MÉTODOS DE SIMULACIÓN Y MODELADO

Solución al Trabajo Práctico - Enero de 2022

EJERCICIO 1

Lea el artículo citado a continuación, que puede descargar de la página web de la asignatura, y conteste a las preguntas.

> Åström, K.J., Elmqvist, H., Mattsson, S.E. *Evolution of continuous-time modeling and simulation*. The 12^{th} European Simulation Multiconference, ESM'98, June 16–19, 1998, Manchester, UK.

1. ¿Qué analogías pueden establecerse entre el modelado basado en diagramas de bloques y el paradigma de la simulación analógica?
2. ¿Qué es el paradigma de modelado físico? ¿Qué tipo de modelos matemáticos se obtienen de aplicar el paradigma del modelado físico?
3. ¿Qué diferencias hay entre el paradigma de la simulación analógica y el paradigma del modelado físico?
4. Observe la Figura 1.1 y comente su contenido basándose en el artículo.

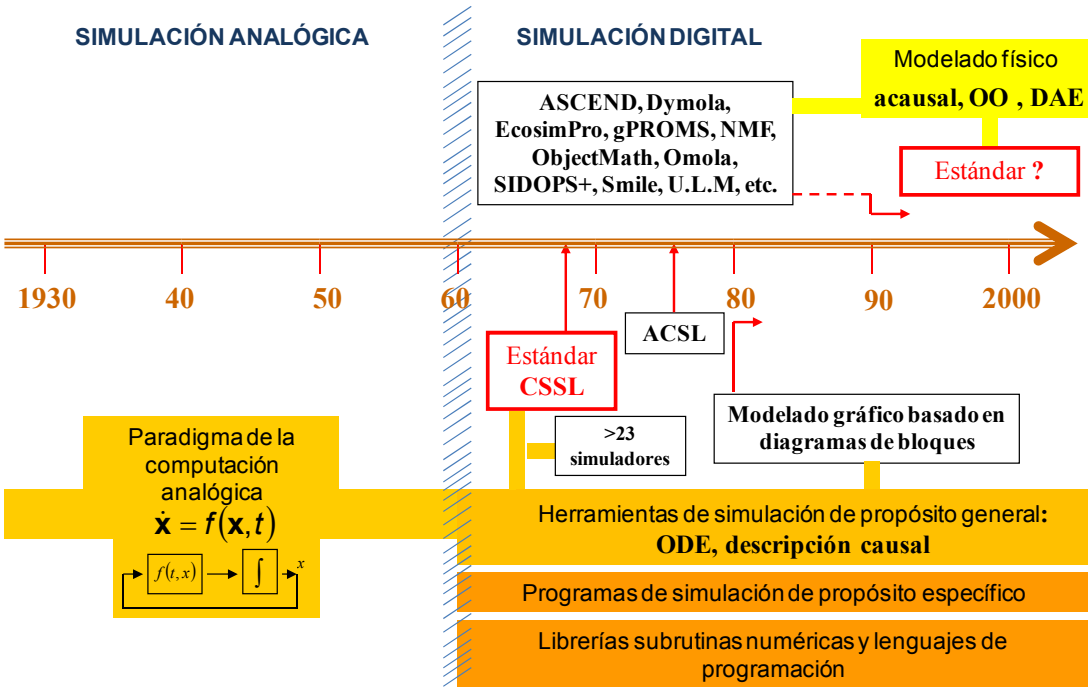


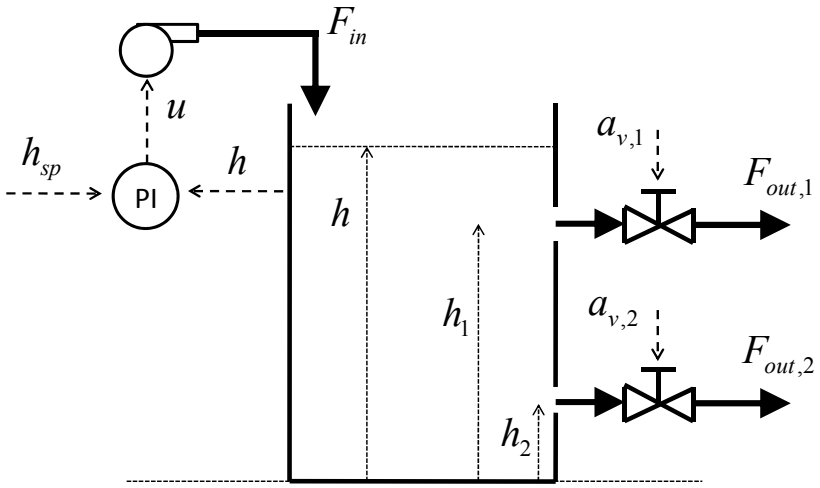
Figura 1.1: Evolución del modelado y simulación de tiempo continuo.

Solución al Ejercicio 1

En este ejercicio el alumno debe contestar, con sus propias palabras, a las cuestiones planteadas.

EJERCICIO 2

Consideremos el sistema mostrado en la figura, que está compuesto por un depósito cilíndrico, una tubería a través de la cual entra un caudal de líquido F_{in} al depósito, y dos tuberías de desagüe con sendas válvulas, a través de las cuales sale del depósito un caudal de líquido $F_{out,1}$ y $F_{out,2}$ respectivamente. El área de la base del depósito, A, es igual a 2 m^2.



Una válvula puede encontrarse en dos fases: abierta ($a_v = 1$) y cerrada ($a_v = 0$). En el primer caso (abierta) permite el paso de líquido y en el segundo (cerrada) lo impide. La fase de la primera válvula cambia cada $T = 30$ segundos, estando inicialmente cerrada. Mientras la primera válvula está abierta, la segunda válvula está cerrada y viceversa.

La primera tubería está situada a una altura $h_1 = 4$ m y la segunda a una altura $h_2 = 1.5$ m. Sólo circula líquido por una tubería si, estando su válvula abierta, la altura del líquido en el depósito (h) es mayor que la altura a la que está situada la tubería. Cuando circula líquido a través de las tuberías, el caudal se calcula de la forma siguiente:

$$\begin{aligned} F_{out,1} &= K \cdot \sqrt{h - h_1} \\ F_{out,2} &= K \cdot \sqrt{h - h_2} \end{aligned}$$

siendo K un parámetro de valor conocido. El valor de K es 0.5, expresado en unidades del sistema internacional.

El caudal de entrada de líquido, F_{in}, es proporcionado por una bomba que es manipulada por un controlador PI. El caudal de la bomba (F_{in}) es proporcional a la tensión de control de la bomba (u) en el rango de valores entre 0 y 20 voltios. La relación constitutiva de la bomba es la siguiente:

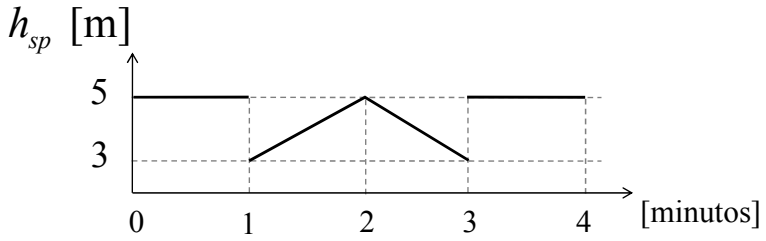
$$F_{in} = B \cdot \text{mín}(20, \text{máx}(0, u))$$

donde B es un parámetro cuyo valor es: $B = 0.3$ m^3/(s·V).

El controlador PI está descrito mediante las ecuaciones siguientes:

$$\begin{aligned} e &= h_{sp} - h \\ \frac{dI}{dt} &= e \\ u &= k_p \cdot e + \frac{1}{k_I} \cdot I \end{aligned}$$

donde los parámetros del controlador valen: $k_p = 5$ V/m, $k_I = 10$ m·s/V. La evolución en el tiempo del valor de consigna para la altura de líquido en el depósito, h_{sp}, es conocida. Se muestra en la figura siguiente h_{sp} frente al tiempo, para el intervalo entre el instante inicial y $t = 240$ s.



La altura inicial de líquido en el depósito es 0.5 m.

1. Escriba las ecuaciones del modelo del sistema. El modelo debe describir la evolución de la altura de líquido (h) y de su valor de consigna (h_{sp}), del caudal de entrada (F_{in}), de los caudales de salida ($F_{out,1}$, $F_{out,2}$), de las variables del controlador (e, u, I), y de las variables que describen las fases de las válvulas ($a_{v,1}$, $a_{v,2}$).

2. Asigne la causalidad computacional. Indique cuántos grados de libertad tiene el modelo.

3. Escriba el diagrama de flujo del algoritmo para la simulación de este modelo. Emplee el método de integración de Euler explícito. La condición de finalización de la simulación es que el tiempo alcance el valor 240 s.

4. Programe el algoritmo anterior en lenguaje R y ejecute la simulación. Represente gráficamente frente al tiempo las variables siguientes: la altura de líquido, su valor de consigna, los flujos de entrada y salida, y las variables que describen las fases de las válvulas. Explique qué criterio ha seguido para escoger el tamaño del paso de integración.

Solución al Ejercicio 2

Las ecuaciones del modelo se muestran a continuación. No se muestran las asignaciones de valor a las constantes y parámetros.

$$
\begin{aligned}
A \cdot \frac{dh}{dt} &= F_{in} - F_{out,1} - F_{out,2} \\
F_{out,1} &= \begin{cases} av_1 \cdot K \cdot \sqrt{h - h_1} & \text{si } h > h_1 \\ 0 & \text{en caso contrario} \end{cases} \\
F_{out,2} &= \begin{cases} av_2 \cdot K \cdot \sqrt{h - h_2} & \text{si } h > h_2 \\ 0 & \text{en caso contrario} \end{cases} \\
F_{in} &= B \cdot \text{mín}(20, \text{máx}(0, u)) \\
e &= h_{sp} - h \\
\frac{dI}{dt} &= e \\
u &= k_p \cdot e + \frac{I}{k_I} \\
av_1 &= f_1(t) \\
av_2 &= 1 - av_1 \\
h_{sp} &= f_2(t)
\end{aligned}
$$

Se ha empleado la notación siguiente: las funciones $f_1(t)$ y $f_2(t)$ representan la dependencia temporal descrita en el enunciado de la apertura de la primera válvula y del valor de consigna para el nivel de líquido.

Asumiendo que las variables que aparecen derivadas pueden ser seleccionadas como variables de estado, las variables del modelo pueden clasificarse de la forma siguiente:

– Parámetros y constantes: A, h_1, h_2, K, B, k_p, k_I
– Variables de estado: h, I
– Variables algebraicas: F_{in}, $F_{out,1}$, $F_{out,2}$, h_{sp}, e, u, av_1, av_2

Para asignar la causalidad computacional al modelo, se sustituyen las derivadas de las variables de estado por variables mudas:

$$\frac{dh}{dt} \rightarrow derh \qquad\qquad \frac{dI}{dt} \rightarrow derI$$

Omitiendo las ecuaciones en las que se asigna valor a los parámetros y las constantes, se obtiene el modelo siguiente:

$$
\begin{aligned}
A \cdot derh &= F_{in} - F_{out,1} - F_{out,2} && (1.1)\\
F_{out,1} &= \begin{cases} av_1 \cdot K \cdot \sqrt{h - h_1} & \text{si } h > h_1 \\ 0 & \text{en caso contrario} \end{cases} && (1.2)\\
F_{out,2} &= \begin{cases} av_2 \cdot K \cdot \sqrt{h - h_2} & \text{si } h > h_2 \\ 0 & \text{en caso contrario} \end{cases} && (1.3)\\
F_{in} &= B \cdot \text{mín}(20, \text{máx}(0, u)) && (1.4)\\
e &= h_{sp} - h && (1.5)\\
derI &= e && (1.6)\\
u &= k_p \cdot e + \frac{I}{k_I} && (1.7)\\
av_1 &= f_1(t) && (1.8)\\
av_2 &= 1 - av_1 && (1.9)\\
h_{sp} &= f_2(t) && (1.10)
\end{aligned}
$$

Las variables del modelo pueden clasificarse en conocidas (el tiempo, los parámetros, las constantes y las variables de estado) y desconocidas (las variables algebraicas y las derivadas de las variables de estado):

– Conocidas: t
A, h_1, h_2, K, B, k_p, k_I
h, I

– Desconocidas: F_{in}, $F_{out,1}$, $F_{out,2}$, h_{sp}, e, u, av_1, av_2
$derh$, $derI$

Con el fin de analizar si el modelo es *estructuralmente singular*, se comprueba que:

1. El número de ecuaciones y de variables desconocidas (incógnitas) es el mismo. Este modelo tiene 10 ecuaciones y 10 incógnitas.

2. Cada incógnita puede emparejarse con una ecuación en que aparezca y con la cual no se haya emparejado ya otra incógnita.

$$
\begin{array}{lcl}
derh & \rightarrow & \text{Ec. (1.1)} \\
F_{out,1} & \rightarrow & \text{Ec. (1.2)} \\
F_{out,2} & \rightarrow & \text{Ec. (1.3)} \\
F_{in} & \rightarrow & \text{Ec. (1.4)} \\
e & \rightarrow & \text{Ec. (1.5)} \\
derI & \rightarrow & \text{Ec. (1.6)} \\
u & \rightarrow & \text{Ec. (1.7)} \\
av_1 & \rightarrow & \text{Ec. (1.8)} \\
av_2 & \rightarrow & \text{Ec. (1.9)} \\
h_{sp} & \rightarrow & \text{Ec. (1.10)}
\end{array}
$$

Aplicando el algoritmo de la partición, se obtiene el siguiente modelo ordenado y resuelto (existen otras posibles ordenaciones de las ecuaciones que son igualmente válidas):

$$
\begin{aligned}
[av_1] &= f_1(t) \\
[h_{sp}] &= f_2(t) \\
[av_2] &= 1 - av_1 \\
[F_{out,1}] &= \begin{cases} av_1 \cdot K \cdot \sqrt{h - h_1} & \text{si } h > h_1 \\ 0 & \text{en caso contrario} \end{cases} \\
[F_{out,2}] &= \begin{cases} av_2 \cdot K \cdot \sqrt{h - h_2} & \text{si } h > h_2 \\ 0 & \text{en caso contrario} \end{cases} \\
[e] &= h_{sp} - h \\
[u] &= k_p \cdot e + \frac{I}{k_I} \\
[F_{in}] &= B \cdot \text{mín}(20, \text{máx}(0, u)) \\
[derh] &= \frac{F_{in} - F_{out,1} - F_{out,2}}{A} \\
[derI] &= e
\end{aligned}
$$

El modelo posee dos variables de estado, por lo que tiene dos grados de libertad. En la Figura 1.2 se muestra el diagrama de flujo para la simulación del modelo, empleando el método de integración de Euler explícito y considerando como condición de finalización que el tiempo simulado alcance el valor 240 s. Se han asignado valores arbitrarios a los valores iniciales de las variables de estado.

Se ha seleccionado un tamaño del paso igual a 0.01 s. Para ello, se ha repetido la simulación empleando diferentes tamaños del paso, encontrándose que el error cometido escogiendo 0.01 s es admisible para el propósito de este estudio.

Se muestra a continuación el código R que implementa el algoritmo mostrado en la Figura 1.2, con un intervalo de comunicación de 0.05 s. En las Figuras 1.3 y 1.4 se muestran las gráficas obtenida al ejecutar el código R.

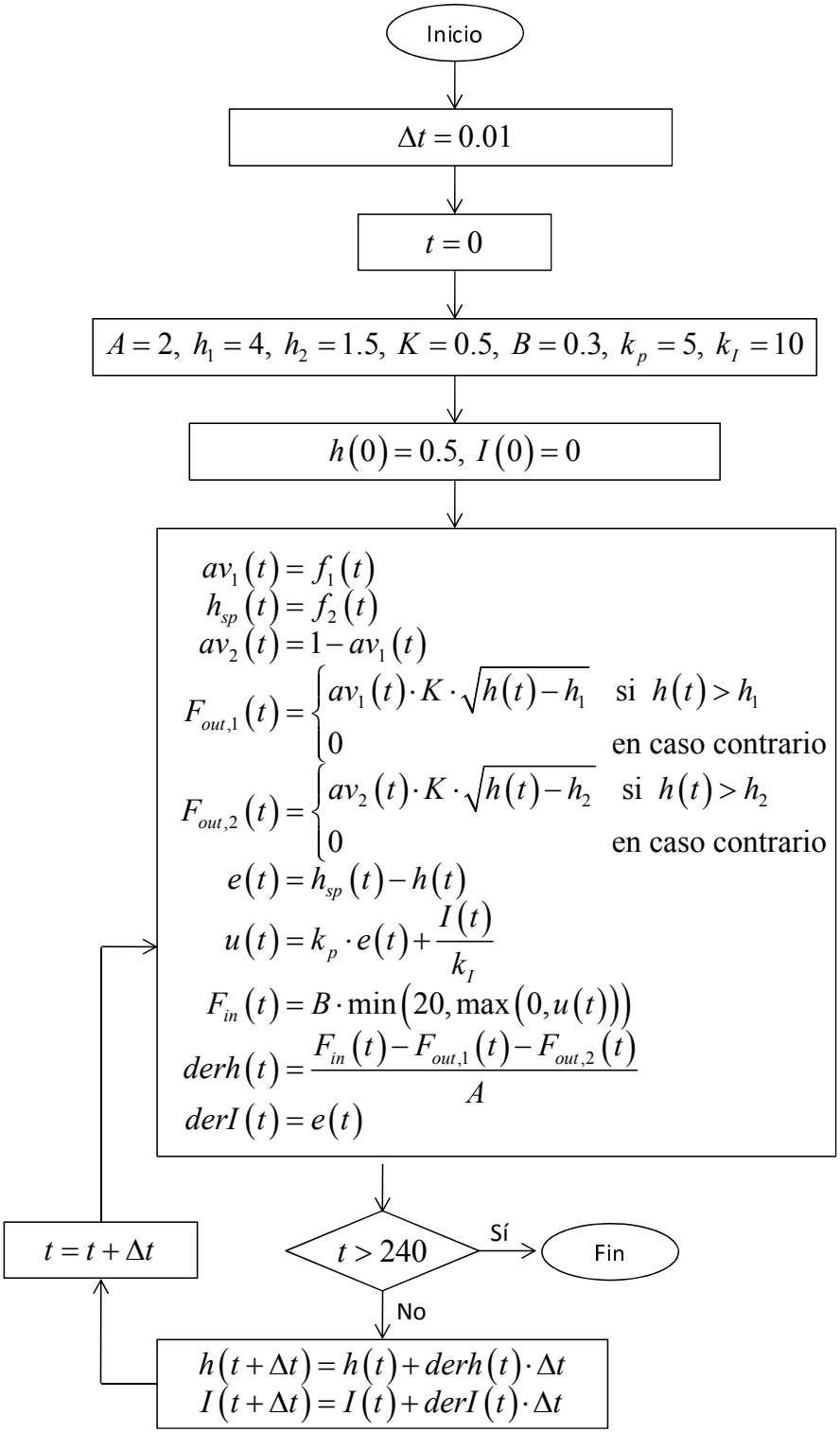


Figura 1.2: Diagrama de flujo de la simulación del modelo.

Código R del algoritmo de la simulación

```
t_fin     <- 240  # Valor final del tiempo
inc_t     <- 0.01 # Tamaño del paso de integración
N_com     <- 5    # Intervalo de comunicación: N_com*inc_t = 0.05 s
Nstep_fin <- round(t_fin / inc_t) # Número de pasos de integración
# Parámetros
A    <- 2
h1   <- 4
h2   <- 1.5
K    <- 0.5
B    <- 0.3
kp   <- 5
kI   <- 10
# Inicialización del tiempo
t    <- 0
# Valor inicial del estado
h    <- 0.5
I    <- 0
# Inicialización resultados salida
t_plot      <- numeric(0)
h_plot      <- numeric(0)
hsp_plot    <- numeric(0)
Fin_plot    <- numeric(0)
Fout1_plot  <- numeric(0)
Fout2_plot  <- numeric(0)
av1_plot    <- numeric(0)
av2_plot    <- numeric(0)
t_ultimaCom <- -Inf
# Bucle de la simulación
termina <- FALSE
n_step  <- 0
while ( !termina ) {
   termina <- n_step == Nstep_fin
   # Cálculo variables algebraicas en t
   av1   <- if (t %% 60 < 30) 0 else 1
   hsp   <- if (t < 60) 5 else if ( t < 120 ) t/30+1 else
            if (t < 180) -t/30+9 else 5
   av2   <- 1 - av1
   Fout1 <- if ( h > h1 ) av1*K*sqrt(h-h1) else 0
   Fout2 <- if ( h > h2 ) av2*K*sqrt(h-h2) else 0
   e     <- hsp - h
   u     <- kp*e + I/kI
   Fin   <- B*min(20,max(0,u))
   derh  <- (Fin - Fout1 - Fout2)/A
   derI  <- e
   # Almacenar resultados de salida
   if ( termina | n_step %% N_com == 0 ) {
      t_plot      <- c(t_plot, t)
      h_plot      <- c(h_plot, h)
      hsp_plot    <- c(hsp_plot, hsp)
      Fin_plot    <- c(Fin_plot, Fin)
      Fout1_plot  <- c(Fout1_plot, Fout1)
      Fout2_plot  <- c(Fout2_plot, Fout2)
      av1_plot    <- c(av1_plot, av1)
      av2_plot    <- c(av2_plot, av2)
   }
   n_step <- n_step + 1
   # Valor en t+inc_t de los estados
   h <- h + inc_t*derh
   I <- I + inc_t*derI
   # Actualización del valor del tiempo
   t <- t + inc_t
}
```

```
# Representación gráfica
# Para poder usar minor.tick es necesario instalar el paquete Hmisc

plot(t_plot, hsp_plot, ylim=c(0,5.2), xlab = "t [s]",
     ylab = "h (rojo), hsp (azul) [m]", type="l", col="blue", lwd=1.5,
     panel.first = abline(v = seq(0,t_fin,10), h = seq(0,5.2,0.2),
                          lwd = 0.5, lty = 3, col="grey") )
points(t_plot, h_plot, type="l", col="red", lwd=1.5)
minor.tick(nx = 5, ny = 5, tick.ratio = 0.5)


dev.new()
plot(t_plot, Fin_plot, ylim=c(0,6), xlab = "t [s]",
     ylab = "Fin [m3/s]", type="l", col="blue", lwd=1.5,
     panel.first = abline(v = seq(0,t_fin,10), h = seq(0,6,0.2),
                          lwd = 0.5, lty = 3, col="grey") )
minor.tick(nx = 5, ny = 5, tick.ratio = 0.5)


dev.new()
plot(t_plot, Fout1_plot, ylim=c(0,1), xlab = "t [s]",
     ylab = "Fout1 (azul), Fout2 (rojo) [m3/s]",
     type="l", col="blue", lwd=1.5,
     panel.first = abline(v = seq(0,t_fin,10), h = seq(0,1,0.02),
                          lwd = 0.5, lty = 3, col="grey") )
points(t_plot, Fout2_plot, type="l", col="red", lwd=1.5)
minor.tick(nx = 5, ny = 5, tick.ratio = 0.5)


dev.new()
par( mfrow = c(2,1) )
plot(t_plot, av1_plot, ylim=c(0,1), xlab = "t [s]",
     ylab = "av1", type="l", col="blue", lwd=1.5,
     panel.first = abline(v = seq(0,t_fin,10), h = c(0,1),
                          lwd = 0.5, lty = 3, col="grey") )
minor.tick(nx = 5, ny = 1, tick.ratio = 0.5)

plot(t_plot, av2_plot, ylim=c(0,1), xlab = "t [s]",
     ylab = "av2", type="l", col="blue", lwd=1.5,
     panel.first = abline(v = seq(0,t_fin,10), h = c(0,1),
                          lwd = 0.5, lty = 3, col="grey") )
minor.tick(nx = 5, ny = 1, tick.ratio = 0.5)
```

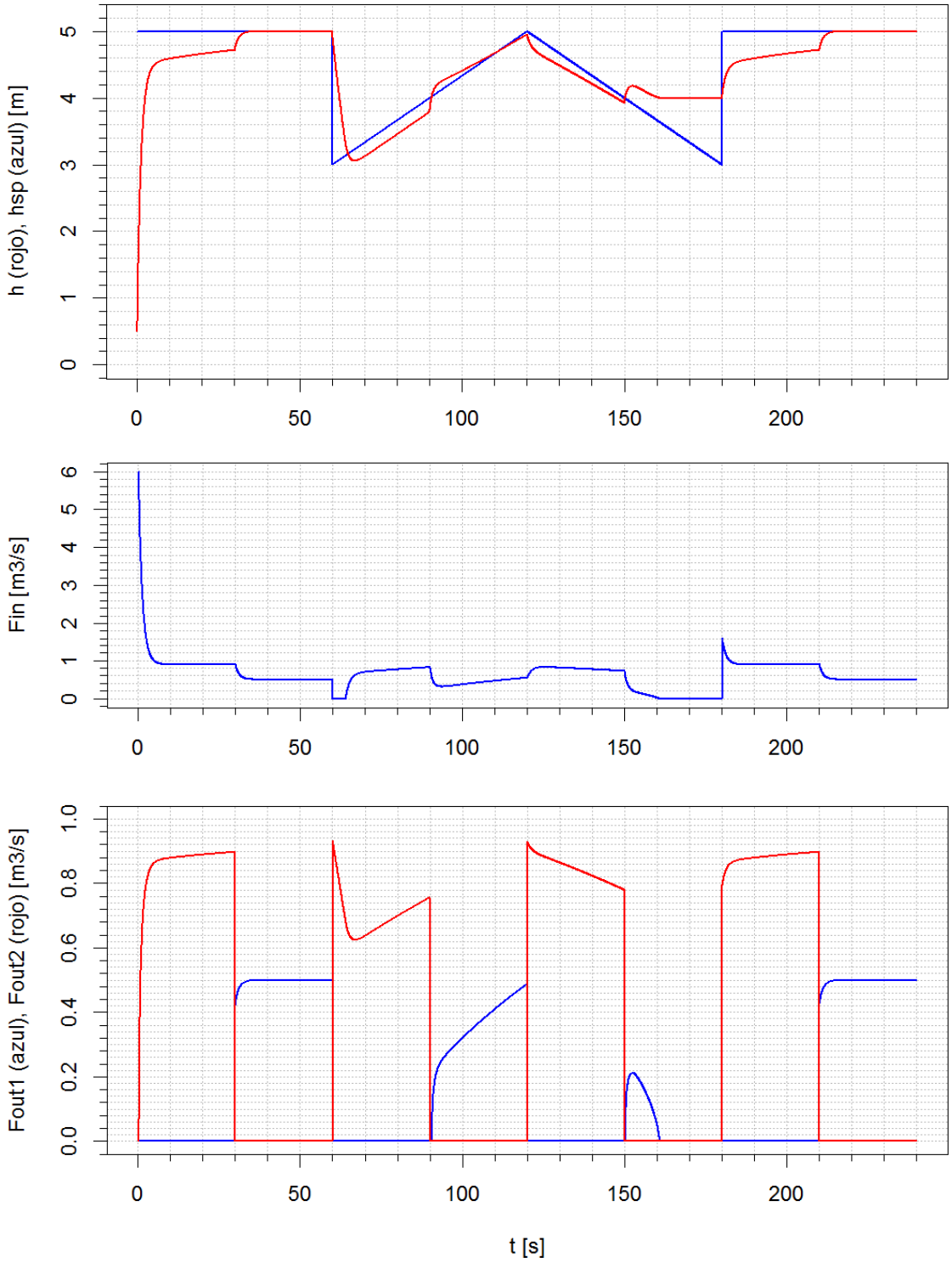


Figura 1.3: Resultado de la ejecución del código R.

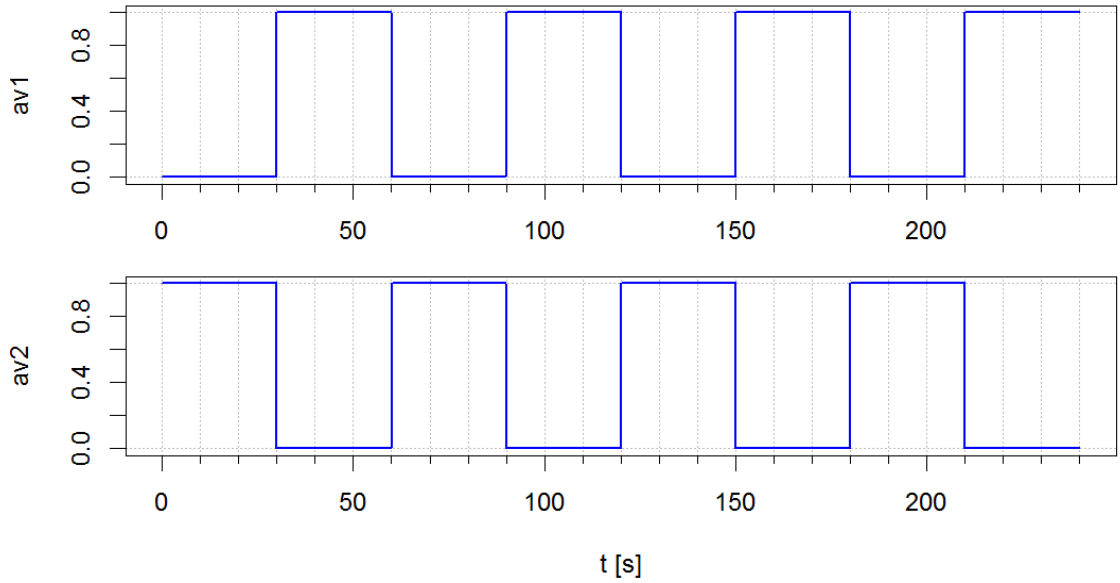


Figura 1.4: Resultado de la ejecución del código R.

EJERCICIO 3

Describa en lenguaje Modelica el sistema del ejercicio anterior, de las dos maneras siguientes:

1. Como un modelo atómico, que viene descrito por las ecuaciones que usted ha planteado al contestar a la pregunta anterior.
2. Programe una librería que contenga los componentes necesarios para componer el sistema descrito en el Ejercicio 2. A continuación, defina dicho sistema como un modelo compuesto, instanciando y conectando componentes de la librería que ha creado.

Asigne a los parámetros los valores indicados en el ejercicio anterior y simule los dos modelos anteriores durante 240 s.

Represente gráficamente frente al tiempo las variables siguientes: la altura de líquido, su valor de consigna, los flujos de entrada y salida, y las variables que describen las fases de las válvulas.

Solución al Ejercicio 3

El modelo atómico en lenguaje Modelica se muestra en Código 1.1. El resultado de la simulación está representado en las Figuras 1.5 y 1.6.

En el Código 1.2 y 1.3 se muestra una posible forma de modelar los componentes, organizándolos en una librería, y de componer el sistema mediante dichos componentes.

```
model deposito
  parameter Real h1(unit="m") = 4;
  parameter Real h2(unit="m") = 1.5;
  parameter Real K = 0.5;
  parameter Real A( unit="m2") = 2;
  parameter Real T( unit="s") = 30;
  parameter Real B(unit="m3/(s.V)") = 0.3;
  parameter Real kp(unit="V/m") = 5;
  parameter Real kI(unit="m.s/V") = 10;
  parameter Real u_max(unit="V")=20;
  parameter Real u_min(unit="V")=0;
  Real Fin(unit="m3/s");
  Real Fout1(unit="m3/s");
  Real Fout2(unit="m3/s");
  Real h(unit="m", start=0.5, fixed=true);
  Real hsp(unit="m");
  Real e(unit="m");
  Real u(unit="V");
  Real u_sat(unit="V");
  Real I(unit="m.s", start=0, fixed=true);
  Boolean av1(start=false, fixed=true), av2;
equation
  // Balance de volumen del líquido
  A*der(h) = Fin - Fout1 - Fout2;
  // Tuberías
  Fout1 = if noEvent(av1 and h>h1) then K*sqrt(h-h1) else 0;
  Fout2 = if noEvent(av2 and h>h2) then K*sqrt(h-h2) else 0;
  // Bomba de líquido
  Fin = B*u_sat;
  u_sat = min(u_max, max(u_min, u));
  // Controlador PI
  e = hsp - h;
  der(I) = e;
  u = kp*e+I/kI;
  // Apertura de las válvulas
  av2 = not av1;
  when sample(T,T) then
    av1 = not pre(av1);
  end when;
  // Consigna para la altura de líquido
  hsp = if time<60 then 5 else if time < 120 then
        time/30+1 else if time < 180 then -time/30+9 else 5;
end deposito;
```

Código 1.1: Ejercicio 3.1: modelo atómico del sistema.

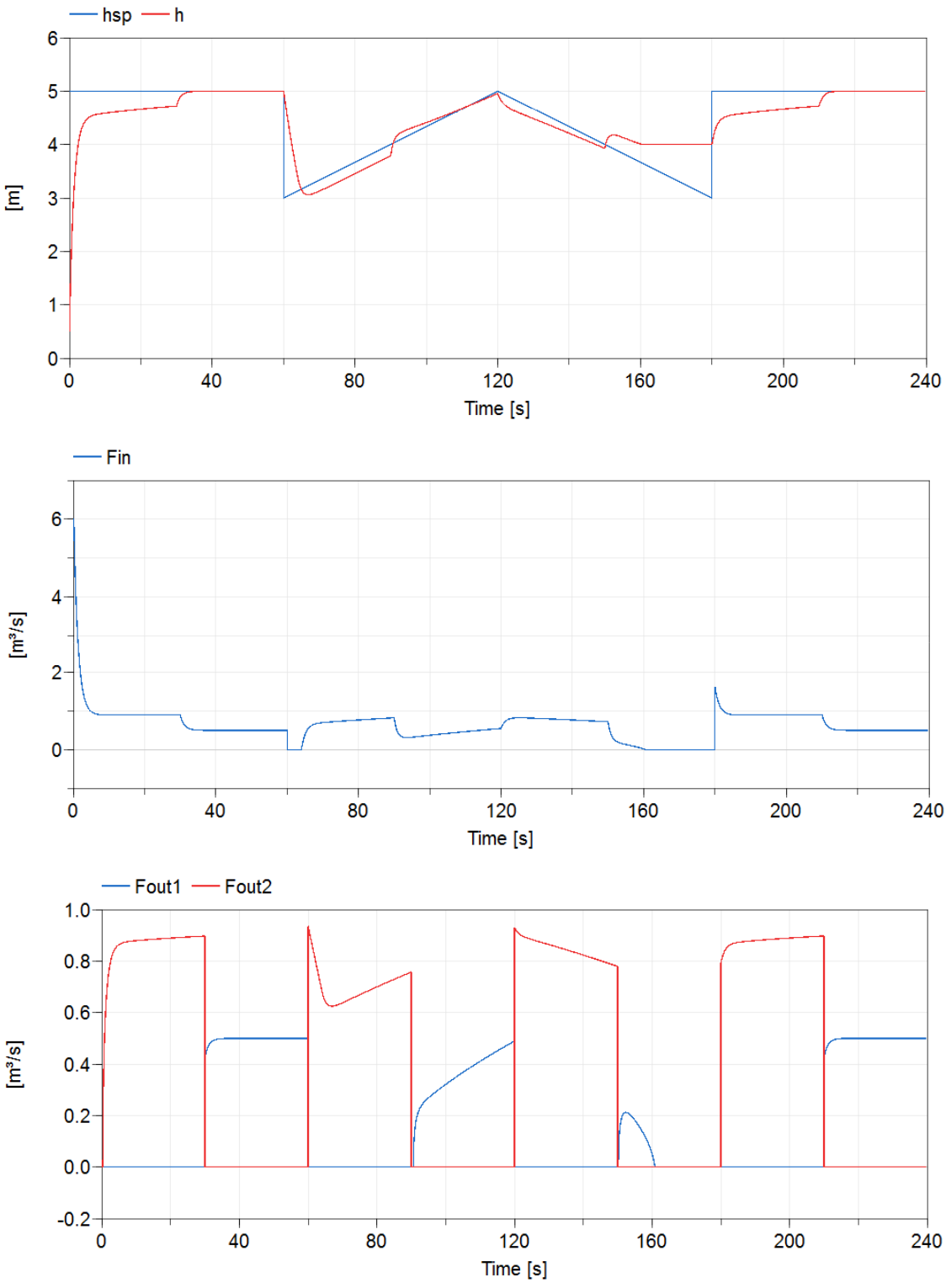


Figura 1.5: Resultado de la simulación usando Dymola.

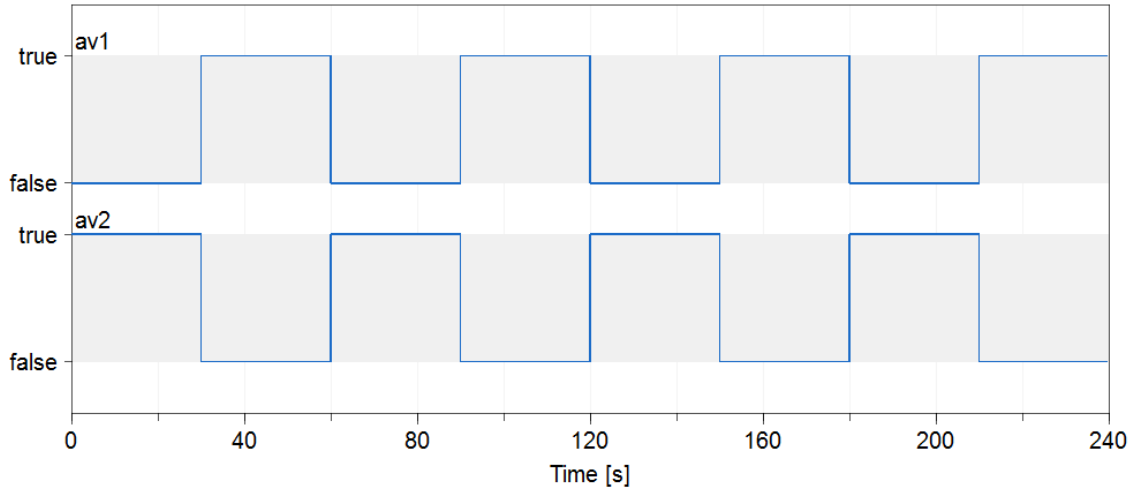


Figura 1.6: Resultado de la simulación usando Dymola.

```
package LibreriaDeposito

package Conector

connector Signal
    Real s;
end Signal;

connector Liquido
    Real h (unit="m" );
    flow Real F (unit="m3/s");
end Liquido;

end Conector;

package Componentes

model BombaLiquido
  Conector.Liquido con_liq;
  Conector.Signal  con_u;
  parameter Real B(unit="m3/(s.V)");
  parameter Real u_max(unit="V") = 20;
  parameter Real u_min(unit="V") = 0;
  Real u_sat(unit="V");
equation
  con_liq.F = - B*u_sat;
  u_sat = min(u_max,max(u_min,con_u.s));
end BombaLiquido;

model ControladorPI
  Conector.Signal con_h;
  Conector.Signal con_u;
  parameter Real kp;
  parameter Real kI;
  input Real hsp;
  Real e;
  Real I(start=0,fixed=true);
equation
  e = hsp - con_h.s;
  der(I) = e;
  con_u.s = kp*e + I/kI;
end ControladorPI;

model ValvulaDesague
  Conector.Liquido con_liq;
  input Boolean av;
  parameter Real K = 0.5;
equation
  con_liq.F = if noEvent(av and con_liq.h > 0)
              then K*sqrt(con_liq.h)
              else 0;
end ValvulaDesague;
```

Código 1.2: Ejercicio 3.2: librería (1/2).

```
model Deposito
  Conector.Liquido con_h1;
  Conector.Liquido con_h2;
  Conector.Liquido con_top;
  Conector.Signal  sensor_h;
  parameter Real h1(unit="m");
  parameter Real h2(unit="m");
  parameter Real A( unit="m2") = 2;
  Real h(unit="m",start=0.5,fixed=true);
equation
  con_h1.h  = h - h1;
  con_h2.h  = h - h2;
  con_top.h = 0;
  A*der(h) = con_top.F + con_h1.F + con_h2.F;
  sensor_h.s = h;
end Deposito;

end Componentes;


model deposito
  parameter Real h1(unit="m") = 4;
  parameter Real h2(unit="m") = 1.5;
  parameter Real K = 0.5;
  parameter Real A( unit="m2") = 2;
  parameter Real T( unit="s") = 30;
  parameter Real B(unit="m3/(s.V)") = 0.3;
  parameter Real kp(unit="V/m") = 5;
  parameter Real kI(unit="m.s/V") = 10;

  Componentes.Deposito deposito(h1=h1,h2=h2,A=A);
  Componentes.ValvulaDesague valvula1(av=av1,K=K);
  Componentes.ValvulaDesague valvula2(av=av2,K=K);
  Componentes.ControladorPI PI(hsp=hsp,kp=kp,kI=kI);
  Componentes.BombaLiquido  bombaLiq(B=B);

  Real hsp(unit="m");
  Boolean av1(start=false,fixed=true),av2;
equation

  connect(deposito.con_top,bombaLiq.con_liq);
  connect(deposito.con_h1,valvula1.con_liq);
  connect(deposito.con_h2,valvula2.con_liq);
  connect(deposito.sensor_h,PI.con_h);
  connect(PI.con_u,bombaLiq.con_u);

  // Apertura de las valvulas
  av2 = not av1;
  when sample(T,T) then
    av1 = not pre(av1);
  end when;
  // Consigna para la altura de liquido
  hsp = if time<60 then 5 else if time < 120 then
          time/30+1 else if time < 180 then -time/30+9 else 5;
end deposito;

end LibreriaDeposito;
```

Código 1.3: Ejercicio 3.2: librería (2/2).

EJERCICIO 4

Escriba un script en FlexPDE para obtener la temperatura y los vectores correspondientes al flujo de calor del sistema descrito a continuación.

El sistema está formado por un bloque de madera de pino. El bloque de madera de pino tiene una conducción térmica anisótropa, siendo su constante de conductividad diferente en la direcciones x e y. La conductividad térmica en la dirección del eje X es $k_x = 0.15$ W/(m·K) y en la dirección del eje Y es $k_y = 0.36$ W/(m·K).

En la superficie superior del bloque de madera hay un calefactor eléctrico que tiene una anchura de 0.02 m. El calefactor produce calor a una tasa constante de 500 W/m^2. La superficie inferior del bloque de madera se mantiene a una temperatura constante de 300 K. El resto de lados del bloque de madera están bien aislados.

El sistema se trata como un problema bidimensional. En la figura se muestra una sección transversal, en la cual el bloque tiene una forma cuadrada de lado 0.2 m. La línea negra más gruesa indica la superficie que está calefactada.

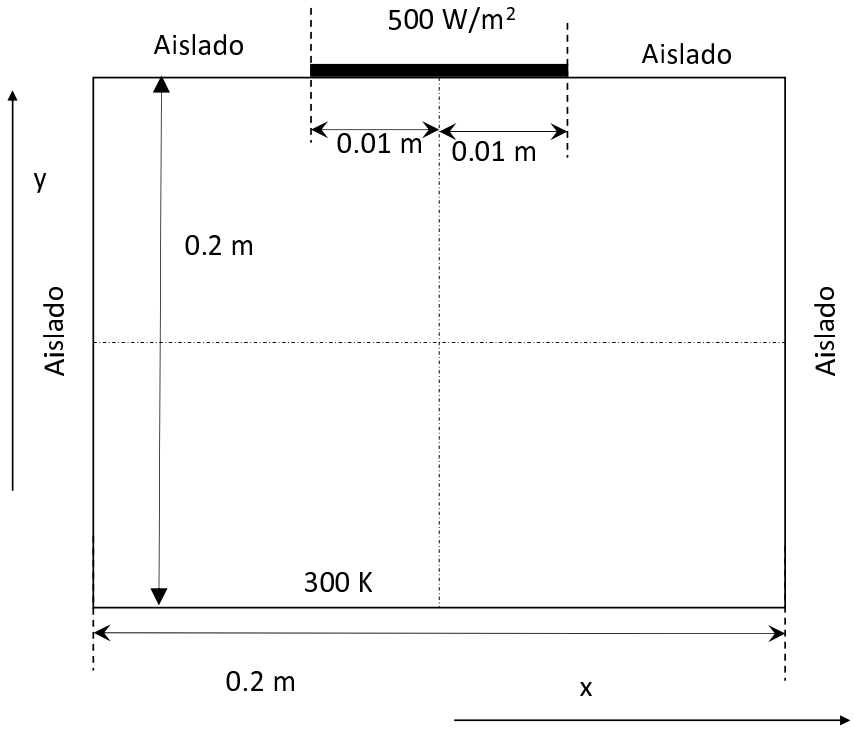


Figura 1.7: Sección del bloque de madera (dibujo no a escala).

Este fenómeno se describe mediante la ecuación diferencial siguiente:

$$\frac{\partial}{\partial x}\left(-k_x \cdot \frac{\partial T}{\partial x}\right) + \frac{\partial}{\partial y}\left(-k_y \cdot \frac{\partial T}{\partial y}\right) = 0$$

donde T es la temperatura.

Escriba el código del *script* de FlexPDE para obtener un gráfico de las líneas de temperatura y un gráfico vectorial del flujo de calor. Las componentes x e y del flujo de calor son, respectivamente, las siguientes:

$$\begin{aligned} q_x &= -k_x \cdot \frac{\partial T}{\partial x} \\ q_y &= -k_y \cdot \frac{\partial T}{\partial y} \end{aligned}$$

Solución al Ejercicio 4

El *script* se muestra en el Código 1.4, el gráfico de la temperatura en la Figura 1.8 y el gráfico vectorial del flujo de calor en la Figura 1.9.

```
TITLE 'Bloque con conductividad anisotropa'
SELECT
  errlim=1e-3 spectral_colors
VARIABLES temp
DEFINITIONS
 L=0.1 w=0.01 kx=0.15 ky=0.36 calor=0
 fluxd_x = -kx*dx(temp) fluxd_y=-ky*dy(temp)
  fluxd=vector(fluxd_x,fluxd_y)
EQUATIONS
  dx(-kx*dx(temp))+dy(-ky*dy(temp))=calor
BOUNDARIES
region 'dominio' {Define el dominio del problema}
  START (-L,-L) VALUE(temp) = 300
      LINE TO (L,-L)    NATURAL(temp) = 0
      LINE TO (L, L) TO (w,L)
      NATURAL(temp) = -500
      LINE TO (-w, L)     NATURAL(temp)= 0  LINE TO
    (-L,L) TO CLOSE
PLOTS
CONTOUR(temp)
VECTOR(fluxd)
END
```

Código 1.4: Distribución de calor en un bloque.

El gráfico con las curvas de nivel de la temperatura y el gráfico vectorial del flujo de calor se muestran en las figuras 1.8 y 1.9.

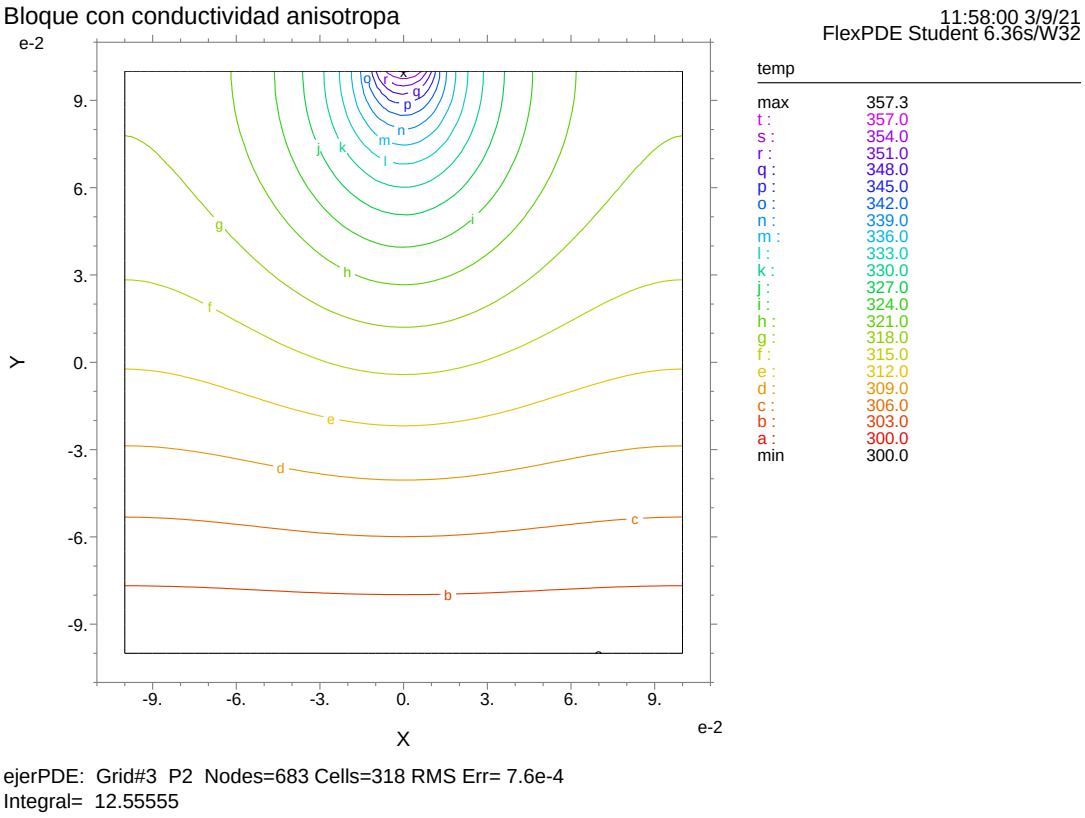


Figura 1.8: Curvas de temperatura.

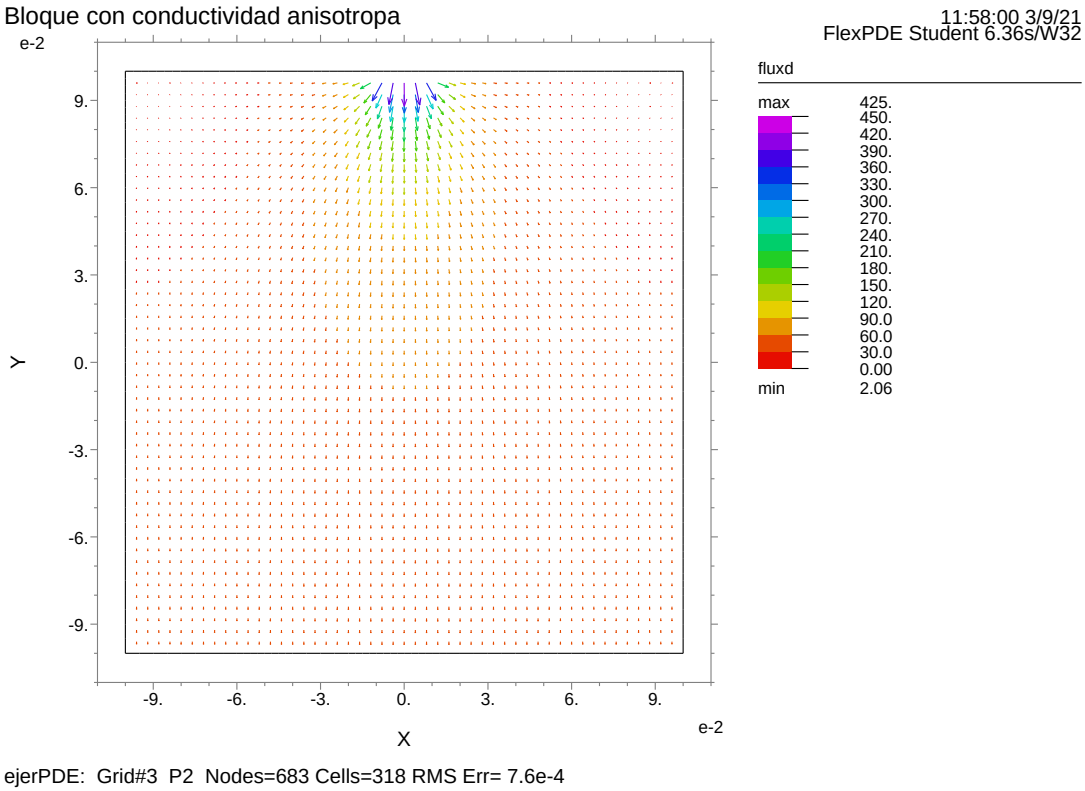


Figura 1.9: Flujo de calor