Matemáticas Computacionales
Practica 1: Gráficas de curvas en R

1860560 Rangel Delgado Jesus Angel

16 de Febrero del 2021

1. Introducción

1.1. Desarrollo de la practica

En esta primera práctica se hará una de las cosas básicas al momento de aprender R. Se repasaran las curvas en $\mathbb{R}^2$ vistas en primer semestre en la materia de Geometría Analítica. [1]. Se graficarán curvas como la recta, parábola, circunferencia, elipse e hipérbola.

2. Curvas de $\mathbb{R}^2$

2.1. Linea Recta

Def. Llamamos **linea recta** al lugar geométrico de los puntos tales que tomados dos puntos diferentes cualesquiera $P_1(x_1, y_1)$ y $P_2(x_2, y_2)$ de lugar, el valor de m se calcula por medio de:

$$m = \frac{y_1 - y_2}{x_1 - x_2} \tag{1}$$

Ecuación de la recta dada su pendiente y ordenada en el origen
Es la recta cuya pendiente **m** y cuya ordenada en el origen es **b**, que tiene por ecuación:

$$y = mx + b \tag{2}$$

Código en R
Para poder graficar esta función, solo basta con darle valor a la pendiente **m** y su ordenada **b**, además falta establecer un limites que serán el extremo al que llegara nuestra función.

Ejemplo 1:

```r
#Linea recta 1:
m <- 7 #pendiente
b <- 7 #interseccion

#funcion de la linea recta
f <- function(m, b, x){
  return(m * x + b)
}

x <- seq(-22, 22, 0.01)
y <- f(m, b, x) #Sirve para evaluar

plot(x, y, type = "l", xlab = "Eje X", ylab = "Eje Y") #graficamos
abline(h = 3, v = 3)
```

Figura 1: Código de la linea 1

y su gráfica seria:

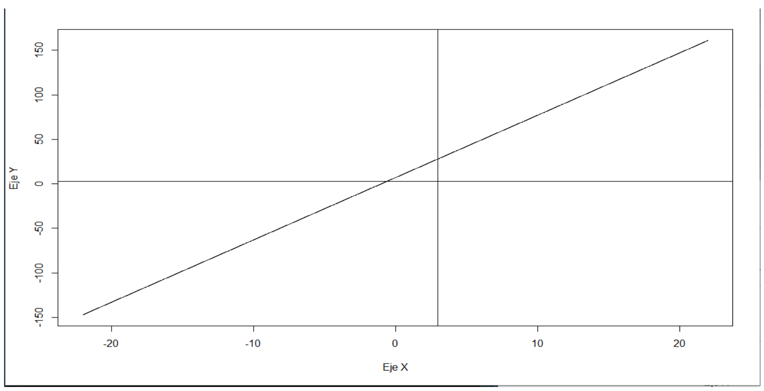


Figura 2: Gráfica del código 1

Ejemplo 2:

```
#funcion de la linea recta
f <- function(m, b, x){
  return(m * x + b)
}

x <- seq(-8, 8, 0.01)
y <- f(m, b, x)

plot(x, y, type = "l", xlab = "Eje X", ylab = "Eje Y")
abline(h = 1, v = 1)
```

Figura 3: Código de la linea 2

y su gráfica seria:

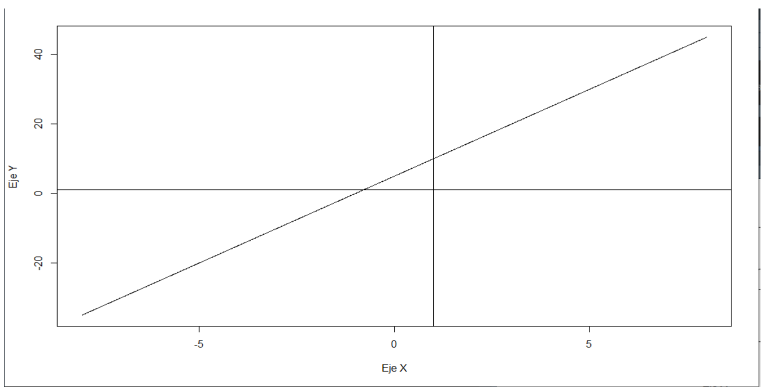


Figura 4: Gráfica del código 2

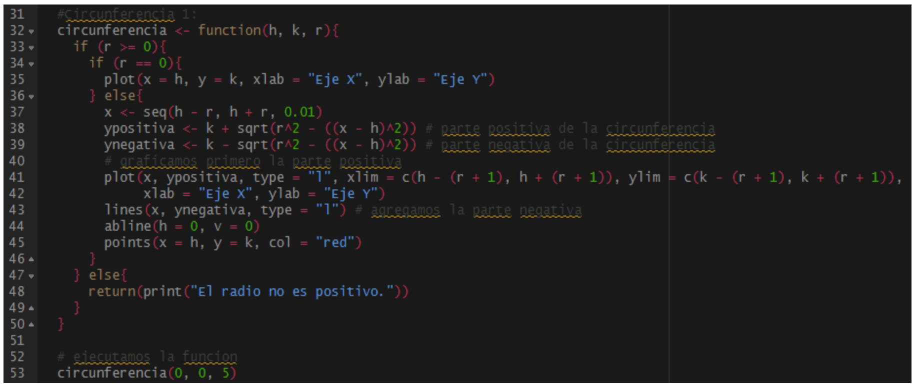
2.2. Circunferencia

Def. Circunferencia es el lugar geométrico de un punto que se mueve en un plano de tal manera que se conserva siempre a una distancia constante de un punto fijo de ese plano

TeoremaLa circunferencia cuyo centro es el punto (h,k) y cuyo radio es la constante r, tiene por ecuación:

$$(x-h)^2 + (y-k)^2 = r^2 \tag{3}$$

Código en R Para poder graficar una circunferencia en R primero tenemos que asegurarnos de que nuestra r o sea el radio de la circunferencia sea mayor que cero, después hay que comenzar a graficar por partes, nuestra circunferencia en este caso iniciamos con la parte positiva para después graficar la negativa, por ultimo se asignan valores a nuestra función, los valores para h, k y r.
Ejemplo 1:



```r
#Circunferencia 1:
circunferencia <- function(h, k, r){
  if (r >= 0){
    if (r == 0){
      plot(x = h, y = k, xlab = "Eje X", ylab = "Eje Y")
    } else{
      x <- seq(h - r, h + r, 0.01)
      ypositiva <- k + sqrt(r^2 - ((x - h)^2)) # parte positiva de la circunferencia
      ynegativa <- k - sqrt(r^2 - ((x - h)^2)) # parte negativa de la circunferencia
      # graficamos primero la parte positiva
      plot(x, ypositiva, type = "l", xlim = c(h - (r + 1), h + (r + 1)), ylim = c(k - (r + 1), k + (r + 1)),
           xlab = "Eje X", ylab = "Eje Y")
      lines(x, ynegativa, type = "l") # agregamos la parte negativa
      abline(h = 0, v = 0)
      points(x = h, y = k, col = "red")
    }
  } else{
    return(print("El radio no es positivo."))
  }
}

# ejecutamos la funcion
circunferencia(0, 0, 5)
```

Figura 5: Código de la circunferencia 1

y su gráfica seria:

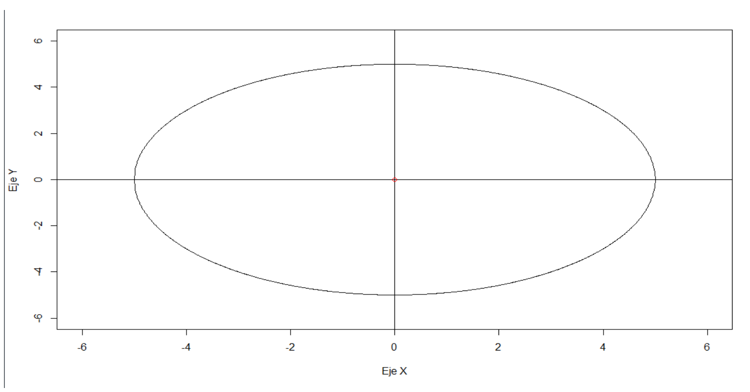


Figura 6: Gráfica de la circunferencia 1

Ejemplo 2:

```r
#Circunferencia 2
circunferencia <- function(h, k, r){
  if (r >= 0){
    if (r == 0){
      plot(x = h, y = k, xlab = "Eje X", ylab = "Eje Y")
    } else{
      x <- seq(h - r, h + r, 0.01)
      ypositiva <- k + sqrt(r^2 - ((x - h)^2)) # parte positiva de la circunferencia
      ynegativa <- k - sqrt(r^2 - ((x - h)^2)) # parte negativa de la circunferencia
      # graficamos primero la parte positiva
      plot(x, ypositiva, type = "l", xlim = c(h - (r + 1), h + (r + 1)), ylim = c(k - (r + 1), k + (r + 1)),
           xlab = "Eje X", ylab = "Eje Y")
      lines(x, ynegativa, type = "l") # agregamos la parte negativa
      abline(h = 0, v = 0)
      points(x = h, y = k, col = "red")
    }
  } else{
    return(print("El radio no es positivo."))
  }
}

# ejecutamos la funcion
circunferencia(1, 2, 3)
```

Figura 7: Código de la circunferencia 2

y su gráfica seria:

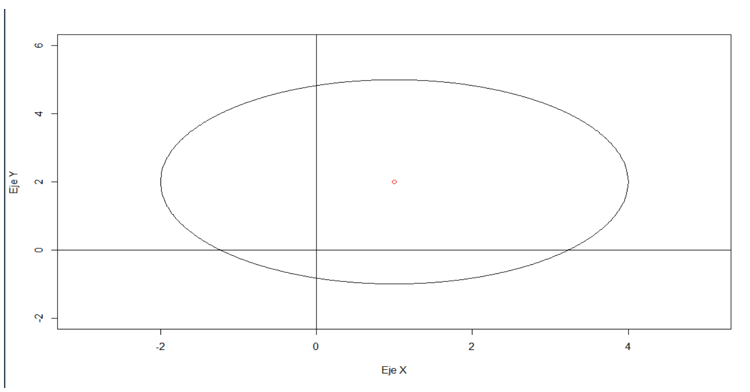


Figura 8: Gráfica de la circunferencia 2

2.3. Parábola

Def. Una parábola es un lugar geométrico de un punto que se mueve en un plano de tal manera que su distancia de una recta fija, situada en el plano, es siempre igual a su distancia de un punto fijo del plano y que no pertenece a la recta.

Las ecuaciones con vértice en el origen

$$y^2 = 4px \tag{4}$$

$$x^2 = 4py \tag{5}$$

Las ecuaciones con vértice fuera del origen

$$(y-k)^2 = 4p(x-h) \tag{6}$$

$$(x-h)^2 = 4p(y-k) \tag{7}$$

Código en R
Para poder graficar una Parábola en R, solo basta con introducir una ecuación que genere una parábola y al igual que en los casos anteriores introducir el código correspondiente para graficar.
Ejemplo 1

```
#Parabola 1
g <- function(x){
  return(3*x^2 + x - 6)
}

x <- seq(-6, 6, 0.01)#vector de -5 a 5
y <- g(x)

plot(x, y, type = "l", xlab = "Eje X", ylab = "Eje Y") #graficamos
abline(h = 0, v = 0) #una linea horizontal que pasa por el 0 en las x y una l
```

Figura 9: Código de la parábola 1

y su gráfica seria:

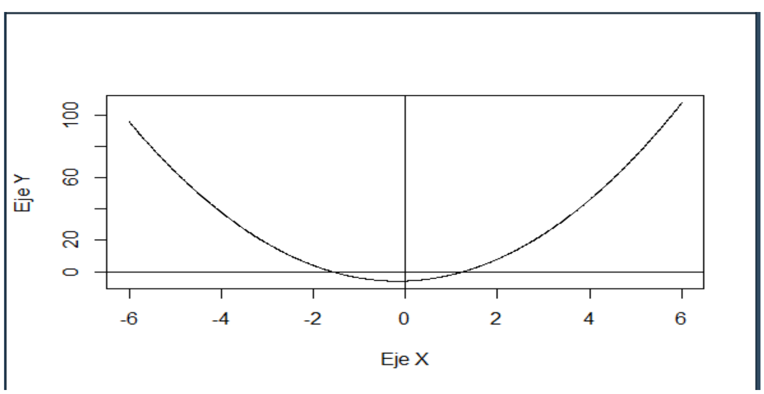


Figura 10: Gráfica de la parábola 1

Ejemplo 2:

```r
#Parabola 2
g <- function(x){
  return(4*x^2 + x - 8)
}

x <- seq(-8, 8, 0.01)#vector de -5 a 5
y <- g(x)

plot(x, y, type = "l", xlab = "Eje X", ylab = "Eje Y") #graficamos
abline(h = 1, v = 1) #una linea horizontal que pasa por el 0 en las x y una
```

Figura 11: Código de la parábola 2

y su gráfica seria:

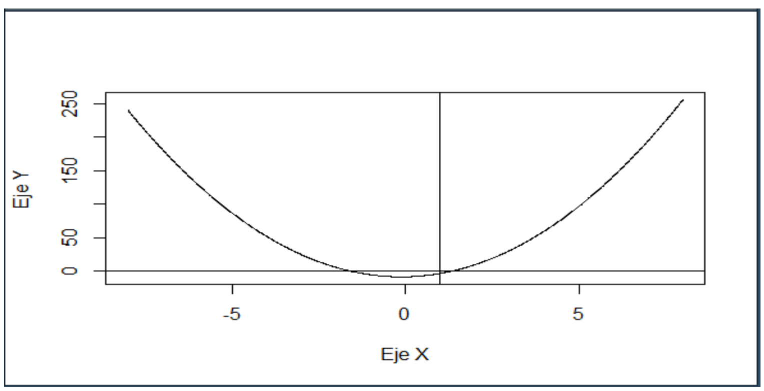


Figura 12: Gráfica de la parábola 2

2.4. Elipse

Def. Una elipse es el lugar geométrico de un punto que se mueve en un plano de tal manera que la suma de sus distancias a dos fijos de ese plano es siempre igual a una constante, mayor que la distancia entre los dos puntos.

Las ecuaciones con vértice en el origen

$$\frac{x^2}{a^2} + \frac{y^2}{b^2} = 1 \tag{8}$$

$$\frac{x^2}{b^2} + \frac{y^2}{a^2} = 1 \tag{9}$$

Las ecuaciones con vértice fuera del origen

$$\frac{(x-h)^2}{a^2} + \frac{(y-k)^2}{b^2} = 1 \tag{10}$$

$$\frac{(x-h)^2}{b^2} + \frac{(y-k)^2}{a^2} = 1 \tag{11}$$

Código en R

Para poder graficar una elipse en R, se sigue casi el mismo procedimiento que al graficar una circunferencia ya que se gráfica por partes, primero la positiva y luego la negativa, solo que a diferencia de la circunferencia esta tiene muchas condiciones a las cuales esta ligada y se tienen que cumplir. Por ultimo solo es cuestión de asignar valores como el centro, a y b para graficar la elipse

Ejemplo 1

```r
#Elipse 1
elipse <- function(h, k, a, b, horizontal){
  if (a > b){ # a tiene que ser mayor que b
    c <- sqrt(a^2 - b^2) # calculamos c
    if (horizontal){ # si es una elipse horizontal
      x <- seq(h - a, h + a, 0.01) #definimos el dominio
      ypositiva <- k + sqrt((b^2 - (b^2/a^2) * ((x - h)^2))) # parte positiva
      ynegativa <- k - sqrt((b^2 - (b^2/a^2) * ((x - h)^2))) # parte negativa
      # graficamos primero la parte positiva
      plot(x, ypositiva, type = "l", xlim = c(h - (a + 1), h + (a + 1)), ylim = c(k - (b + 1), k + (b + 1)),
           xlab = "Eje X", ylab = "Eje Y")
      lines(x, ynegativa, type = "l") # agregamos la parte negativa
      abline(h = 0, v = 0) # ejes coordenados
      points(x = c(h - c, h + c), y = c(k, k), col = "red") # focos
    } else{
      x <- seq(h - b, h + b, 0.01)
      ypositiva <- k + sqrt((a^2 - (a^2/b^2) * ((x - h)^2)))
      ynegativa <- k - sqrt((a^2 - (a^2/b^2) * ((x - h)^2)))
      plot(x, ypositiva, type = "l", xlim = c(h - (b + 1), h + (b + 1)), ylim = c(k - (a + 1), k + (a + 1)),
           xlab = "Eje X", ylab = "Eje Y")
      lines(x, ynegativa, type = "l")
      abline(h = 0, v = 0)
      points(x = c(h, h), y = c(k - c, k + c), col = "red")
    }
  } else {
    return(print("No cumple las condiciones para ser una elipse. (a no es mayor que b)"))
  }
}

elipse(0, 0, 9, 4, TRUE)
```

Figura 13: Código de la elipse 1

y su gráfica seria:

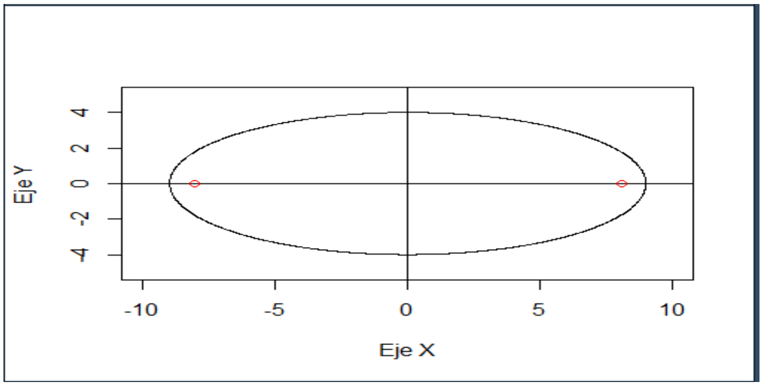


Figura 14: Gráfica de la elipse 1

Ejemplo 2:

```r
#Elipse 2
elipse <- function(h, k, a, b, horizontal){
  if (a > b){ # a tiene que ser mayor que b
    c <- sqrt(a^2 - b^2) # calculamos c
    if (horizontal){ # si es una elipse horizontal
      x <- seq(h - a, h + a, 0.01) #definimos el dominio
      ypositiva <- k + sqrt((b^2 - (b^2/a^2) * ((x - h)^2))) # parte positiva
      ynegativa <- k - sqrt((b^2 - (b^2/a^2) * ((x - h)^2))) # parte negativa
      # graficamos primero la parte positiva
      plot(x, ypositiva, type = "l", xlim = c(h - (a + 1), h + (a + 1)), ylim = c(k - (b + 1), k + (b + 1)),
           xlab = "Eje X", ylab = "Eje Y")
      lines(x, ynegativa, type = "l") # agregamos la parte negativa
      abline(h = 0, v = 0) # ejes coordenados
      points(x = c(h - c, h + c), y = c(k, k), col = "red") # focos
    } else{
      x <- seq(h - b, h + b, 0.01)
      ypositiva <- k + sqrt((a^2 - (a^2/b^2) * ((x - h)^2)))
      ynegativa <- k - sqrt((a^2 - (a^2/b^2) * ((x - h)^2)))
      plot(x, ypositiva, type = "l", xlim = c(h - (b + 1), h + (b + 1)), ylim = c(k - (a + 1), k + (a + 1)),
           xlab = "Eje X", ylab = "Eje Y")
      lines(x, ynegativa, type = "l")
      abline(h = 0, v = 0)
      points(x = c(h, h), y = c(k - c, k + c), col = "red")
    }
  } else {
    return(print("No cumple las condiciones para ser una elipse. (a no es mayor que b)"))
  }
}

elipse(1, 1, 3, 1, TRUE)
```

Figura 15: Código de la elipse 2

y su gráfica seria:

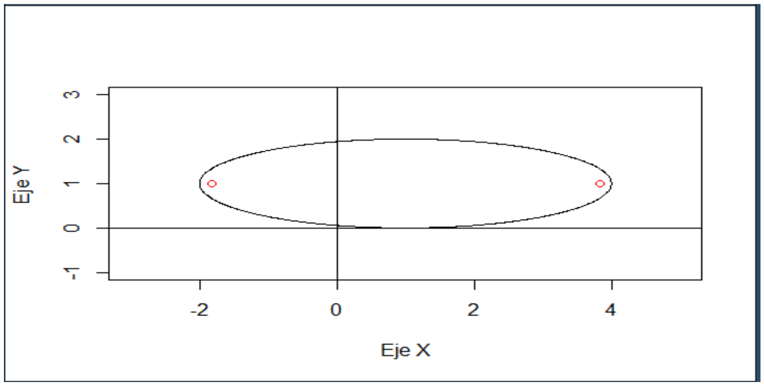


Figura 16: Gráfica de la elipse 2

2.5. Hipérbola

Def. Una hipérbola es el lugar geométrico de un punto que se mueve en un plano de tal manera que el valor absoluto de la diferencia de sus distancias a dos puntos fijos del plano, llamados focos, es siempre igual a una cantidad constante, positiva y menor que la distancia entre los focos

Ecuaciones con vértice en el origen

$$\frac{x^2}{a^2} - \frac{y^2}{b^2} = 1 \tag{12}$$

$$\frac{x^2}{b^2} - \frac{y^2}{a^2} = 1 \tag{13}$$

Ecuaciones con vértice fuera del origen

$$\frac{(x+h)^2}{a^2} - \frac{(y+k)^2}{b^2} = 1 \tag{14}$$

$$\frac{(x+h)^2}{b^2} - \frac{(y+k)^2}{a^2} = 1 \tag{15}$$

Código en R

Para poder graficar una hipérbola en R es necesario crear una función que nos ayude a descifrar como se graficara nuestra función una vez que se ingresen los valores del centro (h, k) y los valores de a y b, es importante que en la función se agregue ciertas condiciones para que se creen las hipérbolas en ambos lados.

Ejemplo 1:

```r
#Hiperbola 1
hiperbola <- function(h, k, a, b, horizontal){
  c <- sqrt(a^2 + b^2) # calculamos c
  if (horizontal){ # hiperbola sobre el eje x
    xizq <- seq(h - (a + 3), h - a, 0.01) # dominio izquierdo
    xder <- seq(h + a, h + (a + 3), 0.01) # dominio derecho
    yizqpositiva <- k + sqrt((b^2/a^2)*((xizq - h)^2) - b^2) # parte positiva del dominio izquierdo
    yizqnegativa <- k - sqrt((b^2/a^2)*((xizq - h)^2) - b^2) # parte negativa del dominio izquierdo
    yderpositiva <- k + sqrt((b^2/a^2)*((xder - h)^2) - b^2) # parte positiva del dominio derecho
    ydernegativa <- k - sqrt((b^2/a^2)*((xder - h)^2) - b^2) # parte negativa del dominio derecho
    # greficamos la parte positiva del dominio izquierdo
    plot(xizq, yizqpositiva, type = "l", xlim = c(h - (a + 4), h + (a + 4)), ylim = c(k - (b + 4), k + (b + 4)),
         xlab = "Eje X", ylab = "Eje Y")
    #Resto de la grafica
    lines(xizq, yizqnegativa, type = "l")
    lines(xder, ydernegativa, type = "l")
    lines(xder, yderpositiva, type = "l")

    abline(h = 0, v = 0) # ejes coordenados
    points(x = c(h - (a + c)), y = c(k), col = "red") # focos
```

Figura 17: Código de la hipérbola 1 parte 1

```r
  } else{ # hiperbola sobre el eje y
    yizq <- seq(k - (a + 3), k - a, 0.01) # rango inferior
    yder <- seq(k + a, k + (a + 3), 0.01) # rango superior
    xizqpositiva <- h + sqrt((b^2/a^2)*((yizq - k)^2) - b^2)
    xizqnegativa <- h - sqrt((b^2/a^2)*((yizq - k)^2) - b^2)
    xderpositiva <- h + sqrt((b^2/a^2)*((yder - k)^2) - b^2)
    xdernegativa <- h - sqrt((b^2/a^2)*((yder - k)^2) - b^2)
    # graficamos
    plot(xizqpositiva, yizq, type = "l", xlim = c(h - (b + 4), h + (b + 4)), ylim = c(k - (a + 4), k + (a + 4)),
         xlab = "Eje X", ylab = "Eje Y")
    lines(xizqnegativa, yizq, type = "l")
    lines(xdernegativa, yder, type = "l")
    lines(xderpositiva, yder, type = "l")
    abline(h = 1, v = 1)
    points(x = c(h), y = c(k - (a + c)), col = "red") # focos
  }
}

hiperbola(5, -3, 4, 8, FALSE)
```

Figura 18: Código de la hipérbola 1 parte 2

y su gráfica seria:

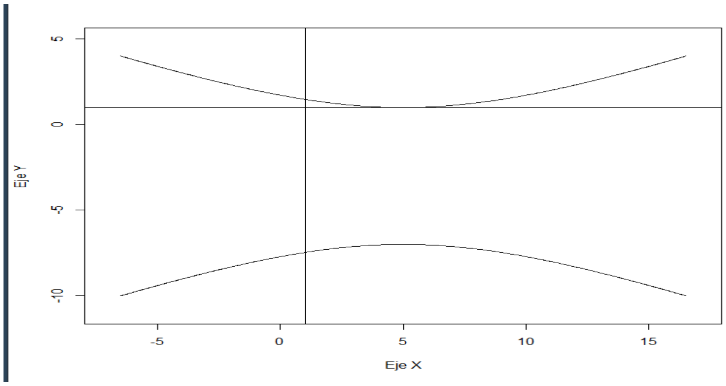


Figura 19: Gráfica de la hipérbola 1

Ejemplo 2:

```r
#Hiperbola 2
hiperbola <- function(h, k, a, b, horizontal){
  c <- sqrt(a^2 + b^2) # calculamos c
  if (horizontal){ # hiperbola sobre el eje x
    xizq <- seq(h - (a + 3), h - a, 0.01) # dominio izquierdo
    xder <- seq(h + a, h + (a + 3), 0.01) # dominio derecho
    yizqpositiva <- k + sqrt((b^2/a^2)*((xizq - h)^2) - b^2) # parte positiva del dominio izquierdo
    yizqnegativa <- k - sqrt((b^2/a^2)*((xizq - h)^2) - b^2) # parte negativa del dominio izquierdo
    yderpositiva <- k + sqrt((b^2/a^2)*((xder - h)^2) - b^2) # parte positiva del dominio derecho
    ydernegativa <- k - sqrt((b^2/a^2)*((xder - h)^2) - b^2) # parte negativa del dominio derecho
    # greficamos la parte positiva del dominio izquierdo
    plot(xizq, yizqpositiva, type = "l", xlim = c(h - (a + 4), h + (a + 4)), ylim = c(k - (b + 4), k + (b + 4)),
         xlab = "Eje X", ylab = "Eje Y")
    #Resto de la grafica
    lines(xizq, yizqnegativa, type = "l")
    lines(xder, ydernegativa, type = "l")
    lines(xder, yderpositiva, type = "l")

    abline(h = 0, v = 0) # ejes coordenados
    points(x = c(h - (a + c)), y = c(k), col = "red") # focos
```

Figura 20: Código de la hipérbola 2 parte 1

```r
  } else{ # hiperbola sobre el eje y
    yizq <- seq(k - (a + 3), k - a, 0.01) # rango inferior
    yder <- seq(k + a, k + (a + 3), 0.01) # rango superior
    xizqpositiva <- h + sqrt((b^2/a^2)*((yizq - k)^2) - b^2)
    xizqnegativa <- h - sqrt((b^2/a^2)*((yizq - k)^2) - b^2)
    xderpositiva <- h + sqrt((b^2/a^2)*((yder - k)^2) - b^2)
    xdernegativa <- h - sqrt((b^2/a^2)*((yder - k)^2) - b^2)
    # graficamos
    plot(xizqpositiva, yizq, type = "l", xlim = c(h - (b + 4), h + (b + 4)), ylim = c(k - (a + 4), k + (a + 4)),
         xlab = "Eje X", ylab = "Eje Y")
    lines(xizqnegativa, yizq, type = "l")
    lines(xdernegativa, yder, type = "l")
    lines(xderpositiva, yder, type = "l")
    abline(h = 1, v = 1)
    points(x = c(h), y = c(k - (a + c)), col = "red") # focos
  }
}

hiperbola(5, 5, 5, 2.64, FALSE)
```

Figura 21: Código de la hipérbola 2 parte 2

y su gráfica seria:

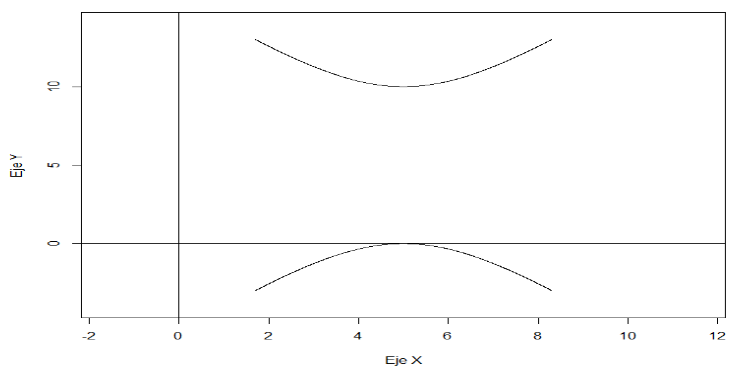


Figura 22: Gráfica de la hipérbola 2

Referencias

[1] Charles H. Lemhann. Geometria Analitica, 1965.

[2] Jesus Rangel, repositorio de GitHub https://github.com/JesusRangel07/MatematicasComputacionales Repositorio de Github