Práctica 7: Búsqueda local

1445183

19 de marzo de 2019

1. Objetivo

Maximizar la función bidimensional de *g(x,y)* encontrando el valor máximo de *g* por medio del método de "búsqueda local".

2. Descripción

Para encontrar el valor máximo de la función *g(x,y)*, se toma como base el código escrito en la práctica 7 [2], se pide como restricciones que los valores de los ejes *x* y *y* deben estar entre *-3* y *3*; cuando *x,y* tienen una "posición actual"tienen ocho vecinos de los cuales tomarán la posición del vecino que tenga el valor mayor, de esta manera el valor de la función *g* será el mayor conforme avance en pasos hasta el punto que llegue a tener el valor máximo.

```
g <- function(x, y) {
  return(((x + 0.5)^4 - 30 * x^2 - 20 * x + (y + 0.5)^4 - 30 * y^2 - 20 * y)/100)
}

low <- -3
high <- 3
step <- 0.10
puntos <- 15
tmax <- 10^pot
```

Para identificar a los vecinos de *g(x,y)* se le añaden al código variables para ubicar la posición y de esta manera se identifique el vecino con mayor valor:

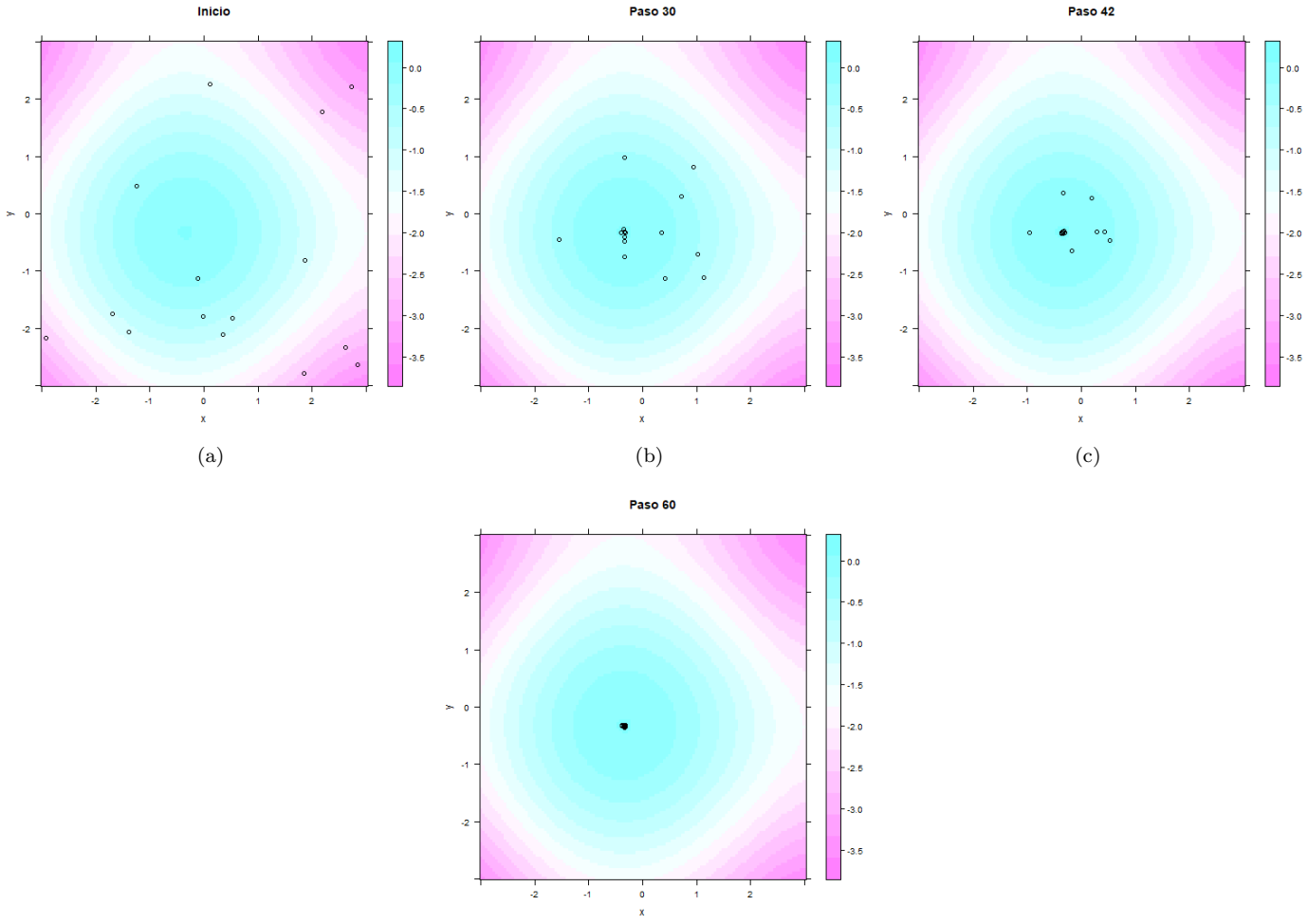
```
posiciones <- data.frame(x = double(), y = double(), bestgxy = double())
for (n in 1:puntos) {
  currx <- runif(1, low, high)
  curry <- runif(1, low, high)
  posiciones <- rbind(posiciones, data.frame(x = currx, y = curry, bestgxy = g(currx, curry)))
}
```

```
for (pot in 1:tmax) {
  resulx <- double()
  resuly <- double()

  for (n in 1:puntos) {
    delta <- runif(1, 0, step)
    deltax <- c(-delta, 0, delta)
    delta <- runif(1, 0, step)
    deltay <- c(-delta, 0, delta)

    #codigo de la practica 4
    vecinosx <- numeric()
    vecinosy <- numeric()
    for (dx in deltax) {
      for (dy in deltay) {
        if (dx != 0 | dy != 0) { # descartar la posicion misma
          vecinosx <- c(vecinosx, dx)
          vecinosy <- c(vecinosy, dy)
        }
      }
    }

    tablax <- rbind(resulx, vecinosx + posiciones$x[n])
    tablay <- rbind(resuly, vecinosy + posiciones$y[n])
  }

  vecinos <- g(resulx, resuly)
  maxvalue <- max.col(vecinos)

  for (i in 1:puntos) {
    posiciones$x[i] <- resulx[i,maxvalue[i]]
    posiciones$y[i] <- resuly[i,maxvalue[i]]
  }
```

Para visualizar mejor la búsqueda se realizan 15 réplicas `puntos`, es decir, 15 diferentes valores de x,y desplazándose hasta llegar a la posición máxima.

3. Resultados

Se puede observar en la figura 1 cómo los puntos (valor de g) tienden a dirigirse hacia la posición máxima al aumentar la cantidad de pasos [1].



(a) (b) (c) (d)

Figura 1: Réplicas simultáneas de la búsqueda

4. Conclusiones

Una función puede ser optimizada por medio del método de búsqueda local, la cual puede ser un poco parecida al método Monte-Carlo, ya que se observó que a mayor cantidad de pasos la precisión aumenta llegando a un solo valor, aunque en este caso es el valor máximo, es decir, no se puede obtener uno mayor a ese. El valor máximo se encontró dentro de los primeros 60 pasos auqnue se esperaba obtener una cantidad de pasos mucho mayor. Si hay algún error probablemente sea en los vectores designados para encontrar a los vecinos.

Referencias

[1] Dulce Carrasco. Práctica 7: Búsqueda local, 2019. URL `https://media.giphy.com/media/3Hz3I7TsfkC32k0mEc/giphy.gif`.

[2] Elisa Schaeffer. Práctica 7: Búsqueda local, 2019. URL `https://elisa.dyndns-web.com/teaching/comp/par/p7.html`.