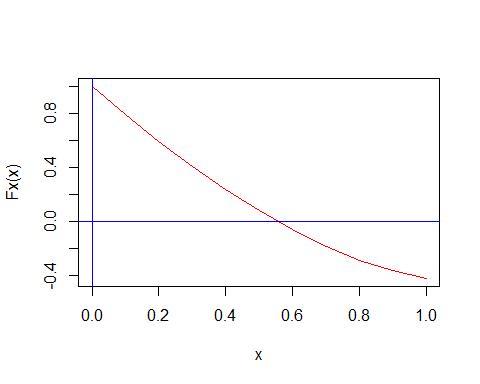
|  |
| --- |
| title: “R Notebook” |
| output: html\_notebook |

Biseccion e^x - pi\*x

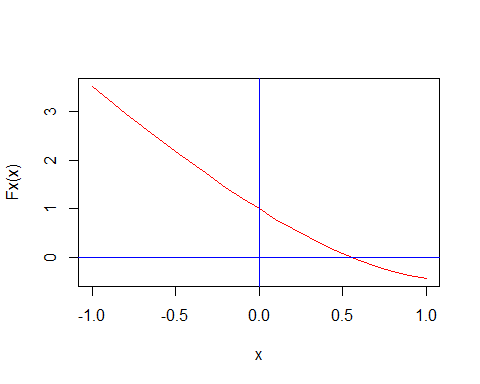
rm(list=ls())  
Fx = function(x){  
 return ((exp(x)) - (pi\*x))  
}  
  
biseccion = function(a, b, error){  
 #Para graficar se crea una secuencia de números entre el rango [a,b]  
 x = seq(a, b, 0.1)  
 plot(x, Fx(x), type = "l", col = "red")  
 abline(h = 0, v = 0, col = "blue")  
 xr = 0  
 fxr = 0  
 contador = 0  
 if (Fx(a) \* Fx(b) > 0){  
 cat("No se puede aplicar el método")  
 }  
 else {  
 xr = (a + b) / 2.0  
 fxr = Fx(xr)  
 if (fxr <= error){  
 break  
 } else {  
 if (fxr \* Fx(a) > 0)  
 a = xr  
 if (fxr \* Fx(b) > 0)  
 b = xr  
 }  
 contador = contador + 1  
 }  
 cat("Iteracciones: ", contador, "Resultado: ", xr, "\n")  
}  
  
biseccion(0, 1, 10e-8)



## Iteracciones: 1 Resultado: 0.5

Método de Newton e^x - pi \* x Derivada

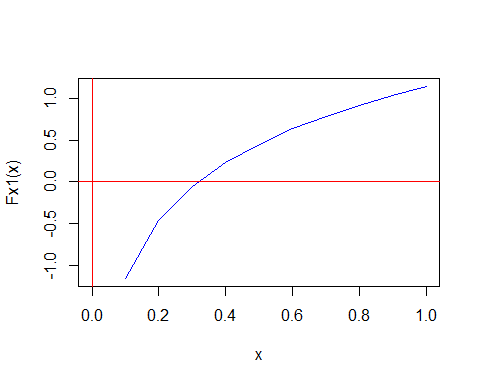
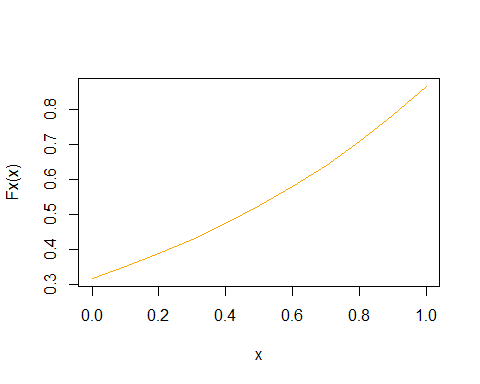
rm(list=ls())  
Fx = function(x) exp(x) - pi \* x  
Fx1 = function(x) exp(x) - pi  
  
Newton = function(a, b, error){  
 x = seq(a, b, 0.1)  
 plot(x, Fx(x), type = "l", col = "red")  
 abline(h = 0, v = 0, col = "blue")  
   
 x\_0 = (a + b) / 2  
   
 contador = 0  
 dx = 0  
 repeat {  
 corr = Fx(x\_0) / Fx1(x\_0)  
 x\_1 = x\_0 - corr  
 dx = abs(corr)  
 x\_0 = x\_1  
 contador = contador + 1  
   
 cat(contador, dx, "\n")  
 if (dx <= error)  
 break  
 }  
 cat ("Iteracciones: ", contador, "Resultado: ", x\_1, "\n")  
}  
Newton(-1, 1, 10e-8)



## 1 0.4669422   
## 2 0.08287642   
## 3 0.003998516   
## 4 9.896266e-06   
## 5 6.078286e-11   
## Iteracciones: 5 Resultado: 0.553827

Método del Punto Fijo

rm(list=ls())  
Fx = function(x) exp(x) / pi  
Fx1 = function(x) log(x\*pi)  
  
PuntoFijo = function(a, b, error){  
 xInicial = a  
 x = seq(a, b, 0.1)  
 plot(x, Fx(x), type = "l", col = "orange")  
 plot(x, Fx1(x), type = "l", col = "blue")  
 abline(h = 0, v = 0, col = "red")  
 if (Fx(a) < a || Fx(b) < b)  
 cat("El intervalo no es valido\n")  
 else {  
 x\_0 = (a + b) / 2  
 contador = 0  
 fxInicial = Fx(a)  
 done = FALSE  
   
 while(abs(xInicial - fxInicial) > error){  
 contador = contador + 1  
   
 if (xInicial < a){  
 done = TRUE  
 }  
 if (done == FALSE){  
 xInicial = fxInicial  
 fxInicial = Fx(xInicial)  
 } else {  
 fxInicial = xInicial  
 xInicial = Fx1(fxInicial)  
 }  
 }  
 cat("Iteracciones: ", contador, "Resultado: ", xInicial, "\n")   
 }  
}  
  
PuntoFijo(0, 1, 10e-8)



## El intervalo no es valido

Método de la secante

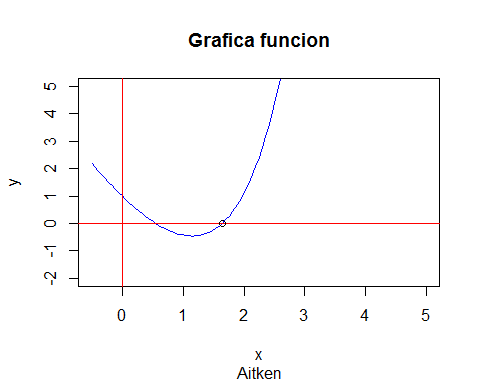
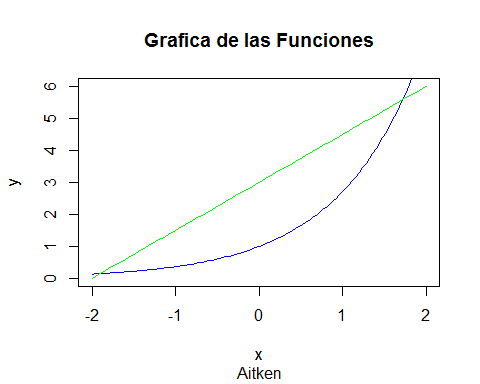
rm(list=ls())  
Fx = function(x) exp(x) - pi \* x  
Fx1 = function(x) exp(x) - pi  
  
Secante = function(x1, x2, error){  
 x = (Fx(x2) \* x1 - Fx(x1) \* x2) / (Fx(x2) - Fx(x1))  
 err = 1  
 contador = 0  
 while (err > error){  
 contador = contador + 1  
 x1 = x2  
 x2 = x  
 x = (Fx(x2) \* x1 - Fx(x1) \* x2) / (Fx(x2) - Fx(x1))  
 if (Fx(x) == 0)  
 break  
 err = abs(Fx(x) / Fx1(x))  
 cat("Valor X: ", x, "\t\tValor del Error: ", err, "\t\tIteraccion: ", contador, "\n")  
 }  
}  
  
Secante(0, 1, 10e-8)

## Valor X: 0.464349 Valor del Error: 0.08524539 Iteraccion: 1   
## Valor X: 0.5626182 Valor del Error: 0.008839985 Iteraccion: 2   
## Valor X: 0.5542804 Valor del Error: 0.0004534648 Iteraccion: 3   
## Valor X: 0.5538245 Valor del Error: 2.495487e-06 Iteraccion: 4   
## Valor X: 0.553827 Valor del Error: 7.02433e-10 Iteraccion: 5

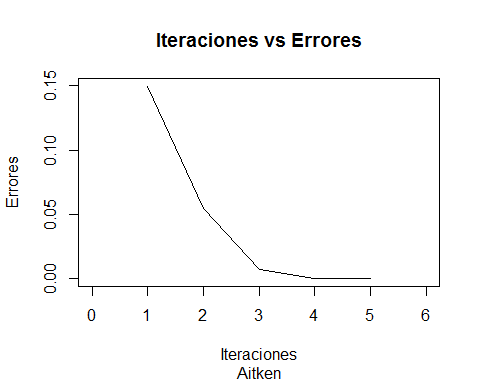
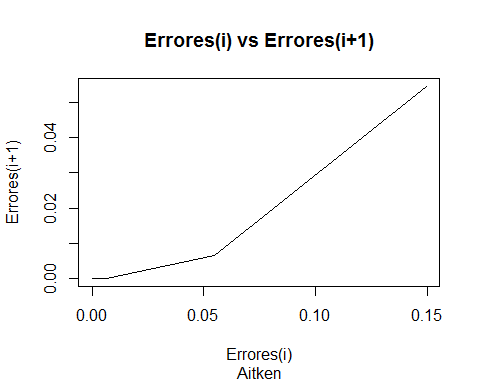
Teorema de Aitken

Método de Aitken

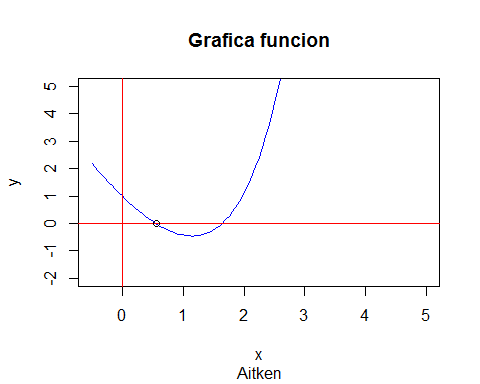
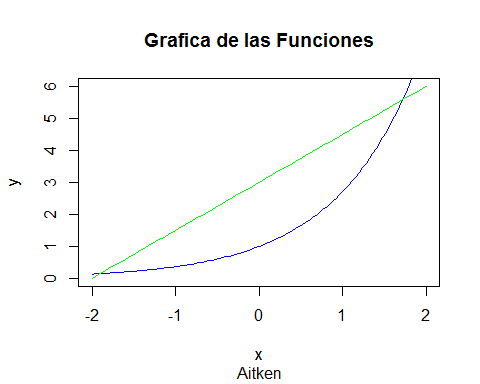
fx<-function(x)  
{  
 signif(exp(1), 5)^x  
}  
  
fx1<-function(x)  
{  
 signif(pi,5)\*x  
}  
  
fx2<-function(x)  
{  
 signif(exp(1), 5)^x-signif(pi,5)\*x  
}  
  
aitken = function(f, m, x0, tol)  
{  
   
 plot(fx, xlim = c(-2,2), ylim = c(0,6), col = "blue", main = "Grafica de las Funciones", sub = "Aitken", xlab = "x", ylab = "y")  
 par(new=TRUE)  
 curve(fx1, type = "l", col="green", axes=FALSE, ylab = "y")  
 par(new=FALSE)  
   
 iteraciones<-c()  
   
 Er1<-c()  
 Er2<-c()  
   
 k<-0  
 E1<-0  
   
 g<-parse(text=f)  
 fx = function(x){eval(g[[1]])}  
 d.<-D(parse(text=f ), "x")  
 df<-function(x) eval(d.)  
   
 plot(fx, xlim = c(-0.5,5), ylim = c(-2,5), col = "blue", main = "Grafica funcion", sub = "Aitken", xlab = "x", ylab = "y")  
 abline(h = 0, v=0, col= "red")  
   
 repeat  
 {  
   
 x1 = x0 - m\*(fx(x0)/df(x0))  
 dx = abs(x1-x0)  
 E2 = E1  
 E1 = dx/x1  
 cat("X=", x1, "\t", "E=", dx, "\t e=", E1,"\t Iteracion", k+1,"\n")  
   
 if(k >= 1)  
 {  
 Er1<-c(Er1, E2)  
 Er2<-c(Er2, E1)  
 }  
   
 k = k + 1  
   
 if (dx < tol) break;  
   
 x0 = x1  
   
   
 }  
   
 points(x1,0)  
   
 plot(fx, xlim = c(0,max(Er1)), ylim = c(0,max(Er2)), col = "white", main = "Errores(i) vs Errores(i+1)", sub = "Aitken", xlab = "Errores(i)", ylab = "Errores(i+1)")  
 lines(Er1, Er2, type = "l")  
   
 Er1<-c(Er1,Er2[k])  
 iteraciones<-c(1:k)  
 plot(fx, xlim = c(0,iteraciones[k]), ylim = c(0,Er1[1]), col = "white", main = "Iteraciones vs Errores", sub = "Aitken", xlab = "Iteraciones", ylab = "Errores")  
 lines(iteraciones, Er1, type = "l")  
}  
  
aitken("2.7182^x-3.1415\*x", 1, 2, 10^-8)



## X= 1.739666 E= 0.2603344 e= 0.1496462 Iteracion 1   
## X= 1.6496 E= 0.09006603 e= 0.05459873 Iteracion 2   
## X= 1.638732 E= 0.01086763 e= 0.00663173 Iteracion 3   
## X= 1.638579 E= 0.0001525962 e= 9.312712e-05 Iteracion 4   
## X= 1.638579 E= 2.987868e-08 e= 1.823451e-08 Iteracion 5   
## X= 1.638579 E= 1.776357e-15 e= 1.084084e-15 Iteracion 6



aitken("2.7182^x-3.1415\*x", 1, 0, 10^-8)



## X= 0.4669558 E= 0.4669558 e= 1 Iteracion 1   
## X= 0.5498345 E= 0.08287861 e= 0.1507338 Iteracion 2   
## X= 0.5538331 E= 0.003998596 e= 0.007219858 Iteracion 3   
## X= 0.553843 E= 9.896336e-06 e= 1.786849e-05 Iteracion 4   
## X= 0.553843 E= 6.078171e-11 e= 1.097454e-10 Iteracion 5

