UNIVERSIDAD NACIONAL ABIERTA
VICERRECTORADO ACADÉMICO
AREA: INGENIERÍA

ASIGNATURA: SIMULACIÓN DE SISTEMAS

CÓDIGO: 337

FECHA DE PUBLICACIÓN EN BLOG DEL SUBPROGRAMA DISEÑO ACADÉMICO: En las Primeras semanas del Lapso 2022-2.

FECHA DE DEVOLUCIÓN POR PARTE DEL ESTUDIANTE:
El estudiante contará hasta el día **22/10/2022 sin prórroga** para su realización y envío.

NOMBRE DEL ESTUDIANTE:Samuel Jesús Rodríguez Figueroa

CÉDULA DE IDENTIDAD: 27.240.851

CORREO ELECTRÓNICO DEL ESTUDIANTE: samuelr76@gmail.com

TELÉFONO: 0414-3966902 (solo whatsApp).

CENTRO LOCAL: (0400) Aragua Sede. CARRERA: 236

LAPSO ACADÉMICO: 2022-2

NUMERO DE ORIGINALES:

FIRMA DEL ESTUDIANTE:
UTILICE ESTA PÁGINA COMO CARÁTULA DE SU TRABAJO

RESULTADOS DE CORRECCIÓN:

OBJ. Nº		4	7
0:NL	1:L		

Índice

M 2, U 4, O 4	3
Pseudocódigo método Runge-Kutta	10
Diagrama de estado método de Runge-Kutta	11
Función Runge-Kutta implementado en Javascript	12
Grafica función solución de la ecuación diferencial	14
Aproximación de la curva obtenida por el metodo de Runge-Kutta para h = 0,1 y h = 0,2	15
Entrega del programa	17
M 3, U 7, O 7	18
Diagrama de estado para la simulación de las reservaciones del Hotel	19
Función simulación para n días.	22
Tabla resumen del servicio al cliente	26
Entrega del programa	27
Conclusión	28

M 2, U 4, O 4

1. Calcular una solución aproximada, es decir, $y_i \sim y(t_i)$ de:

$$\frac{dy}{dx} = -\frac{2}{5}y + \frac{4}{7}e^{-x}$$

$y(0) = 2;$ Para $0 \leq x \leq 1$ usando el método de Runge-kutta de cuarto orden (RK4), para los pasos $h = 0{,}1; h = 0{,}2$.

Reflejar en una tabla los resultados obtenidos del método de RK4 y comparar con la solución exacta de la ecuación.

Solución

El problema trata sobre ecuaciones diferenciales, La ecuación diferencial dada es de primer orden, esto quiere decir que solamente hay un nivel en las derivadas. La ecuación diferencial dada también se puede clasificar como una ecuación diferencial lineal.

De acuerdo al libro **"ecuaciones diferenciales y problemas con valores en la frontera" (capitulo 2** ecuaciones diferenciales de primer orden**)** de **William E. Boyce** y **Richard C.DiPrima**. Existe un método generalizado para resolver ecuaciones diferenciales lineales de primer orden que cumplen la siguiente estructura:

$$y' + ay = g(x)$$

Este método se basa principalmente en multiplicar la ecuación diferencial por:

$$e^{ax}$$

Obteniendo:

$$e^{ax}y' + ae^{ax}y = e^{ax}g(x)$$

Igualando la primera expresión del primer miembro con la segunda parte de la ecuación se tiene:

$$\frac{d}{dx}\left(e^{ax}y\right) = e^{ax}g(x)$$

La solución se tiene al integrar el segundo miembro y despejando "y".

$$y = \phi(x) = e^{-ax}\int^{x} e^{at}g(t)\,dt + ce^{-ax}$$

$\phi(x)$ Representa la solución exacta.

Para resolver la ecuación diferencial se emplea la formula dada por el libro antes mencionados, para que no se vayan a confundir con el desarrollo del ejercicio, se especifican las letras del libro en color azul, y la del enunciado y resolución en color negro.

La ecuación diferencial dada se puede adaptar a la estructura buscada cambiando la notación del diferencial:

$$\frac{dy}{dx} = y'$$

$$y' = -\frac{2}{5}y + \frac{4}{7}e^{-x}$$

Y reordenando las “y” del lado izquierdo de la ecuación, y las “x” del otro lado:

$$y' + \frac{2}{5}y = \frac{4}{7}e^{-x} \quad (1)$$

Al comparar esta ecuación con la estructura objetivo:

$$y' + ay = g(x)$$

Se tiene que:

$$a = \frac{2}{5}$$

Por tanto se multiplica la ecuación diferencial (1) por:

$$e^{\frac{2}{5}x}$$

Quedando:

$$e^{\frac{2}{5}x}y' + \frac{2}{5}ye^{\frac{2}{5}x} = \frac{4}{7}e^{-x}e^{\frac{2}{5}x}$$

Simplificando el lado derecho de la ecuación:

$$e^{\frac{2}{5}x}y' + \frac{2}{5}ye^{\frac{2}{5}x} = \frac{4}{7}e^{-\frac{3}{5}x}$$

Ahora se debe igualar la primera parte del lado izquierdo de acuerdo a la siguiente fórmula:

$$\frac{d}{dx}\left(e^{ax}y\right) = e^{ax}g(x)$$

Quedando:

$$\frac{d}{dx}\left(e^{\frac{2}{5}x}y\right)=\frac{4}{7}e^{-\frac{3}{5}x}$$

Para obtener la solución final se procede a integrar la parte derecha de la ecuación y despejando “y”.

$$e^{\frac{2}{5}x}y=\frac{4}{7}\int^{x}e^{-\frac{3}{5}t}dt$$

$$e^{\frac{2}{5}x}y=\frac{4}{7}\left(-\frac{5}{3}e^{-\frac{3}{5}x}+c\right)$$

$$e^{\frac{2}{5}x}y=-\frac{20}{21}e^{-\frac{3}{5}x}+\frac{4}{7}c$$

$$y=\frac{-\frac{20}{21}e^{-\frac{3}{5}x}+\frac{4}{7}c}{e^{\frac{2}{5}x}}$$

$$y=-\frac{20}{21}e^{-x}+\frac{4}{7}e^{-\frac{2}{5}x}c$$

Para hallar el valor de la constante “c”, se sustituyen los valores iníciales dados “y(0) = 2” y se despeja “c”:

$$2=-\frac{20}{21}e^{-(0)}+\frac{4}{7}e^{-\frac{2}{5}(0)}c$$

$$2=-\frac{20}{21}+\frac{4}{7}c$$

$$2+\frac{20}{21}=\frac{4}{7}c$$

$$c=\frac{31}{6}$$

Ahora se sustituye el valor de “C” en la solución general, y se obtiene una solución única de acuerdo a los valores iníciales dados:

$$y=-\frac{20}{21}e^{-x}+\frac{4}{7}e^{-\frac{2}{5}x}\left(\frac{31}{6}\right)$$

$$y=-\frac{20}{21}e^{-x}+\frac{62}{21}e^{-\frac{2}{5}x}$$

Esta solución se puede comprobar reemplazando “y” y su derivada (y’) en la ecuación diferencial original:

$$y'+\frac{2}{5}y=\frac{4}{7}e^{-x}\quad(1)$$

Tenemos que la solución encontrada y su derivada son:

$$y=-\frac{20}{21}e^{-x}+\frac{62}{21}e^{-\frac{2}{5}x}$$

$$y'=\frac{20}{21}e^{-x}-\frac{124}{105}e^{-\frac{2}{5}x}$$

Reemplazando en la ecuación original:

$$y'+\frac{2}{5}y=\frac{4}{7}e^{-x}$$

$$\left(\frac{20}{21}e^{-x}-\frac{124}{105}e^{-\frac{2}{5}x}\right)+\frac{2}{5}\left(-\frac{20}{21}e^{-x}+\frac{62}{21}e^{-\frac{2}{5}x}\right)=\frac{4}{7}e^{-x}$$

$$\frac{20}{21}e^{-x}-\frac{124}{105}e^{-\frac{2}{5}x}+\left(-\frac{40}{105}e^{-x}+\frac{124}{105}e^{-\frac{2}{5}x}\right)=\frac{4}{7}e^{-x}$$

$$\frac{20}{21}e^{-x}-\cancel{\frac{124}{105}e^{-\frac{2}{5}x}}-\frac{40}{105}e^{-x}+\cancel{\frac{124}{105}e^{-\frac{2}{5}x}}=\frac{4}{7}e^{-x}$$

$$\frac{20}{21}e^{-x}-\frac{8}{21}e^{-x}=\frac{4}{7}e^{-x}$$

$$\frac{4}{7}e^{-x}=\frac{4}{7}e^{-x}$$

Se comprueba entonces que:

$$y=-\frac{20}{21}e^{-x}+\frac{62}{21}e^{-\frac{2}{5}x}$$

Es una solución de la ecuación diferencial:

$$y'+\frac{2}{5}y=\frac{4}{7}e^{-x}$$

La solución $y=-\frac{20}{21}e^{-x}+\frac{62}{21}e^{-\frac{2}{5}x}$ es la fórmula que será utilizada para hallar los valores exacto en el intervalo del $0\le x\le 1$, estos valores servirán como referencia para comprobar la exactitud de la solución dada por el método de runge kutta de cuarto orden.

Método de RUNGE-KUTTA

El método de Runge-Kutta es un método numérico para aproximar los valores de una función desconocida. Esta función desconocida es la solución de la ecuación diferencial en estudio, muchas ecuaciones diferenciales no tienen ó es muy difícil encontrar una solución, el método de RK4 (runge kutta de cuarto orden) permite entonces realizar aproximación de los valores de esa función desconocida en base a los valores iníciales, allí es donde radica su importancia.

La fórmula clásica de Runge-Kutta es equivalente a una formula de cinco términos de la serie de Taylor:

$$y_{n+1} = y_n + hy'_n + \frac{h^2}{2!} y''_n + \frac{h^3}{3!} y'''_n + \frac{h^4}{4!} y^{iv}_n$$

La formula de Runge-Kutta comprende un promedio pesado de los valores de $f(x, y)$, tomado en puntos diferentes del intervalo $x_n \le x \le x_{n+1}$.

La formula de Runge-Kutta de cuarto orden está dada por:

$$y_{n+1} = y_n + \frac{h}{6}\left(k_{n1} + 2k_{n2} + 2k_{n3} + k_{n4}\right)$$

donde

$$\begin{aligned}
k_{n1} &= f\left(x_n, y_n\right), \\
k_{n2} &= f\left(x_n + \frac{1}{2}h,\ y_n + \frac{1}{2}hk_{n1}\right), \\
k_{n3} &= f\left(x_n + \frac{1}{2}h,\ y_n + \frac{1}{2}hk_{n2}\right), \\
k_{n4} &= f\left(x_n + h,\ y_n + hk_{n3}\right),
\end{aligned}$$

La suma $\left(k_{n1} + 2k_{n2} + 2k_{n3} + k_{n4}\right)/6$ puede interpretarse como una pendiente promedio. K_{n1} es la pendiente en el extremo izquierdo del intervalo; k_{n2} es la pendiente en el punto medio usando la formula de Euler para ir de x_n a $x_n + h/2,$ K_{n3} es una segunda aproximación de la pendiente en el punto medio y, finalmente K_{n4} es la pendiente en $x_n + h$ usando la formula de Euler y la pendiente K_{n3} para ir de x_n a $x_n + h.$

Se puede utilizar esta fórmula para aproximar el valor de la función desconocida en todo el intervalo:

$$0 \le x \le 1$$

Y compararla manualmente en una tabla con los valores exactos de la función encontrada:

$$y = -\frac{20}{21}e^{-x} + \frac{62}{21}e^{-\frac{2}{5}x}$$

Sin embargo este proceso sería muy largo y tedioso realizarlo a mano, por ello la importancia y la necesidad del poder computacional para la ejecución de cálculos repetitivos y largos. Sin embargo se puede proceder a utilizar la formula dada para hallar una aproximación de:

$$y = \phi(x) \quad en \quad x = 0.1$$

Tomando la ecuación diferencial no resoluble original:

$$y' = -\frac{2}{5}y + \frac{4}{7}e^{-x}$$

El punto inicial: y(0) = 2. Y tomando h = 0,1 (altura de la escalera para la aproximación). Se puede proceder entonces a calcular las K_n .

$$k_{n1} = f\left(0,2\right) = -\frac{2}{5}\left(2\right) + \frac{4}{7}e^{-(0)} = -\frac{4}{5} + \frac{4}{7} = -\frac{8}{35}$$

$$k_{n2} = f\left(0 + \frac{1}{20},\ 2 + \frac{1}{20}k_{n1}\right) = f\left(0 + \frac{1}{20},\ 2 - \frac{2}{175}\right) = -\frac{2}{5}\left(\frac{348}{175}\right) + \frac{4}{7}e^{-\left(\frac{1}{20}\right)} = -0,2518689$$

$$k_{n3} = f\left(0 + \frac{1}{20},\ 2 + \frac{1}{20}k_{n2}\right) = f\left(0 + \frac{1}{20},\ 2 - 0,012593445\right) = -0,25140295$$

$$k_{n4} = f\left(0 + \frac{1}{10},\ 2 + \frac{1}{10}k_{n3}\right) = f\left(0 + \frac{1}{10},\ 2 + \frac{1}{10}k_{n3}\right) = -0,272893928$$

Ahora se sustituye en:

$$y_{n+1} = y_n + \frac{h}{6}\left(k_{n1} + 2k_{n2} + 2k_{n3} + k_{n4}\right)$$

Y se obtiene la solución aproximada:

$$y_1 = 2 + \frac{0,1}{6}\left(\left(-\frac{8}{35}\right) + 2\left(-0,2518689\right) + 2\left(-0,25140295\right) + \left(-0,272893928\right)\right)$$

$$y_1 = 2 + \frac{0,1}{6}\left(\left(-\frac{8}{35}\right) + 2\left(-0,2518689\right) + 2\left(-0,25140295\right) + \left(-0,272893928\right)\right)$$

$$y_1 = 1,974866516$$

Al compararla con la solución exacta dada por la ecuación diferencial:

$$y = \phi\left(x\right) = -\frac{20}{21}e^{-x} + \frac{62}{21}e^{-\frac{2}{5}x}$$

$$\phi\left(0.1\right) = -\frac{20}{21}e^{-(0.1)} + \frac{62}{21}e^{-\frac{2}{5}(0.1)} = 1,974866517$$

Se tiene que los números enteros y los primeros 8 dígitos decimales concuerdan, obteniendo una diferencia apenas del:

$$y_1 = 1,974866516$$

$$\phi\left(0.1\right) = 1,974866517$$

$$\phi\left(0.1\right) - y_1 = 1\times 10^{-9}$$

Se puede concluir entonces que el método de cuarto orden de Runge-Kutta ofrece una aproximación muy exacta del los valores originales que da la función desconocida.

Se puede comprobar lo largo y tedioso que es aplicar el método de Runge-Kutta de forma manual, Es por ello que se procede a desarrollar un programa computacional para calcular de forma rápida todas las aproximaciones dadas por el metodo de Runge-Kutta en el intervalo:

$$0 \le x \le 1$$

Algoritmo de Runge-Kutta

Para poder desarrollar un programa que resuelva un ejercicio matemático, es necesario identificar y especificar todos los pasos exactos para la resolución. Es por ello que a continuación se esboza una serie de componentes que serán necesarios para la programación del algoritmo de Runge-kutta:

1- Deben haber variables globales que indiquen el estado del programa, tales como la condición inicial y(0) = 2. Y la altura de aproximamiento h.
2- Una función principal que calcule todos los K_n y obtenga sucesivamente los valores de y_n buscados.
3- Debe haber finalmente una función que regresa los valores de $f(x_n, y_n)$.

Una vez identificado algunos componentes clave para la resolución computacional, se puede proceder a diseñar el algoritmo por medio del lenguaje universal de programación, el pseudocódigo. El pseudocódigo permite especificar el algoritmo que resolverá el problema dado, una vez que se cuenta con una explicación paso a paso del algoritmo a programar, es relativamente fácil adaptarlo a cualquier lenguaje de programación.

El pseudocódigo para la aplicación del método de Runge-Kutta a la ecuación diferencial dada se especifica a continuación:

****

Inicio del programa

//Establecer variables globales para la conducción del programa;
//variables X_0 = 0, y = 2, x = 1, h = 0,1 ó h = 0,2;

// Llamar a la función principal Runge-kutta con las variables globales de control
“Runge-Kutta(X_0,y,x,h)”;

//Funcion Runge-Kutta (Parámetros X_0,y,x,h) {
//
// Contar números de interacciones usando el ancho ó la altura de h hasta llegar a x;
// Numero de pasos n = (x- X_0) / h;
//
// Establecer las variables locales K_{n1}, K_{n2}, K_{n3}, K_{n4}.
// Establecer el valor de y inicial a y = y_0;

```
// Inicializar registro que contendrá todos los valores aproximados calculados Aprox[ ]
//
//Iniciar bucle de n interacciones {
// Aplicar formula de Runge-Kutta
//
//K n1 = Llamar función para calcular dx/dy(Parámetros X0 , y);
//K n2 = Llamar función para calcular dx/dy(Parámetros X0 + (1/2)h, y+(1/2)hKn1);
//K n3 = Llamar función para calcular dx/dy(Parámetros X0 + (1/2)h, y+(1/2)hKn2);
//K n4 = Llamar función para calcular dx/dy(Parámetros X0 + h, y+hKn3);
//
// Actualizar el valor de y
//
// y = y+ (h/6) * (K n1+ 2K n2+ 2K n3+ K n4)
//
// Cada valor de y se va registrando en la variable del registro Aprox[ ] =y;
// Actualizar el valor de X0 = X0 + h;
// } Fin del bucle
//
//Regresar ó devolver el registro que contiene todos los valores aproximados Aprox[ ];
//
//} Fin del programa principal de runge Kutta.
//
// función que contiene la formula de la ecuación diferencial del problema;
// función dydx( Parmetros X,Y){
// Regresar ((-2/5)y+(4/7)*exp(-X));
//} Fin de la función que contiene la formula.
*****************************************************************************************************
```

Fin del programa

El pseudocódigo ayuda a entender el algoritmo del programa, pero también ayuda implementar un diagrama de flujo de estado, pues este clarifica la secuencia en como el programa ejecuta cada paso, para resolver el problema final.

A continuación se especifica un diagrama de flujo de estado para el algoritmo especificado en el pseudocódigo:

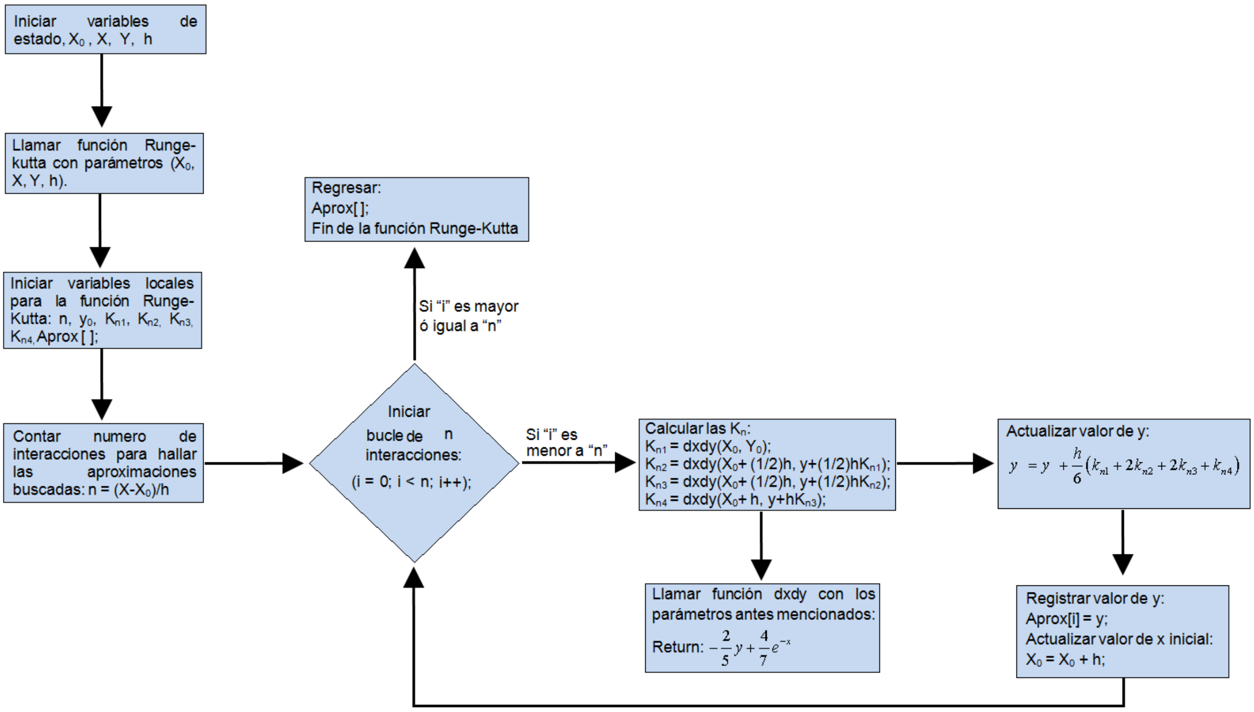


Diagrama de estado "Método de Runge-Kutta"

Ahora que se cuenta con un algoritmo bastante definido, se puede proceder a programar dicho algoritmo mediante algún lenguaje de programación. Para este caso se escogió el lenguaje "Javascript", Javascript tiene la característica de ser el lenguaje nativo de los programas de navegación web, como google, mozilla, brave y edge. En tal sentido, no hace falta descargar e instalar complicadas librerías como sería el caso de pascal y c/c++. Si se cuenta con una aplicación de navegación web, se puede probar el código suministrado en la consola del mismo.

A continuación se especifica la codificación del algoritmo en lenguaje javascript:

```
// A sample differential equation "dy/dx = ((-2/5)*y)+((4/7)*Math.exp(-x))"
function dydx(x, y)
{
    return(((-2/5)*y)+((4/7)*Math.exp(-x)));
}
```

Función que calcule las f(X_n, Y_n).

El objeto Math.exp(-x) realiza la función matemática: e^{-x}

```
1
2   // Finds value of y for a given x using step size h
3   // and initial value y0 at x0.
4   function rungeKutta(x0, y0, x, h)
5   {
6
7       // Count number of iterations using
8       // step size or step height h
9       let n = parseInt((x - x0) / h, 10);
10
11      let k1, k2, k3, k4, k5;
12      let aprox = [];
13
14      // Iterate for number of iterations
15      let y = y0;
16      for(let i = 1; i <= n; i++)
17      {
18
19          // Apply Runge Kutta Formulas to find
20          // next value of y
21          k1 = h * dydx(x0, y);
22          k2 = h * dydx(x0 + 0.5 * h, y + 0.5 * k1);
23          k3 = h * dydx(x0 + 0.5 * h, y + 0.5 * k2);
24          k4 = h * dydx(x0 + h, y + k3);
25
26          // Update next value of y
27          y = y + (1 / 6) * (k1 + 2 * k2 +
28                              2 * k3 + k4);;
29
30          // Update next value of x
31          x0 = x0 + h;
32          aprox.push(y.toFixed(12));
33      }
34      return aprox;
35  }
```

Funcion Runge-Kutta

La función ParseInt en la línea 9 **es una función de alto nivel y no está asociada a ningún objeto**. La función parseInt comprueba el primer argumento, una cadena, e intenta devolver un entero de la base especificada. Esta función se utiliza para asegurar que el número de interacciones “n” sea un entero.

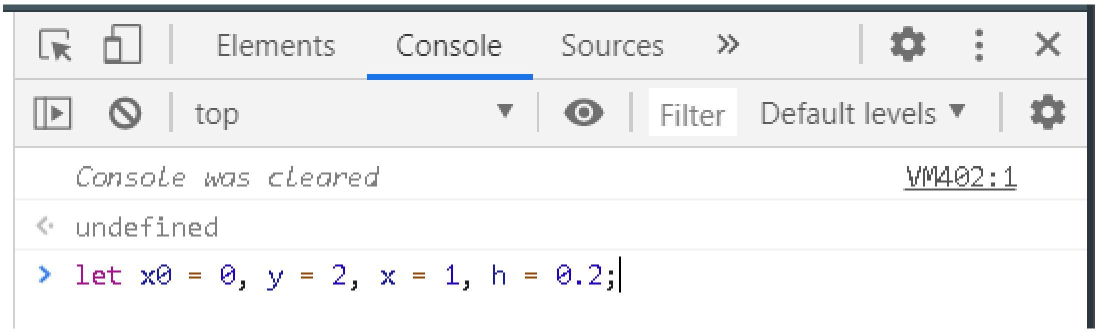
Ya codificado el algoritmo se procede a comprobar si funciona, para ello se busca calcular una aproximación de todos los números de la función objetivo en el intervalo:

$$0 \leq x \leq 1$$

Y comprobar que efectivamente sean números muy próximos a los valores dados por la solución exacta:

$$\phi(x)$$

Se comienza especificando los valores iníciales y el intervalo en donde calculará los valores aproximados de y:



Ahora se llama a la función Runge-Kutta con esas variables:

```
> rungeKutta(x0, y, x, h);
```

Y se obtienen las aproximaciones buscadas:

```
> rungeKutta(x0, y, x, h);
  f(0)=1.94564748796331.                              main.js:31
  f(0.2)=1.8774529588826547.                          main.js:31
  f(0.4)=1.799747260470279.                           main.js:31
  f(0.6000000000000001)=1.7159361308248013.           main.js:31
  f(0.8)=1.6286786606834252.                          main.js:31
```

Estas aproximaciones a primera vista parecen correctas. Para confirmar que sean correctas se procede a realizar una tabla comparativa entre los valores exactos y las aproximaciones dadas por el metodo de Runge-kutta para h = 0,1 y h = 0.2. A continuación se realiza esta tabla, recordando que todos los valores aproximados se consiguen con el programa antes especificado.

x	Runge-Kutta con h = 0.1	Runge-Kutta con h = 0.2	$\phi(x)$ Valor exacto
0	2	2	2
0,1	1,9748665156725558		1,9748665174630882
0,2	1,9456475403429825	1,94564748796331	1,9456475435434184
0,3	1,9129858368074484		1,9129858410871148
0,4	1,877453042317438	1,8774529588826547	1,8774530473901105
0,5	1,8395568275516256		1,8395568331705814
0,6	1,7997473590105046	1,799747260470279	1,7997473649640847
0,7	1,7584231317222194		1,7584231378298472
0,8	1,7159362327592196	1,7159361308248013	1,7159362388678292
0,9	1,6725970902854457		1,672597096266283
1	1,6286787576247548	1,6286786606834252	1,628678763370514

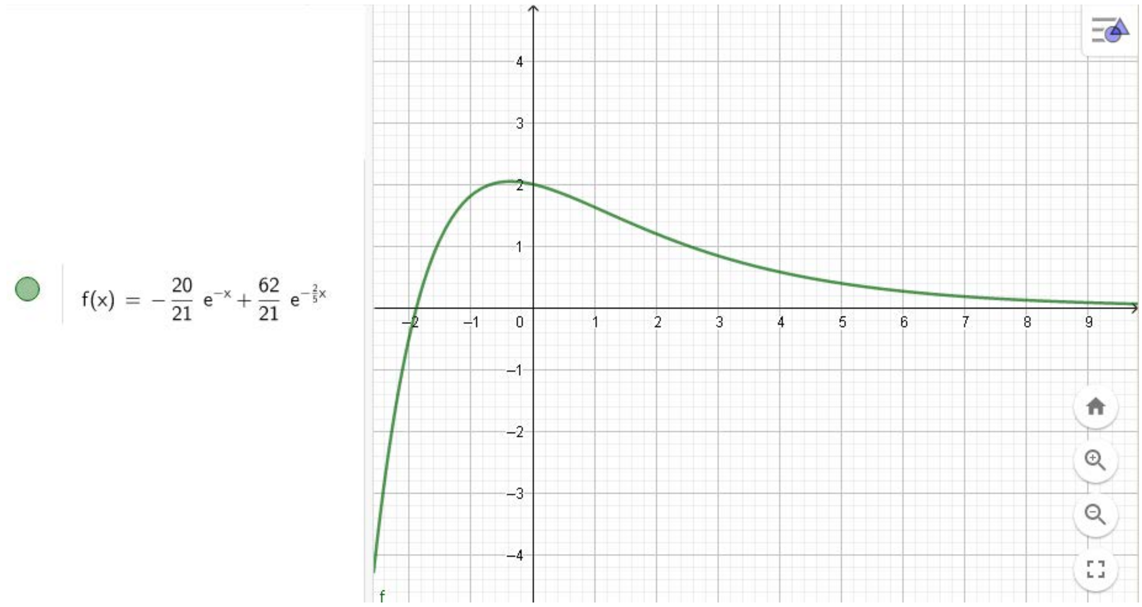
La aproximación por el método de Runge-Kutta siempre es más exacto mientras mas pequeños sea “h”, por ello es que se obtiene una mejor aproximación en nuestro ejemplo con h = 0,1 que con h =0,2.

Se comprueba entonces que las aproximaciones por el método de Runge-Kutta son muy cercanas, prácticamente idénticas en comparación con los valores exactos.

Se puede asimilar la exactitud de los valores calculados con el método de Runge-kutta graficando los puntos de los valores aproximados y comparándolos con la curva exacta dada por la ecuación solución:

$$y = -\frac{20}{21}e^{-x} + \frac{62}{21}e^{-\frac{2}{5}x}$$

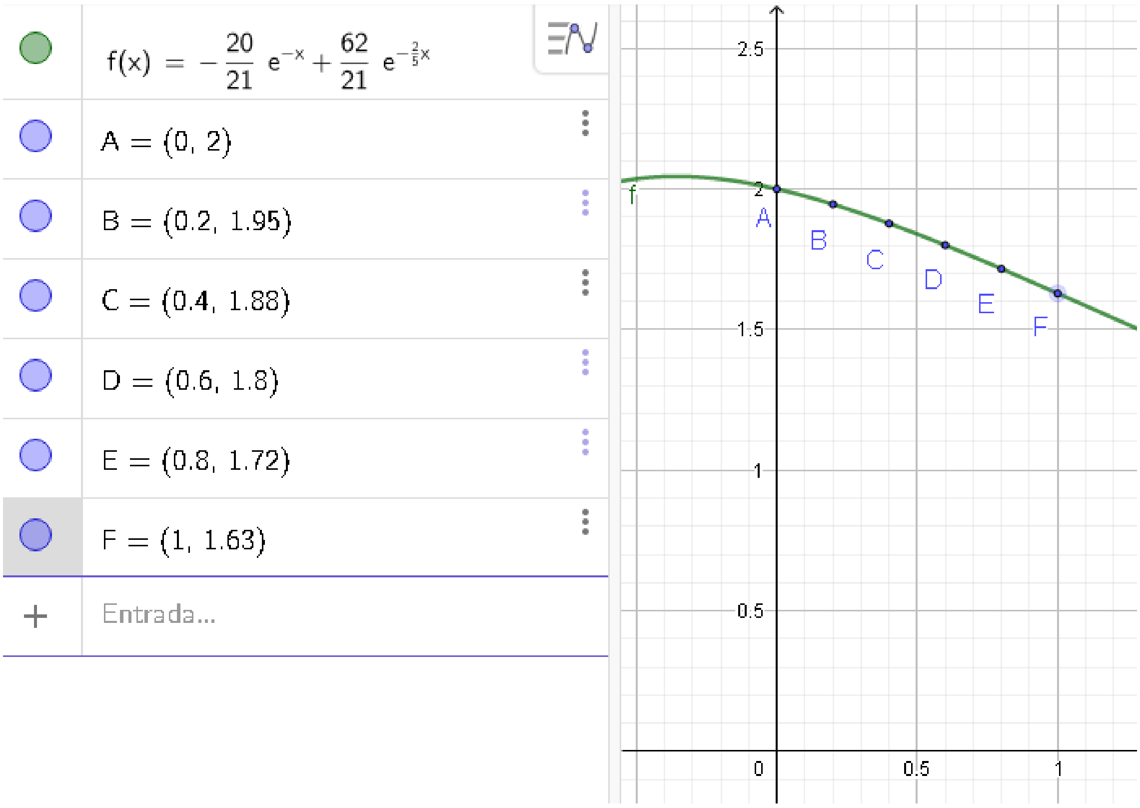
Primero se procede graficando la curva exacta:



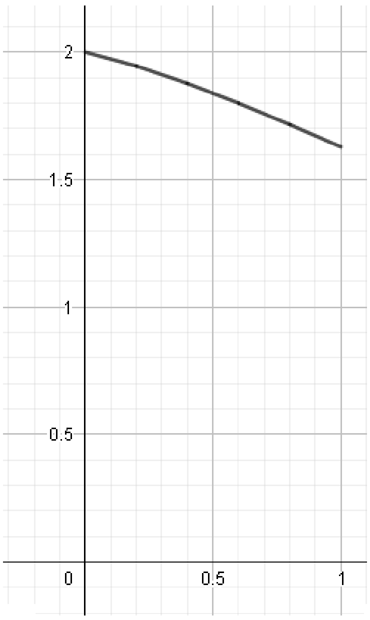
Grafica función solución de la ecuación diferencial.

De la grafica se observa la curva decreciente en todo el dominio x > 0. Eso explica porque los valores aproximados van decreciendo desde 2 hasta 1,62.

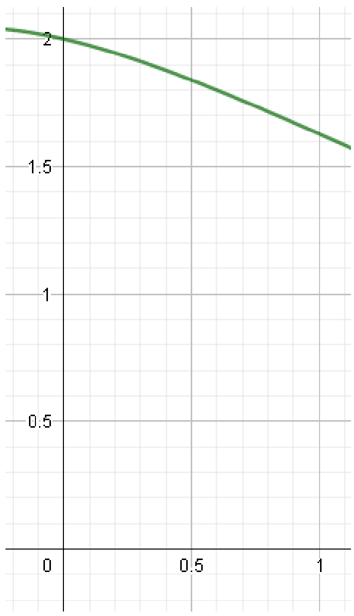
Los valores aproximados por el método de Runge-Kutta con h = 0,2, se grafican por medio de puntos de color azules:



Al unir todos los puntos azules se puede observar la aproximación con que se puede graficar una curva desconocida por el método de Runge-Kutta:

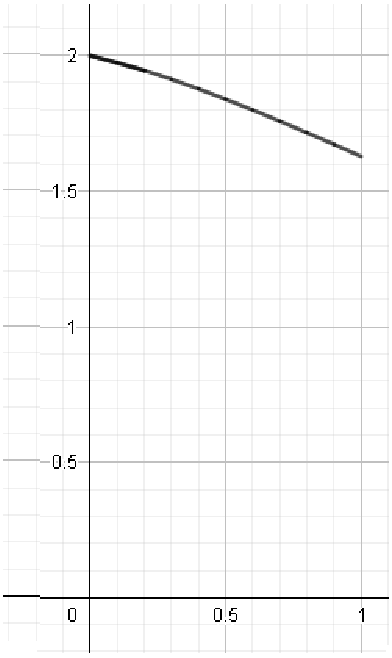


Curva aproximada h = 0,2.

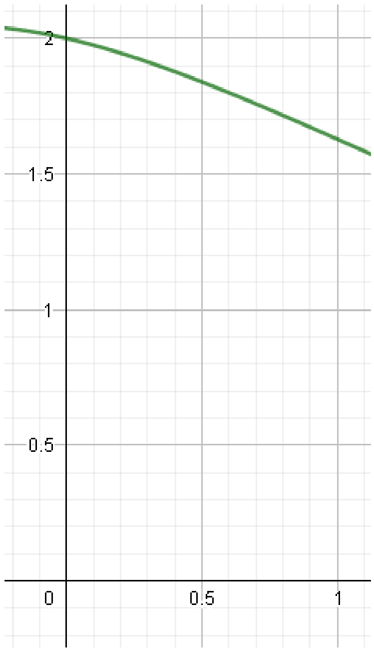


Curva Exacta

Cuando se aplica el método de Runge-Kutta con h más pequeño se puede obtener una mejor aproximación a la curva real, tal como se evidencia en la siguiente grafica:



Curva aproximada h = 0,1



Curva Exacta

Queda de esta forma comprobada la exactitud de las aproximaciones realizadas por el método de Runge-Kutta de cuarto orden.

Entrega del programa

Junto al presente trabajo práctico se entregará el código en lenguaje Javascript, del programa que aplica el método de Runge-Kutta.

Como se había comentado, el lenguaje javascript se ejecuta de forma nativa en todos los programas de navegación web, por lo que es ideal como método programable para sitios web. Se facilita entonces el desarrollo de una página web estática que ejecute este programa. Habría que implementar además los lenguajes HTML y CSS, para la estructuración de la información y el diseño grafico respectivamente. Los lenguajes HTML y CSS no son lenguajes de programación, HTML en un lenguaje de maquetado, y CSS es un lenguaje de especificación. Por lo que no es necesario mostrar en este trabajo los códigos de estos archivos, pero de ser necesario, se puede visualizar desde mi perfil de github dichos archivos, por medio del siguiente Link:

https://github.com/samuelpklm/samuelpklm.github.io-RungeKutta

Github es la plataforma que funge como servidor de la página web estática desarrollada. Además github es un repositorio mundial de códigos de programación, el cual personalmente empleo para todos mis proyectos.

La página web tiene su propia documentación sobre el funcionamiento del programa, un vistazo preliminar de la misma sería la siguiente:

Método de Runge-Kutta (RK4)

El método de Runge-Kutta permite hallar aproximaciones de una función desconocida, esta función desconocida es la solución de la ecuación diferencial en estudio.

El programa aquí desarrollado busca encontrar aproximaciones de la ecuación solución de:

$$\frac{dy}{dx} = -\frac{2}{5}y + \frac{4}{7}e^{-x}$$

Con los valores iníciales y(0) = 2. En el intervalo $0 \leq x \leq 1$.

Para ello se desarrolla un programa que aplica el método de Runge-Kutta, Este programa acepta como valor de entrada:

1. h = Altura con la cual se avanza en las aproximaciones. h debe tener un valor comprendido entre 0.1 a 0.5 . En el ejercicio se prueban con los valores h = 0.1 y h = 0.2 .

Una vez dado el parámetro h de entada se puede ejecutar el programa, el cual comparará los valores aproximados con los reales, e indicará la diferencia entre estos

Sobra decir que la pagina esta optimizada para todos los dispositivos, así que siéntase libre de visitar la página ya sea desde un teléfono ó una computadora.

A continuación se especifica el link del sitio web desarrollado para implementar el método de runge-Kutta de cuarto orden:

https://samuelpklm.github.io/samuelpklm.github.io-RungeKutta/

M 3, U 7, O 7

2.- El hotel de un Aeropuerto tiene 100 habitaciones. En una noche dada se reciben hasta 105 reservaciones debido a la posibilidad de que no todos se presenten. Los registros indican que el número de reservaciones diarias se distribuye uniformemente en el intervalo de enteros de 96 a 105. Esto es, cada número entero tiene una probabilidad de ocurrir de 0.1.

Los clientes que no llegan se representan por la siguiente distribución:

No. Que no se presenta	0	1	2	3	4	5
Probabilidad	0.1	0.2	0.25	0.30	0.10	0.05

Elabore un modelo de simulación para determinar el número esperado de habitaciones.

Solución

Este es un problema de incertidumbre pues no se sabe qué cantidad de habitaciones serán alquiladas por noche. Puede ser un número aleatorio comprendido entre 96 y 105. El hotel toma hasta 105 reservaciones por noches porque es recurrente que uno ó varios clientes no se presenten, pero habría que ver si en realidad esta es una buena estrategia, porque que cabe la posibilidad de que clientes que hicieron reserva se encuentren sin habitaciones, ó que el hotel se quede con habitaciones vacías por noche .

Para comprobar la estrategia antes mencionada se procede a simular el sistema planteado para un periodo de muchos días, del cual se podrá extraer medias importantes, como la cantidad de habitaciones sin ocupar ó la cantidad de clientes que se quedaron sin habitaciones después de reservar.

Primero se procede realizando un diagrama de estado, del cual se podrá visualizar los pasos necesarios para la subsecuente simulación en algún lenguaje de programación:

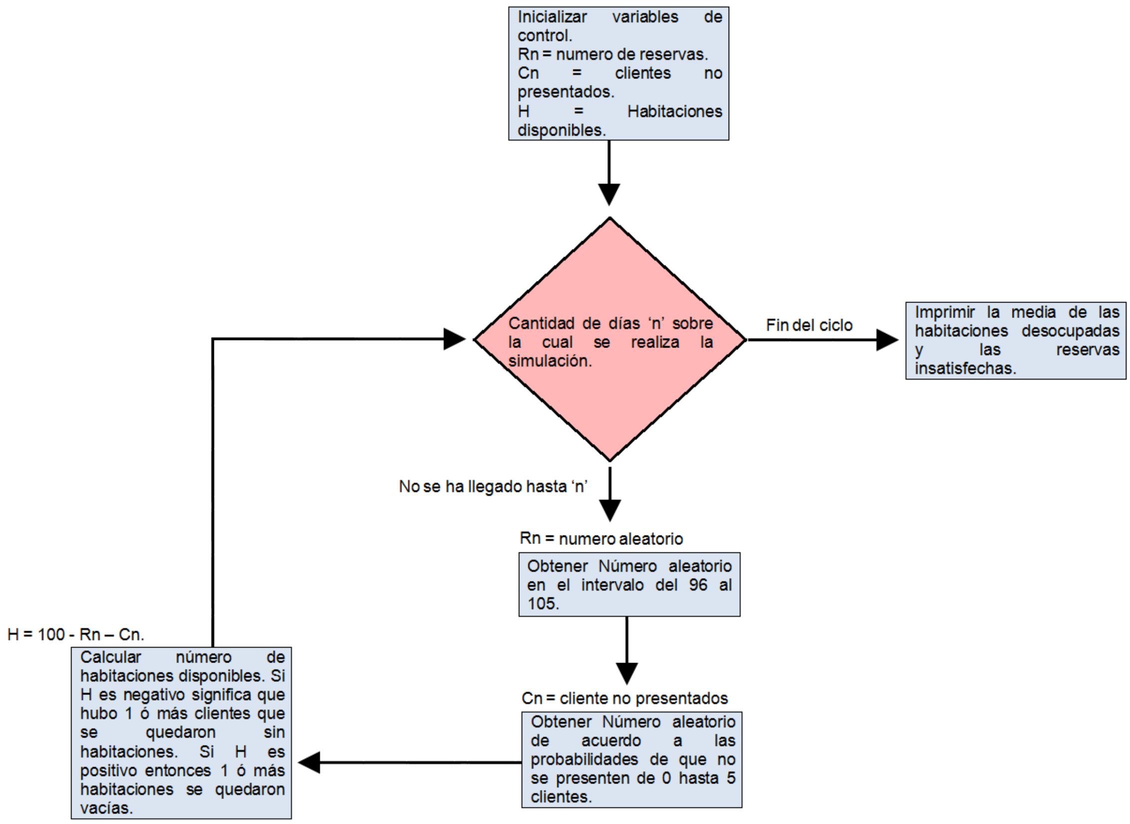


Diagrama de estado para la simulación de las reservaciones del Hotel

Una vez que el algoritmo está lo suficientemente descrito, se puede proceder a programarlo en el lenguaje de preferencias, al igual que el objetivo anterior, se programará en javascript.

El punto esencial sobre el algoritmo implementado, incurre en el tema de los números aleatorios. Para poder realizar una simulación fiable, es preciso que el cálculo de los números aleatorios mantenga la misma distribución especificada en el enunciado del ejercicio.

Para generar números aleatorios de acuerdo a una distribución uniforme en el intervalo del 0 al 1, javascript ofrece el método "Math.random()", se puede obtener el intervalo deseado multiplicando a la función por "tamaño del intervalo", luego al resultado obtenido se le traslada sumando "valor inicial", tal como se especifica a continuación:

```
(Math.random() * 9) + 96)
```

Pero como la función Math.random genera números decimales, luego de multiplicar y sumar el resultado todavía tendrá números decimales, por lo que es preciso usar la función "Math.round()" para redondear el resultado al entero más cercano:

```
Math.round((Math.random() * 9) + 96);
```

La última expresión genera entonces números aleatorios en el intervalo 96≤x≤105.

Para generar números aleatorios de acuerdo de acuerdo a la distribución:

No. Que no se presenta	0	1	2	3	4	5
Probabilidad	0.1	0.2	0.25	0.30	0.10	0.05

Se empleará una tabla de valores del “0 al 100”, que represente la distribución descrita. Es decir:

- Los valores del “0 al 10” indicaran la probabilidad de que no falte ningún cliente.
- Los valores del “10 al 30” indicaran la probabilidad de que solamente falte un cliente.
- Los valores del “30 al 55” indicaran la probabilidad de que solamente falten dos clientes.
- Los valores del “55 al 85” indicaran la probabilidad de que solamente falten tres clientes.
- Los valores del “85 al 95” indicaran la probabilidad de que solamente falten cuatro clientes.
- Los valores del “95 al 100” indicaran la probabilidad de que solamente falten cinco clientes.

Es decir, se generan números aleatorios con la función de distribución uniforme, en el intervalo 0≤x≤100. Y en relación al intervalo en donde caiga se le asignará el valor de clientes que faltaron, a continuación en código:

```
Math.round(Math.random() * 100)
```

Genera un entero aleatorio en el intervalo 0≤x≤100

```
aleatoriofaltas = Math.round(Math.random() * 100);
if(aleatoriofaltas >= 0 && aleatoriofaltas <= 10){
  aleatoriofaltas = 0;
}else if(aleatoriofaltas > 10 && aleatoriofaltas <= 30){
  aleatoriofaltas = 1;
} else if(aleatoriofaltas > 30 && aleatoriofaltas <= 55){
  aleatoriofaltas = 2;
}else if(aleatoriofaltas > 55 && aleatoriofaltas <= 85){
  aleatoriofaltas = 3;
}else if(aleatoriofaltas > 85 && aleatoriofaltas <= 95){
  aleatoriofaltas = 4;
}else if(aleatoriofaltas > 95 && aleatoriofaltas <= 100){
  aleatoriofaltas = 5;
}
```

Asigna el valor de clientes faltantes de acuerdo a la distribución descrita no uniforme.

Las variables de control del programa, ó que reflejan el estado del programa son las siguientes:

```
let aleatorio = 0;
let aleatoriofaltas = 0;

let reservaciones = [];
let inasistencias = [];
let habitaciones = [];
```

Variables de control

La función principal y que simula el caso en cuestión es el siguiente:

```
function simulacion(n){

  for (let i = 0; i < n; i++) {

    aleatorio = Math.round((Math.random() * 9) + 96);
    reservaciones.push(aleatorio);

    aleatoriofaltas = Math.round(Math.random() * 100);
    if(aleatoriofaltas >= 0 && aleatoriofaltas <= 10){
      aleatoriofaltas = 0;
    }else if(aleatoriofaltas > 10 && aleatoriofaltas <= 30){
      aleatoriofaltas = 1;
    } else if(aleatoriofaltas > 30 && aleatoriofaltas <= 55){
      aleatoriofaltas = 2;
    }else if(aleatoriofaltas > 55 && aleatoriofaltas <= 85){
      aleatoriofaltas = 3;
    }else if(aleatoriofaltas > 85 && aleatoriofaltas <= 95){
      aleatoriofaltas = 4;
    }else if(aleatoriofaltas > 95 && aleatoriofaltas <= 100){
      aleatoriofaltas = 5;
    }
     inasistencias.push(aleatoriofaltas);

     habitaciones.push(100 - aleatorio + aleatoriofaltas);

  }

  return "fin";
}
```

Función simulación para n días.

Con este programa se puede proceder a simular el sistema en estudio, basta con especificar el tiempo de la simulación. Como el tamaño de la tabla es directamente proporcional a la cantidad de días, se procederá a escoger un tiempo de simulación corto, de 30 días.

A continuación se especifica el inicio del programa para un periodo de simulación de 30 días:

```
> simulacion(30);
<· "fin"
```

En las variables de control se puede observar los valores obtenidos:

```
> reservaciones
<· (30) [101, 100, 98, 105, 103, 99,
  104, 105, 104, 103, 100, 104, 10
 ▶4, 99, 97, 104, 105, 98, 104, 10
  3, 102, 97, 103, 96, 101, 101, 10
  4, 97, 104, 99]
> inasistencias
<· (30) [4, 2, 3, 2, 2, 3, 1, 4, 2,
 ▶0, 3, 4, 3, 2, 1, 0, 4, 0, 0, 2,
  2, 1, 1, 5, 2, 2, 3, 4, 2, 0]
> habitaciones
<· (30) [3, 2, 5, -3, -1, 4, -3, -1,
  -2, -3, 3, 0, -1, 3, 4, -4, -1,
 ▶2, -4, -1, 0, 4, -2, 9, 1, 1, -1,
  7, -2, 1]
```

De estos valores se puede observar que para la primera noche, se hicieron 101 reservaciones, sin embargo faltaron 4 clientes, por lo que el hotel se quedo con tres habitaciones vacías esa noche.

Para asimilar más fácil los datos obtenidos tras la simulación, se procede a ordenar los valores en una tabla:

Dias	Reservaciones	Inasistencias	Habitaciones
1	101	4	3
2	100	2	2
3	98	3	5
4	105	2	-3
5	103	2	-1
6	99	3	4
7	104	1	-3
8	105	4	-1
9	104	2	-2
10	103	0	-3
11	100	3	3
12	104	4	0
13	104	3	-1
14	99	2	3
15	97	1	4
16	104	0	-4
17	105	4	-1
18	98	0	2
19	104	0	-4
20	103	2	-1
21	102	2	0

22	97	1	4
23	103	1	-2
24	96	5	9
25	101	2	1
26	101	2	1
27	104	3	-1
28	97	4	7
29	104	2	-2
30	99	0	1

Esta tabla permite apreciar la simulación de cada noche en el intervalo de 30 dias, sin embargo cuesta asimilar el promedio ó el porcentaje de veces que hubo clientes desatendidos ó habitaciones vacías. Por lo cual se procede a desarrollar tablas más específicas en las cuales se contraste el porcentaje de reservas, inasistencias y habitaciones servidas.

A continuación se especifica en forma tabular la frecuencia de reservaciones por noche:

Reservaciones	**Cantidad de veces**	**Porcentaje**
96	1	1/30
97	3	3/30
98	2	2/30
99	3	3/30
100	2	2/30
101	3	3/30
102	1	1/30
103	4	4/30
104	8	8/30
105	3	3/30

A continuación se especifica en forma tabular la frecuencia de inasistencias por noche:

Inasistencias	**Cantidad de veces**	**Porcentaje**
0	5	5 / 30
1	4	4 / 30
2	10	10 / 30
3	5	5 / 30
4	5	5 / 30
5	1	1 / 30

Finalmente los datos objetivos de la simulación, el porcentaje de clientes desatendidos ó habitaciones vacías por noche, recuerde que un numero negativo de las habitaciones indica la cantidad de cliente que no se atendieron ó se quedaron sin habitaciones, mientras que los números positivos indica la cantidad de habitaciones vacías por noche. En caso de ser cero, significa que se atendieron a todos los clientes y no quedaron habitaciones vacías, el cual sería el valor buscado por cualquier hotel.

Habitaciones	Cantidad de veces	Frecuencia	Porcentaje
-5	0	0 / 30	0 %
-4	2	2 / 30	6,66 %
-3	3	3 / 30	10 %
-2	3	3 / 30	10%
-1	6	6 / 30	20%
0	2	2 / 30	6,66%
1	3	3 / 30	10%
2	2	2 / 30	6,66 %
3	3	3 / 30	10%
4	3	3 / 30	10%
5	1	1 / 30	3,33 %
6	0	0 / 30	0 %
7	1	1 / 30	3,33 %
8	0	0 / 30	0
9	1	1 / 30	3,33 %

De la tabla anterior se puede concluir que hay una probabilidad por lo menos del 46,6% de que haya 1 ó más clientes desatendidos por noche. Siendo más frecuente que haya un cliente desatendido por noche. Por otro lado hay una probabilidad del 46,74 % de que el hotel se quede con una ó más habitaciones vacías por noche. Habiendo finalmente una probabilidad del 6,66 % de que se reserven todas las habitaciones y que todos los clientes sean atendidos. Todas estas conclusiones en base a la simulación realizada en el periodo de 30 días.

Para obtener una aproximación más fiable de los porcentajes para cada caso, se requiere aumentar el periodo de simulación, ya se comento que esto resultaría en una tabla resumen muy larga, sin embargo, se podría excluir esa tabla, y solamente emplear las tablas resumen de frecuencias, obteniendo de esta forma una salida del programa corta, y con una mejor aproximación de los porcentajes de ocurrencia para cada caso en especifico.

Se procede entonces a simular las reservaciones del hotel para un periodo de 10 años, lo que equivale aproximadamente a 3.650 días:

```
> simulacion(3650);
<· "fin"
```

Los resultados de la simulación se resumen en las siguientes 3 tablas:

Reservaciones	Cantidad de veces	Frecuencia	Porcentaje
96	363	363 / 3650	9,945%
97	360	360 / 3650	9,863%
98	343	343 / 3650	9,397%
99	362	362 / 3650	9,919%
100	382	382 / 3650	10,466%
101	380	380 / 3650	10,411%
102	370	370 / 3650	10,137%
103	346	346 / 3650	9,479%
104	385	385 / 3650	10,547%
105	359	359 / 3650	9,836%

Tablas resumen de las reservaciones

Inasistencias	Cantidad de veces	Frecuencia	Porcentaje
0	386	386 / 3650	10,575%
1	766	766 / 3650	20,986%
2	871	871 / 3650	23,86%
3	1088	1088 / 3650	29,808%
4	353	353 / 3650	9,671%
5	186	186 / 3650	5,095%

Tabla resumen de la frecuencia de inasistencias

Habitaciones	Cantidad de veces	Frecuencia	Porcentaje
-5	34	34 / 3650	0,931%
-4	119	119 / 3650	3,260%
-3	180	180 / 3650	4,931%
-2	313	313 / 3650	8,575%
-1	369	369 / 3650	10,109%
0	349	349 / 3650	9,561%
1	384	384 / 3650	10,520%
2	376	376 / 3650	10,301%
3	394	394 / 3650	10,794%
4	370	370 / 3650	10,136%
5	308	308 / 3650	8,438%
6	229	229 / 3650	6,273%
7	150	150 / 3650	4,109%
8	55	55 / 3650	1,506%
9	20	20 / 3650	0,547%

Tabla resumen del servicio al cliente

La simulación para un periodo de 10 años indica que hay una probabilidad por lo menos del 28% de que haya 1 ó más clientes desatendidos por noche. Siendo más frecuente que haya un cliente desatendido por noche. Por otro lado hay una probabilidad del 62 % de que el hotel se quede con una ó más habitaciones vacías por noche. Habiendo finalmente una probabilidad del 9,561 % de que se reserven todas las habitaciones y que todos los clientes sean atendidos.

Se observa entonces que la estrategia empleada por el hotel es en su mayor medida acertada, pues permite atender a una mayor capacidad de clientes logrando minimizar las habitaciones vacías por noche. Sin embargo, se corre un riesgo del 28% de dejar desatendido a 1 ó más clientes, lo cual es un riego que habría que ponderar para medir su impacto negativo sobre el status del hotel.

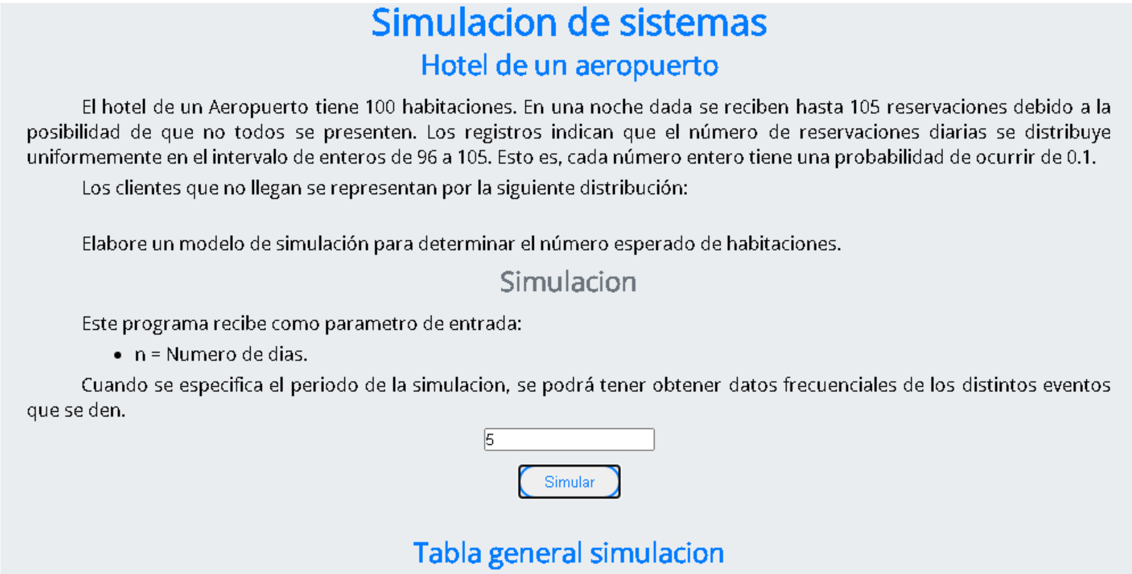
Entrega del programa

Junto al presente trabajo práctico se entregará el código en lenguaje Javascript, del programa que aplica la simulación pertinente.

Al igual que el programa del objetivo 4, este se desarrollará con el lenguaje javascript, y se alojará en una página web estática en los dominios de github. El código HTML, CSS y JS del sitio se puede revisar desde el siguiente link:

https://github.com/samuelpklm/samuelpklm.github.io-Simulacion_de_sistemas_337

La página web tiene su propia documentación sobre el funcionamiento del programa, un vistazo preliminar de la misma sería la siguiente:



Sobra decir que la pagina esta optimizada para todos los dispositivos, así que siéntase libre de visitar la página ya sea desde un teléfono ó una computadora.

A continuación se especifica el link del sitio web desarrollado en donde se ejecuta un programa que simula las reservaciones diarias de un hotel.

https://samuelpklm.github.io/samuelpklm.github.io-Simulacion_de_sistemas_337/

Conclusiones

En matemática existen infinidad de problemas matemáticos que no pueden resolverse por métodos analíticos, ejemplo de ello son las ecuaciones diferenciales. Las ecuaciones diferenciales se podrían dividir en tres tipos de acuerdo a su dificultad de resolución; 1.- Las que comprenden estructuras conocidas, en donde se pueden aplicar métodos conocidos de resolución. 2.- Las que comprenden estructuras conocidas pero no se le puede aplicar ningún método sencillo, sino que es necesario ensayar con diferentes métodos, y so solución es muy complejas. 3.- Las que comprenden estructuras que se sabe no tienen solución.

Para los casos 2 y 3 se hace necesario aplicar entonces métodos numéricos, métodos que aplicando repetidos pasos se puedan encontrar aproximaciones de la solución exacta. En este sentido se han desarrollados métodos prácticos y sencillos de aproximación como el de “Euler”, seguido del metodo “Euler mejorado”, hasta métodos más complicados y de difícil aplicación como el de los tres términos de la serie de Taylor.

No fue si no Carl Runge (1856-1927) en 1895 quien inicio un método practico y verdaderamente exacto para la aproximación de soluciones de ecuaciones diferenciales. Siendo M. W. Kutta (1867-1944) en 1901 el que retomó el estudio de este último, y lo culminó. Definiendo el método de Runge-Kutta tal como lo conocemos hoy en día.

El método de Runge-Kutta se hace largo inclusive para calcular una sola aproximación, pero en sí está compuesto de pasos muy sencillos, en los cuales no se requiere de conocimientos avanzados de matemática. Por lo mismo es que se hace ideal implementar este método en el ordenador, ya que solamente se realizan repetidas multiplicaciones, sumas y a lo sumo se utiliza aproximaciones del número de euler. Programando el algoritmo de Runge-Kutta se pueden hallar infinidad de puntos de la solución original, lo que permite observar la curva de la solución exacta, esto es muy importante para la interpretación física, tal como la identificación de puntos discontinuos ó picos.

En el ámbito empresarial y en los negocios en general existen infinidades de estrategias que pueden servir para hacer más rentable el negocio en sí. Sin embargo, aunque se cuente con estudios razonables, siempre existirá un grado de incertidumbre, por lo que valdría realizar diferentes técnicas de simulación. Luego de aplicar alguna técnica de simulación se obtienen valoraciones probabilistas sobre los distintos eventos que puedan ocurrir. Estas aproximaciones son más fiables que cualquier estudio de mercado, porque se basan en datos reales (Es decir, datos obtenidos en base a eventos ya estudiados).

Para simular un sistema lo suficientemente descrito, es necesario aplicar métodos fiables para la generación de números aleatorios. No se recomienda implementar algoritmos propios para la generación de números aleatorios, porque podrían resultar en la generación de números cíclicos y previsibles, hay que tener en cuenta que cada lenguaje de programación emplea métodos seguros y fiables de generación de números aleatorios, todos ellos generan por defecto números aleatorios en el intervalo $0 \leq x \leq 1$.