 y 3.3.

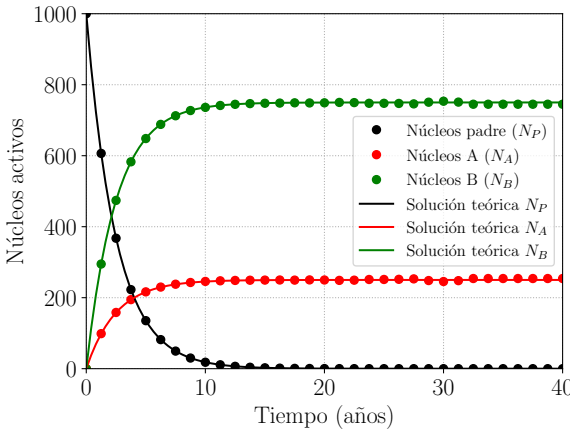

Figura 3.2

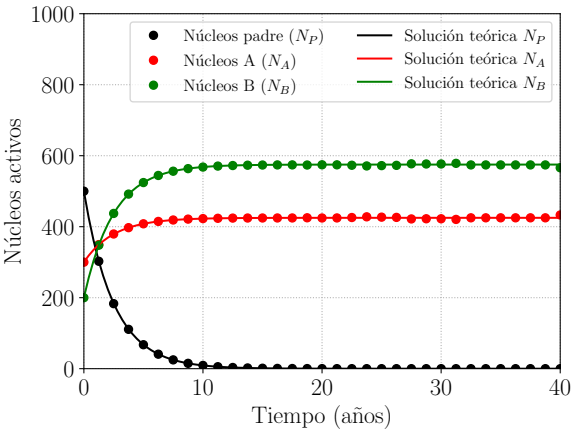

Figura 3.3

En ambos casos los datos producidos por el código se ajustan a los predichos por el modelo teórico.

Cuando $t \to \infty$ tenemos que $N_A = 250$ y $N_B = 750$ para la figura 3.2, y $N_A = 425$ y $N_B = 575$ para la figura 3.3. La diferencia entre los núcleos activos A y B es menor en la segunda figura pese a tener unos parámetros muy similares, aunque se parte de un mayor número de núcleos A el factor determinante en esta diferencia es el número de núcleos P que se desintegran.

También podemos ver que el periodo de semidesintegración T de los núcleos P está en torno a los $2 - 2{,}5$ años.

Cerca de los $30 - 35$ años en la figura 3.2 y los $25 - 30$ años en la figura 3.3 se puede apreciar ruido en los datos. Estas anomalías son propias del código y aparecen al hacer el promedio de las simulaciones. Ocurren cuando para un cierto año (o conjunto de años) se han desintegrado menos núcleos de los que deberían o lo han hecho con valores de núcleos activos restantes que eran menos probables.

Nótese también que cuando se producen más núcleos B se producen menos A y vice versa, esto se debe a las probabilidades asociadas a la desintegración, $P_A + P_B = 1$.

Este ruido se podría mitigar incrementando el número de simulaciones M. Por otro lado, con el número de simulaciones propuesto, los promedios suelen terminar entre los 30 y 35 años, y con menos frecuencia cerca de los $41 - 50$ años, como fue el caso de los datos utilizados para las gráficas.