Búsqueda Local

Abraham Azael Morales Juárez 1422745

19 de marzo de 2019

1. Introducción

Se generarán funciones capaces de detectar máximos locales. Primeramente, se realizará una función en una dimensión, donde se minimiza la función f(x). Será necesario realizar una función bidimencional g(x, y). Está nueva función debe poder identificar los máximos locales. Es posible implementar métodos eficientes [2] para identificar los valores determinados por las máximas locales de una función[1].

2. Objetivos

Realizar combinaciones que proporcionan ocho posiciones posibles vecinos.
Graficar en tres dimensiones.
Crear una visualización animada de cómo proceden 15 réplicas simultáneas de la búsqueda encima de una gráfica de proyección plana.

3. Resultados

Se modificó el código final que fue proporcionado (1) para agregar la función g(x,y).

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

Además se modificó el código para la generación de vecinos, donde se colocó la opción para poder generar 8 distintas posiciones que pudiese elegir para posicionarse. Se coloca el fragmento del código a continuación.

```
replica <- function(t) {
  curr <- runif(2, low, high)
  best <- curr
  for (tiempo in 1:t) {
    deltax <- runif(1,0,step)
    deltay <- runif(1,0,step)

    up <- best + deltay
    down <- best - deltay
    right <- best + deltax
    left <- best - deltax

    upright <- best + c(deltax,deltay)
    upleft <- best + c(-deltax,deltay)

    downright <- best + c(deltax,-deltay)
    downleft <- best +c(-deltax,-deltay)
```

Los resultados obtenidos se pueden observar en la Figura 1 y 2, donde una línea verde representa el valor máximo real del experimento, y se puede observar como la línea va cambiando según los pasos que se estén dando. Hubo un inconveniente, no se pudo hacer que el código marcara los puntos del análisis en rojo.

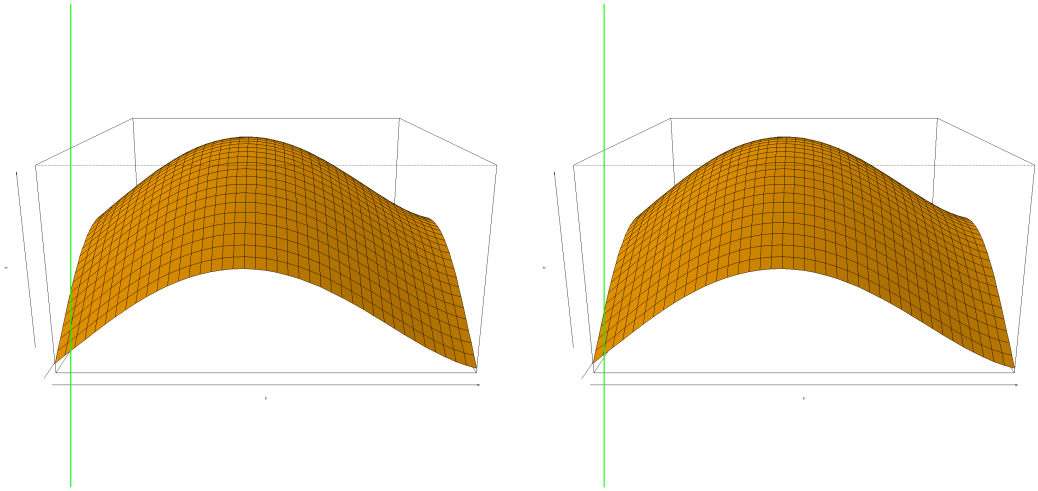

Figura 1: Lado izquierda representa el análisis con 100 pasos y 1000 pasos el de la derecha

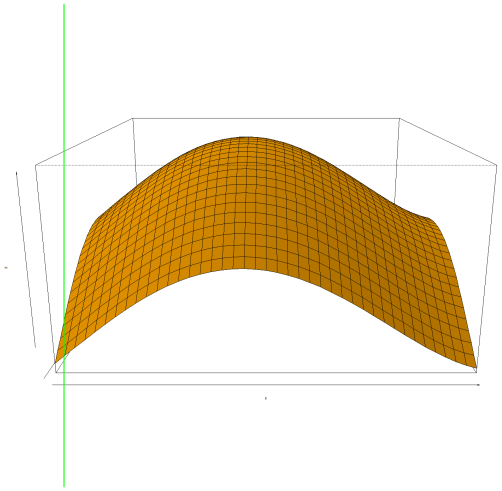

Figura 2: Análsis de 10000 pasos

Las gráficas se obtuvieron con el siguiente código.

```
suppressMessages(library(doParallel))
registerDoParallel(makeCluster(detectCores() - 1))

for (pot in 2:4) {
  tmax <- 10^pot
  resultados <- foreach(i = 1:replicas, .combine=c) %dopar% replica(tmax)
  png(paste("grafica_3D", tmax, ".png", sep=""), width=1500, height=1500)
  x <- seq(low, high, step)
  y <- x
  z <- outer(x, y, g)
  persp(x,y,z, col='orange',expand = 0.5, shade=0.2)
  valores <- f(resultados)
  points(resultados, valores, pch=1600, col="red")
  mejor <- which.max(valores)
  abline(v =resultados[mejor], col="green", lwd=3)
  graphics.off()
   }
stopImplicitCluster()
```

No se pudo realizar el mapa de calor (heatmap), el código es el siguiente:

```
suppressMessages(library(doParallel))
registerDoParallel(makeCluster(detectCores() - 1))

for(graf in 2:4) {
  tmax <- graf
  resultados <- foreach(i = 1:replicas, .combine=c) %dopar% replica(tmax)
  valores <- f(resultados)
  mejor <- which.max(valores)
  x <- seq(low, high, step)
  y <- x
  z <- outer(x, y, g)
  dimnames(z) <- list(x, y)
  d <- melt(z)
  names(d) <- c("x", "y", "z")
  points(resultados, valores, pch=16, col="red")
  png("das.png", width=900, height=900)
  levelplot(z ~ x * y, data = d, main = "")
  graphics.off()
}

stopImplicitCluster()
```

4. Conclusiones

NA

Referencias

[1] Alexia Ibarra. Práctica 7: Búsqueda local. *Universidad Autónoma de Nuevo León*, 2013.

[2] Satu Elisa Schaeffer. Búsqueda local, 2019. URL `https://elisa.dyndns-web.com/teaching/comp/par/p6.html`.