

Raiz cuadrada

Juan Anagrita, Hector Hernandez, Aldemar Ramirez

July 28, 2019

Problema

Implemente en cualquier lenguaje el siguiente algoritmo que sirve para calcular la raíz cuadrada. Aplíquelo para evaluar la raíz cuadrada de 7, analice su precisión, como podría evaluar la convergencia y validez del algoritmo.

Solución

Lenguaje de programación: R

función principal: $y = 1/2 * (x + n/x)$

Parametros:

n -> Datos

x -> Valor inicial

y -> Respuesta calculada con error E

E -> Error permitido

```
funcion = function(x,n){  
  return ((1/2)*(x+n/x))  
}
```

Función principal para aproximar un numero a su raíz cuadrada

```
raizCuadrada = function(x,n,E){  
  y <- funcion(x,n)  
  resultadosParciales =c()  
  errorAbsoluto<-c()  
  errorRelativo<-c()  
  while(abs(x-y)>E){  
    errorAbsoluto <- c(errorAbsoluto,abs(x-y))  
    errorRelativo <- c(errorRelativo,abs(x-y)/abs(y))  
    resultadosParciales = c(resultadosParciales,y)  
    x <- y  
    y <- funcion(x,n)  
  }  
  errorAbsoluto <- c(errorAbsoluto,abs(x-y))  
  errorRelativo <- c(errorRelativo,abs(x-y)/abs(y))  
  resultadosParciales = c(resultadosParciales,y)  
  lista = list("errorAbsoluto" = errorAbsoluto,"errorRelativo" = errorRelativo,  
              "resultadosParciales" = resultadosParciales, "y" = y)  
}
```

Algoritmo principal para calcular la raíz cuadrada de un número. La función devuelve una lista con todos los resultados parciales, así como sus errores absolutos y relativos a través de cada iteración.

Valores que devuelve:

errorAbsoluto -> vector con los errores absolutos a trav?s de las diferentes iteraciones

errorRelativo -> vector con errores relativos a trav?s de las diferentes iteraciones

resultadosParciales -> vector con los valores que devuelve la funci?n principal

y -> resultadoFinal(rain del numero)

Datos iniciales

```
x <- 3
n <- 7
E <- 1e-8
```

Para solucionar el problema dado se sacará la raíz cuadrada de 7 con un valor inicial de 3

Resultados

```
resultados = raizCuadrada(x,n,E)
iteraciones = length(resultados$errorAbsoluto)
resultadoFinal = resultados$resultadosParciales[iteraciones]

cat("Raiz cuadrada de", n, "=", resultadoFinal, "\n")
```

```
## Raiz cuadrada de 7 = 2.645751
```

```
cat("Error < ", 1e-8, "\n")
```

```
## Error < 1e-08
```

```
cat("Error de truncamiento:", abs(7-resultadoFinal^2), "\n")
```

```
## Error de truncamiento: 8.881784e-16
```

```
cat("Estimacion del error de truncamiento ", resultados$errorAbsoluto[iteraciones], "\n")
```

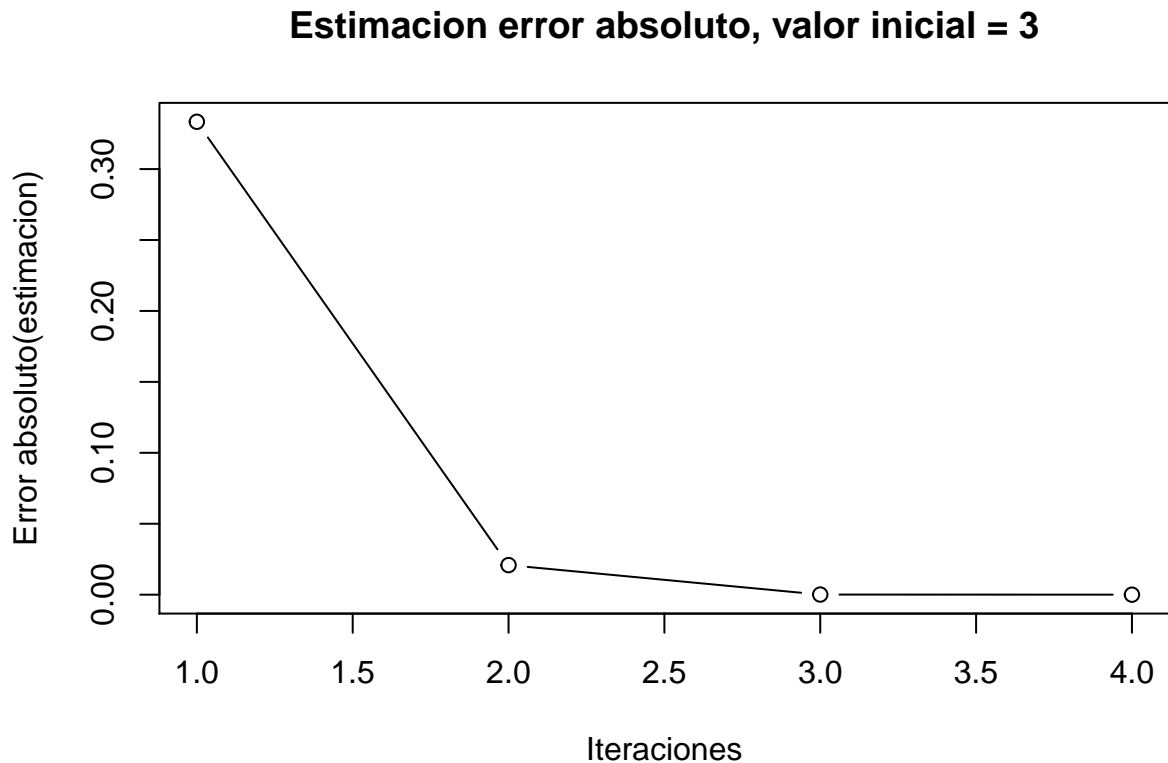
```
## Estimacion del error de truncamiento 1.271367e-09
```

```
cat("Estimacion del error relativo: ", resultados$errorRelativo[iteraciones], "\n")
```

```
## Estimacion del error relativo: 4.805316e-10
```

gráficas

```
plot(resultados$errorAbsoluto, type = "b", xlab="Iteraciones", ylab="Error absoluto(estimacion)",
     main = "Estimacion error absoluto, valor inicial = 3")
```



Grafica de iteraciones vs estimación del error absoluto, para la raíz de siete con valor inicial de 3.

Tabla de errores

```
tablaErrores <- data.frame(
  "iteraciones" = 1:iteraciones,
  "x_i" = resultados$resultadosParciales,
  "errorAbsoluto" = resultados$errorAbsoluto,
  "errorRelativo" = resultados$errorRelativo
)

print(tablaErrores)
```

##	iteraciones	x_i	errorAbsoluto	errorRelativo
## 1	1	2.666667	3.333333e-01	1.250000e-01
## 2	2	2.645833	2.083333e-02	7.874016e-03
## 3	3	2.645751	8.202100e-05	3.100102e-05
## 4	4	2.645751	1.271367e-09	4.805316e-10

Tabla con los resultados parciales, estimación de errores absolutos y relativos, al sacar la raíz de 7 con un valor inicial de 3.

pruebas con otros valores iniciales

tabla de errores

```
results = c()
iterations = c()
absoluteError = c()
relativeError = c()

for(i in 1:10){
  resultados = raizCuadrada(i,n,E)
  results = c(results,resultados$y)
  i = length(resultados$errorAbsoluto)
  iterations = c(iterations,i)
  absoluteError = c(absoluteError,resultados$errorAbsoluto[i])
  relativeError = c(relativeError,resultados$errorRelativo[i])
}

tablaErrores <- data.frame(
  "x_0" = 1:10,
  "raiz" = results,
  "iteraciones" = iterations,
  "errorRelactivo" = relativeError,
  "errorAbsoluto" = absoluteError
)

print(tablaErrores)
```

##	x_0	raiz	iteraciones	errorRelactivo	errorAbsoluto
## 1	1	2.645751	6	1.768064e-11	4.677858e-11
## 2	2	2.645751	5	3.877334e-14	1.025846e-13
## 3	3	2.645751	4	4.805316e-10	1.271367e-09
## 4	4	2.645751	5	1.768064e-11	4.677858e-11
## 5	5	2.645751	6	0.000000e+00	0.000000e+00
## 6	6	2.645751	6	1.386441e-13	3.668177e-13
## 7	7	2.645751	6	1.768064e-11	4.677858e-11
## 8	8	2.645751	6	5.620063e-10	1.486929e-09
## 9	9	2.645751	7	0.000000e+00	0.000000e+00
## 10	10	2.645751	7	1.678499e-15	4.440892e-15

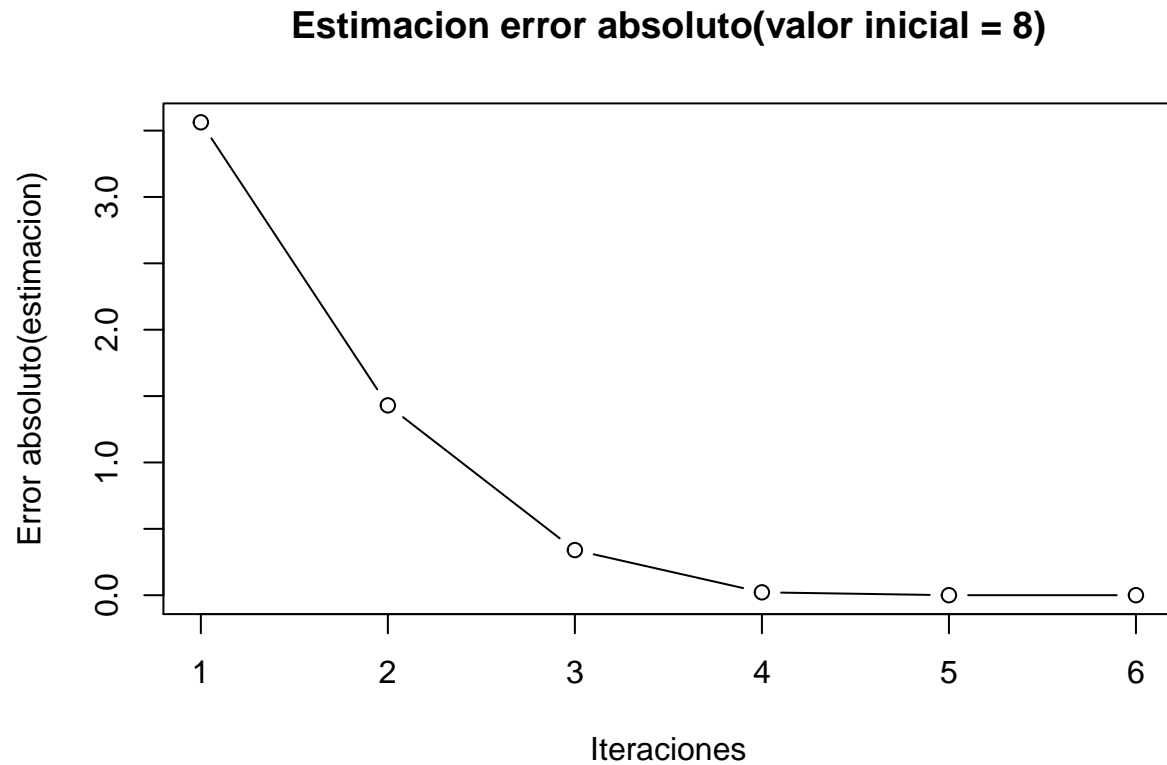
Tabla con los resultados parciales, estimación de errores absolutos y relativos, al sacar la raíz de 7 con un valor inicial que varía entre 1 y 10. Además se muestra el número de iteraciones necesarias para lograr el error absoluto de 1×10^{-8} .

Gráfica de la convergencia del algoritmo para raíz de 7 con valor inicial = 8

Se escogió un valor bastante alejado de la raíz, para que haya más iteraciones y se pueda ver de manera mejor la convergencia del algoritmo

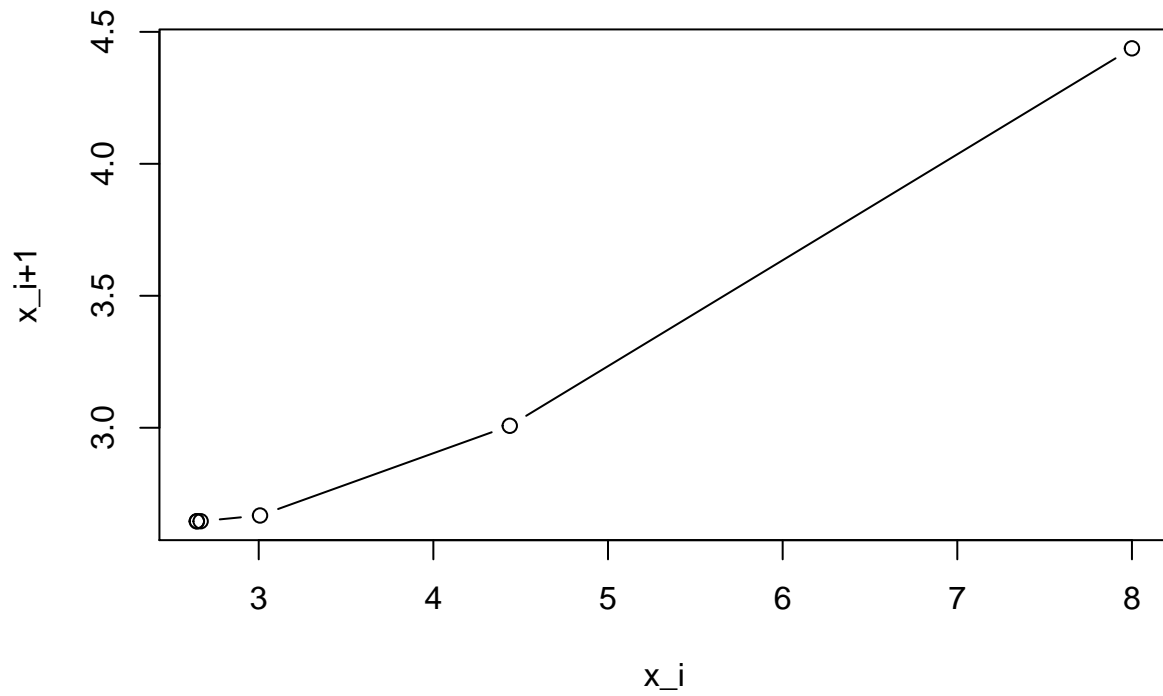
```
resultados = raizCuadrada(8,n,E)
```

```
plot(resultados$errorAbsoluto, type = "b", xlab="Iteraciones", ylab="Error absoluto(estimacion)",
     main = "Estimacion error absoluto(valor inicial = 8)")
```



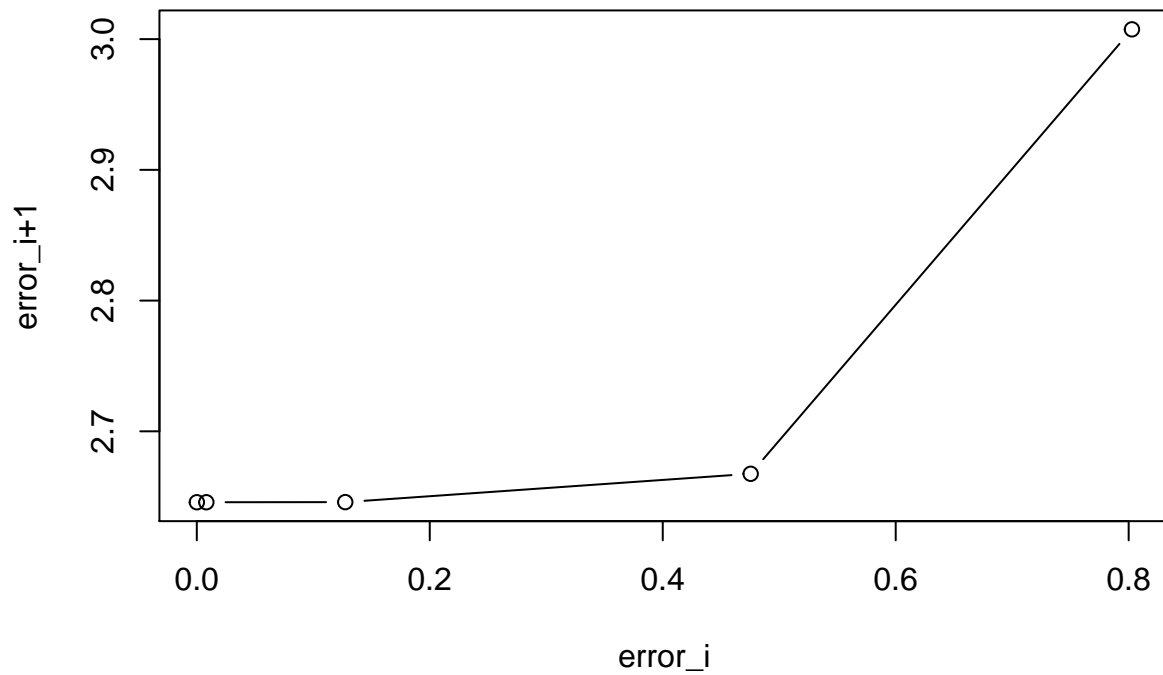
```
iteraciones = length(resultados$resultadosParciales)
xi = (resultados$resultadosParciales)[-iteraciones]
plot(x = c(8,xi), y = resultados$resultadosParciales, type = "b", xlab="x_i", ylab="x_{i+1}",
     main = "Convergencia de la funcion")
```

Convergencia de la funcion



```
plot(x = resultados$errorRelativo[-iteraciones], y = resultados$resultadosParciales[-1], type = "b",  
     xlab="error_i", ylab="error_i+1", main = "Convergencia de la funcion(Error relativo)")
```

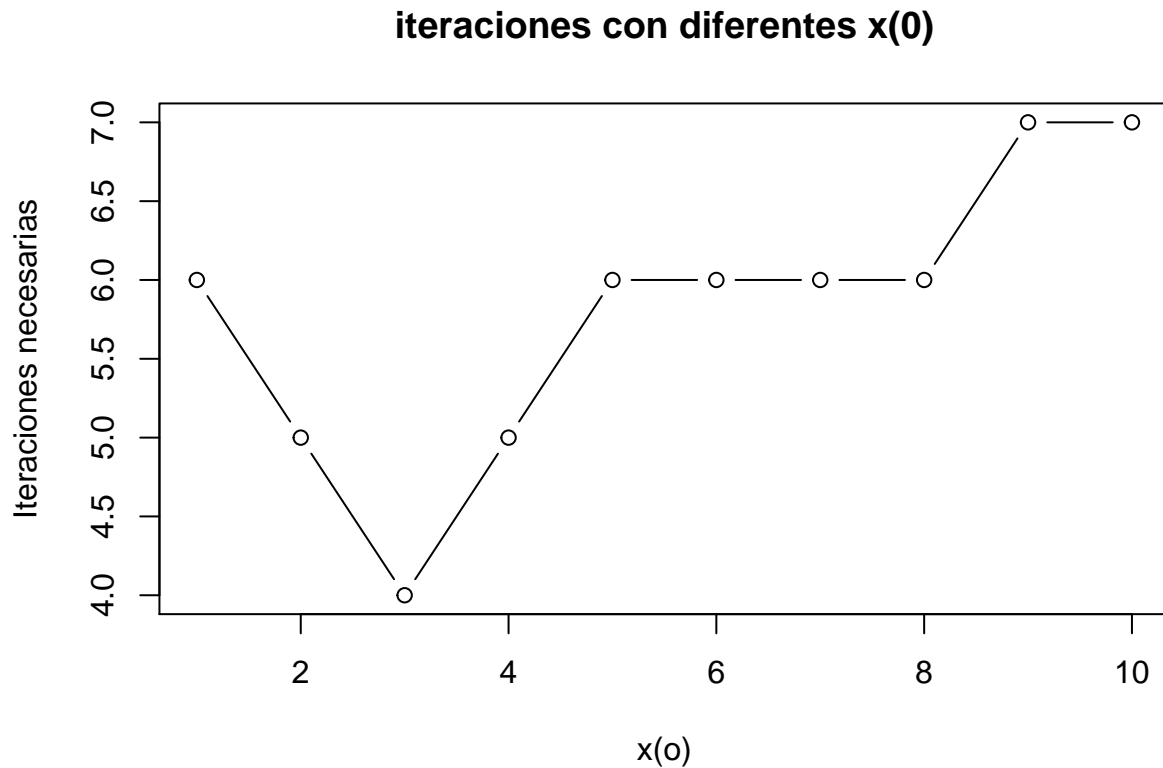
Convergencia de la funcion(Error relativo)



Convergencia: Entre más se aleja el valor inicial al valor de la raíz del número, se producen más iteraciones con lo cual se logra apreciar la convergencia de orden cuadrado que tiene el algoritmo. En todos los casos el algoritmo converge a 0.

gráfica del número de iteraciones necesarias vs valor inicial

```
plot(x = 1:10,y = iterations, type = "b", xlab="x(o)", ylab="Iteraciones necesarias",  
     main = "iteraciones con diferentes x(0)")
```



validez del algoritmo: Para la raíz de 7 con 3 iteraciones el error de truncamiento es de $1.271367 \cdot 10^{-9}$. En todos los casos vistos el algoritmo de acerca de manera apropiada al valor esperado, con la diferencia del número de iteraciones, el cuál crece dependiendo de que tan lejos esté el valor inicial del valor verdadero de la raíz.

```
options(digits = 17)
cat("Raiz cuadrada de", n, "=", resultadoFinal, "\n")
```

```
## Raiz cuadrada de 7 = 2.6457513110645907
```

Precision: El algoritmo tiene presicion hasta la 16 cifra significativa, que es el maximo de cifras significativas que trabaja r. A partir de esta cifra el valor de la raíz varia de manera erronea asi se trabaje con un error de truncamiento mayor.