Nombres:

- Geraldine Gómez.
- Bryan González.
- Santiago Roa.

Taller 1

1. Método de Horner:

```
rm(list=ls())
Horner <-function(P,Xn)
{
    y <- 0
    i <- 1
    while (i < length(P) + 1)
    {
        y = Xn*y + P[i]
        cat("y= ",y,"\tXn=",Xn,"\tOperaciones Mínimas = ",i*2,"\n")
        i <- i + 1

    }


    cat("El valor del polinomio es: ",y)

  }
```

a. $2x^4 - 3x^2 + 3x - 4$

$x_0 = -2$

```
P <- c (2,0,-3,3,-4)
Horner(P ,-2)
```

Resultado:

```
y=  2   Xn= -2  Operaciones Minimas =  2
y=  -4  Xn= -2  Operaciones Minimas =  4
y=  5   Xn= -2  Operaciones Minimas =  6
y=  -7  Xn= -2  Operaciones Minimas =  8
y=  10  Xn= -2  Operaciones Minimas =  10
El valor del polinomio es:  10
```

b. $7x^5 + 6x^4 - 6x^3 + 3x - 4$
$x_0 = 3$

```
P2 <- c(7,6-6,0,3,-4)
Horner(P2,3)
```

Resultado:

```
y=  7           xn= 3   Operaciones Mínimas =  2
y=  21          xn= 3   Operaciones Mínimas =  4
y=  63          xn= 3   Operaciones Mínimas =  6
y=  192         xn= 3   Operaciones Mínimas =  8
y=  572         xn= 3   Operaciones Mínimas =  10
El valor del polinomio es:  572
```

c. $-5x^6 + 3x^4 + 2x^2 - 4x$
$x_0 = -1$

```
P3 <- c(-5,0,3,0,2,-4,0)
Horner(P3,-1)
```

Resultado:

```
y=  -5          xn= -1  Operaciones Mínimas =  2
y=  5           xn= -1  Operaciones Mínimas =  4
y=  -2          xn= -1  Operaciones Mínimas =  6
y=  2           xn= -1  Operaciones Mínimas =  8
y=  0           xn= -1  Operaciones Mínimas =  10
y=  -4          xn= -1  Operaciones Mínimas =  12
y=  4           xn= -1  Operaciones Mínimas =  14
El valor del polinomio es:  4
```

2.

```
rm(list=ls())
eficiencia <- function(n)
{
 x <- n
 i <- 0
 while ( n > 0)
 {
  d <- n%%2
  n <- floor(n/2)
  i <- i+1
  cat("i= ",i*2,"\tn= ",n,"\t","d= ",d,"\n")
 }
}
```

a. **n = 73**

eficiencia (73)

Resultado:

i=	2	n=	36	d=	1
i=	4	n=	18	d=	0
i=	6	n=	9	d=	0
i=	8	n=	4	d=	1
i=	10	n=	2	d=	0
i=	12	n=	1	d=	0
i=	14	n=	0	d=	1

b.

T(n) = 2 + k (k: número de iteraciones necesarias).

$\mathbf{n_1}$ = n/2
$\mathbf{n_2}$= n/4
$\mathbf{n_3}$ = n/8
$\mathbf{n_k}$ = 1

$\mathbf{n/2^k < 2}$ (Tamaño de n después de k iteraciones).
$n/2^k < 2$
$\mathbf{n < 2^{k+1}}$ (Fórmula de acotamiento).
$\ln(n) < \ln(2^{k+1})$
$\ln(n) < (k+1) * \ln(2)$
$k > \ln(n) / \ln(2) - 1$

$\mathbf{O = O(\log_2(n))}$

3. **R(t) = (2cos(t), sen(t),0), P (2,1,0)**

f(t) = d’
f´(t) = d’’

d = $\sqrt{(2\cos(t)-2)^2+(\sin(t)-1)^2(t)+(0-0)^2}$

d’= $\dfrac{8\sin(t)-3\sin(2t)-2\cos(t)}{2\sqrt{4\,cos^2(t)+sin^2(t)-8\cos(t)+2\sin(t)+5}}$

d’ = 3 sin(t)cos(t) – 4sin(t) + cos(t) = 0

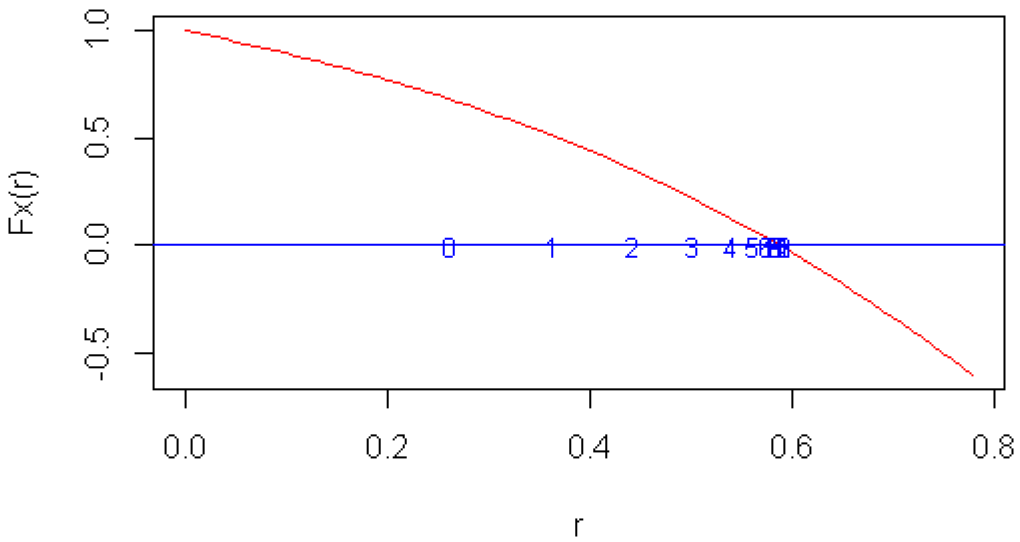
d’’= - sin(t) - 4cos(t) - 3cos(2t)

```r
rm(list=ls())
Fx <- function(t) 3*sin(t)*cos(t)-4*sin(t)+cos(t)
Dx <- function(t) -sin(t) -4*cos(t) -3*cos(2*t)
Hx <- function(t) t - (Fx(t)/Dx(t))

 newton <- function(a,b)
 {


   if((Fx(a)-a)*(Fx(b)-b)<0)
   {
     r<-seq(a,b,0.01)
     plot(r,Fx(r),type="l",col="red")
     abline(h=0,col="blue")


     t<-0
     r<-Hx(t)
     i<-0
     while (Fx(r) != 0 )
     {

       error<-abs(Fx(t)/Dx(t))

       if(error >  1.e-4)
         t<-r
       else break
       r<-Hx(t)

       text(r,0,i,cex=0.8,col="blue")

       i<-i+1

       cat("I=",i,"\tF(t) =",Fx(t),"\tT=",signif(t, digits = 4),"\tE=",error,"\n")
     }

   }
   else
   {
     cat("No tiene raíz la función en ese intervalo.\n")
   }

 }

 newton(0,pi/4)
```

Resultado:

```
I= 1     F(t) = 0.8430906          T= 0.1429      E= 0.1428571
I= 2     F(t) = 0.6778332          T= 0.2636      E= 0.1207865
I= 3     F(t) = 0.507396           T= 0.3646      E= 0.1009444
I= 4     F(t) = 0.3469569          T= 0.4447      E= 0.08014699
I= 5     F(t) = 0.2143954          T= 0.5032      E= 0.05850295
I= 6     F(t) = 0.1200839          T= 0.5416      E= 0.03834727
I= 7     F(t) = 0.06207925         T= 0.564       E= 0.02245124
I= 8     F(t) = 0.03039057         T= 0.576       E= 0.01193781
I= 9     F(t) = 0.0144131          T= 0.5819      E= 0.005936326
I= 10    F(t) = 0.006723327        T= 0.5847      E= 0.002837908
I= 11    F(t) = 0.003110912        T= 0.5861      E= 0.001328921
I= 12    F(t) = 0.00143391         T= 0.5867      E= 0.0006160144
I= 13    F(t) = 0.0006597475       T= 0.587       E= 0.0002841785
I= 14    F(t) = 0.000303301        T= 0.5871      E= 0.0001308026
```

Gráfica:



4. **r = 2 + cos(3t)**
   r = 2 – e^{t}

   a. **Posición Falsa – Intersección:**

```
#Graficar en Polares

rm(list=ls())
polar <- function (theta, r, color=4)
{
  y <- 0
  x <- 0
  ejex <- 1
```

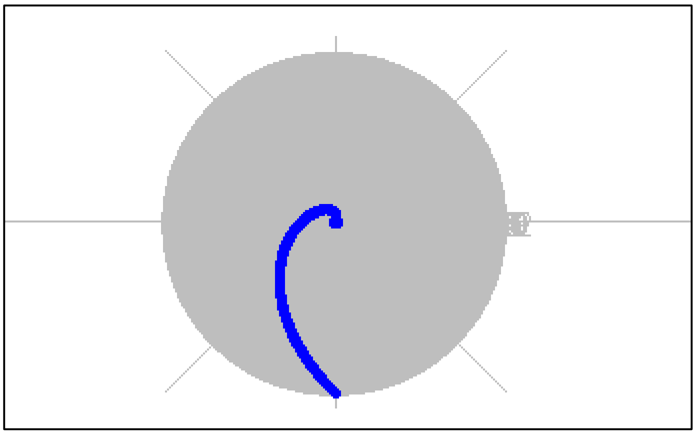
```
  for (i in 1:length(r))
 {
   if(is.nan(r[i])== T)
   {
     r[i] <- 0
   }
  }

  angulo <- seq(-max(theta),max(theta),by=theta[2]-theta[1])
  y <- r*sin(theta)
  x <- r*cos(theta)
  plot.new()
  plot.window(xlim = c(-max(r), max(r)), ylim = c(-max(r), max(r)), asp
= 1)

  aux <- max(r)
  # Dibuja los ejes.
  while (aux > 0)
  {
    fi <- aux*sin(angulo)
    cir <- aux*cos(angulo)
    points(cir,fi,pch="-",col="gray",cex=0.3)
    text(ejex+0.2,-0.2,ejex,col="gray")
    ejex <- ejex + 1
    aux <- aux - 1
    }

  abline(v=((max(cir)+min(cir))/2),col="gray")
  abline(h=((max(cir)+min(cir))/2),col="gray")
  segments(-max(r)+0.5,-max(r)+0.5,max(r)-0.5,max(r)-0.5,col="gray")
  segments(-max(r)+0.5,max(r)-0.5,max(r)-0.5,-max(r)+0.5,col="gray")

  points(x,y,pch=20,col=color,cex=1)
}

dim <- seq(-pi, 3*pi/2, by=pi/600)
r <- cos(3*dim) - exp(dim)
polar(dim,r,"blue")


#  Posicion falsa - Intersección

    rm(list=ls())
    Fr<-function(x) 2+cos(3*x)
    Gr<-function(x) 2-exp(x)
    regulaFalsi <- function(a,b)
   {
```

```
  x<-((Fr(b)*a)-(Fr(a)*b))/(Fr(b)-Fr(a))
  error<-abs(Fr(x)/Gr(x))
  while(error>1.e-4)
 {
      x<-((Fr(b)*a)-(Fr(a)*b))/(Fr(b)-Fr(a))
      if(Fr(x)==0)
      {
        break
      }
      if(Fr(x)*Fr(a)<0)
      {
        b<-x
      }
      else
      {
        a<-x
      }
      error<-abs(Fr(x)/Gr(x))
      cat("r=",x,"    \t","E=",error,"\n")
   }
   }
   regulaFalsi(pi,3*pi/2)
```

Resultado:

```
r= 1.570796                 E= 0.7116229
r= 2.021211e+15             E= 0
```

Gráfica:



b. Secante - Intersección

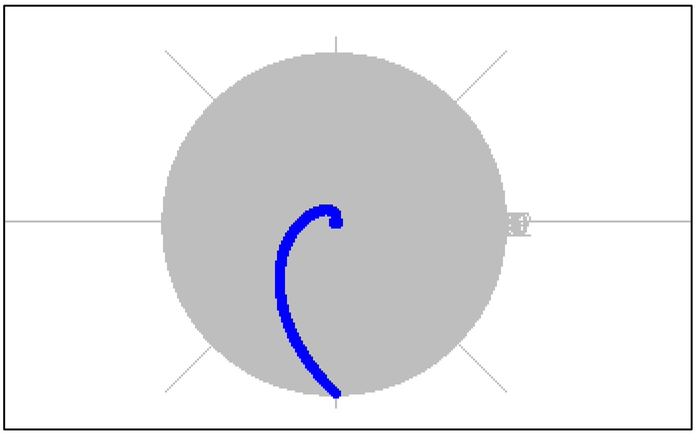
```
#Graficar en Polares

rm(list=ls())
polar <- function (theta, r, color=4)
{
  y <- 0
  x <- 0
  ejex <- 1

  for (i in 1:length(r))
 {
    if(is.nan(r[i])== T)
    {
      r[i] <- 0
    }
  }

  angulo <- seq(-max(theta),max(theta),by=theta[2]-theta[1])
  y <- r*sin(theta)
  x <- r*cos(theta)
  plot.new()
  plot.window(xlim = c(-max(r), max(r)), ylim = c(-max(r), max(r)), asp
= 1)

  aux <- max(r)
  # Dibuja los ejes.
  while (aux > 0)
  {
    fi <- aux*sin(angulo)
    cir <- aux*cos(angulo)
    points(cir,fi,pch="-",col="gray",cex=0.3)
    text(ejex+0.2,-0.2,ejex,col="gray")
    ejex <- ejex + 1
    aux <- aux - 1
    }

  abline(v=((max(cir)+min(cir))/2),col="gray")
  abline(h=((max(cir)+min(cir))/2),col="gray")
  segments(-max(r)+0.5,-max(r)+0.5,max(r)-0.5,max(r)-0.5,col="gray")
  segments(-max(r)+0.5,max(r)-0.5,max(r)-0.5,-max(r)+0.5,col="gray")

  points(x,y,pch=20,col=color,cex=1)
}

dim <- seq(-pi, 3*pi/2, by=pi/600)
```

```
r <- cos(3*dim) - exp(dim)
polar(dim,r,"blue")

 rm(list=ls())
 Fx <- function(x) 2+cos(3*x)
 Gx <- function(x) 2-exp(x)


 #Secante – Intersección

 secante <- function(a,b)
 {
  x<-(Fx(b)*a-Fx(a)*b)/(Fx(b)-Fx(a))
  error <-1
  while (error > 1.e-4)
  {
    a<-b
    b<-x
    x<-(Fx(b)*a-Fx(a)*b)/(Fx(b)-Fx(a))

    if (Fx(x) == 0) break

    error<-abs(Fx(x)/Gx(x))
    cat("r=",x,"\t","E=",error,"\n")
  }
}
secante(pi,3*pi/2)
```

Resultado:

```
r= 2.021211e+15          E= 0
```

Gráfica:



5.

13.

a. **Formula iterativa:**

f(x) = $x - \frac{x^n - r}{nx^{(n-1)}}$
x_0 $= \frac{v}{2}$
x_1 = f(x_0)
x_2 = f(x_1)
x_3 = f(x_2)
x_{k+1} = f (xk), **k**: creciente

n: Raíz enésima
r: Número al cual se le va a calcular la raíz enésima

Demostración:

g(x) $= x^n - r$
g'(x) $= n\, e^{n-1}$
tg(e) $= e - \frac{e^n - r}{n\, e^{n-1}}$

Teorema de Convergencia cuadrática:

$$\begin{aligned} \boldsymbol{e_{n+1}} &= |x_n - \sqrt[n]{r}| \\ &= |(x_n^n - r) - (s^n - r)| \\ &= |(n\, s^{n-1})(x_n - s) + \frac{1}{n}(n(n-1))s^{n-2}(x_n - s)^2| \\ &= 0 + \frac{1}{n}(n(n-1))s^{n-2}e^2 n \\ &\lim_{n\to\infty} \frac{1}{n}|n^2 - n\,(s^{n-2})| \neq \infty \end{aligned}$$

Intervalo de Convergencia:

$r = 2^n r_1$

$\frac{1}{n} < \sqrt[n]{r} < 1$

b. Algoritmo:

```
Fx <- function(x,n,v) x-(x^n-v)/(n*x^(n-1))
Gx <- function(x,n,v) x^n-v
Hx <- function(x,n) n*x^(n-1)

  calcularRaiz <- function(v,n)
  {

    x <- v/2
    error <- 1
    i <- 0
    while(error > 1.e-8)
    {

      x <- Fx(x,n,v)

      error <- abs(Gx(x,n,v))/Hx(x,n)
      cat(" x= ",x," \terror= ",error,"\n")
      i <- i + 1

    }


  }
```

- Se toma como valor de prueba $\sqrt[3]{120}$

  calcularRaiz (125,3)

Resultado:

```
x=  40.01111   error=  13.31205
x=  26.69906   error=  8.843573
x=  17.85549   error=  5.826366
x=  12.02912   error=  3.733273
x=  8.295849   error=  2.184066
x=  6.111783   error=  0.966422
x=  5.145361   error=  0.204246
x=  4.941115   error=  0.008675378
x=  4.932439   error=  1.527651e-05
x=  4.932424   error=  4.731362e-11
```

Calculo con calculadora: 4,93242414

14.

a. Las condiciones para que la raíz exista, sea única y pueda ser calculada son:

1. La función debe ser continua en el intervalo [a, b].
2. El valor de x debe existir en el intervalo [a, b].
3. Debe existir un cambio de signo cuando se evalúa el valor a y b en la función.
4. Debe tener una pendiente m < 1 para que exista una única raíz.

b. El orden de convergencia es lineal.

c. Algoritmo:

```
rm(list=ls())
Fx <- function(x) 2*x^4-3*x^2+3*x-4

RaizReal <- function(a, b)
{
  if(a/b < 1){
    x<-a
    d<-(b-a)/10
    x<-(x+d)
    E<- 1.e-4
    while (d > E)
    {
      if (Fx(a)*Fx(b) < 0)
      {
        x<-(x-d)
        d<-d/10
      }
      cat("x= ",x,"   \tE= ",d,"\n")
    }

  }
  else
  {
    cat("La función no tiene raiz real positiva.")
 }

}
RaizReal(1, 2)
```

Resultado:

```
x=  1           E=  0.01
x=  0.99        E=  0.001
x=  0.989       E=  1e-04
```

15.

a. **Algoritmo:**

```
rm(list=ls())
Fx <- function(x) 5*x-exp(x)-1
Gx <- function(x) (exp(x)+1)/5

puntoFijo <- function(a,b)
{
  x<-(a+b)/2
  i<-0
  while (Gx(x) != x )
  {
    error<-abs(a-b)/2
    if(error > 1.e-8)
      if (Gx(x) < x) b <- x
          else
          {
            a <- x
          }
    else
   {
      break
   }
    if(i>5)
    {
      break
    }
      x<-(a+b)/2
```
– **Condición de convergencia del punto fijo**

```
      i<-i+1
      cat("I=",i,"\tG(x) =",Gx(x),"      \tX=",x,"\tE=",error,"\n")

  }

}
```

puntoFijo(0.5,0.8) – **Intervalo de convergencia.**

Resultado:

I= 1	G(X) = 0.5554261	X= 0.575	E= 0.15
I= 2	G(X) = 0.5423444	X= 0.5375	E= 0.075
I= 3	G(X) = 0.548824	X= 0.55625	E= 0.0375
I= 4	G(X) = 0.545569	X= 0.546875	E= 0.01875
I= 5	G(X) = 0.5439529	X= 0.5421875	E= 0.009375
I= 6	G(X) = 0.54476	X= 0.5445313	E= 0.0046875