Práctica 9: interacciones entre partículas

10 de octubre de 2017

1. Introducción

Tenemos un grupo de partículas que habitan en un mismo espacio, lo que se ejemplifica en esta práctica, sería mediante la interacción que poseen una con otra, la distancia recorrida a lo largo del tiempo y su desplazamiento. Toda la primera sección de la práctica se hará referencia tanto la la tarea como al reto uno, los cuales consisten en asignarle un cierto peso a cada una de las partículas, así como observar el cambio de velocidad con la que se mueve y delimitar la relación que posee su masa con la velocidad. Par el reto uno lo que se desea es que la representación gráfica de las partículas dependa ahora de los tamaños de la masa, es decir que posean un radio de dimensión proporcional al de su peso. La partícula de inicio posee una coordenada en cada eje `x` y `y`, así como una carga, como lo es en la vida ordinaria, las cargas iguales se repelen y las opuestas se atraen, así mismo se les fue asignado un peso.

2. Parámetros de trabajo

Las pruebas se corrieron en una iMac con procesador 3.1 Ghz, Intel Core i7 con 16 GB de memoria y 1600 Mhz y con ocho núcleos. Se contemplarán los requerimientos del reto uno, así la variable n representa la cantidad total de partículas, de esta forma existirán 50. La prueba fue corrida con panorama de tiempo de cien movimientos, es decir las partículas se desplazaron y actualizaron su posición cien veces.

3. Modificaciones del código

Para poder rescatar los cambios de posición y medir la distancia recorrida para la tarea original fue necesario modificar la forma en la que actualizaba las variables de posición. Así mismo se incluyó una columna más al *data.frame* original, en el cual fueran asignados los pesos de la partícula en proporción a su carga original.

```
c=rnorm(n)
m <- (floor(abs(c)*50) + 1)
p <- data.frame(x = rnorm(n), y=rnorm(n),c, m)
tiemx <- c()
tiemy <- c()
```

Así mismo se crearon los vectores `tiemx` y `tiemy` con el fin de utilizarlos más adelante para salvar las posiciones de las partículas, los cuales son empleados en la paralelización como se muestra. De igual forma para calcular las distancias de los movimientos se incluye la variable distancia, la cual por el modo en el que están guardados los datos te ayuda para el cálculo. Se llegó a la conclusión que no era necesario hacer el ciclo de distancia como una función propiamente ni paralelizarlo, ya que el programa tardaba más tiempo en ejecutarse. Así mismo se muestra la forma que afecta al `delta` el peso de la partícula.

```
for (iter in 1:tmax) {
...
 p$x <- foreach(i = 1:n, .combine=c) %dopar% max(min(p[i,]$x +
 (delta/p[i,]$m) * f[c(TRUE, FALSE)][i], 1), 0)
 tiemx <- c(tiemx,p$x)
```

```
 p$y <- foreach(i = 1:n, .combine=c) %dopar% max(min(p[i,]$y +
 (delta/p[i,]$m) * f[c(FALSE, TRUE)][i], 1), 0)
 tiemy <- c(tiemy,p$y)
...
}
distancia <- c()
for(i in 1:(n*(tmax))){
distancia <- c(distancia, ((tiemx[i]-tiemx[i+50])^2 +
(tiemy[i]-tiemy[i+50])^2)^(1/2))
}
```

Para generar el gráfico adecuado de los datos por tamaño de partícula como lo pide el reto uno fue necesario crear otro `data.frame` con el fin de generar una gráfica tipo `boxplot` que representara los tiempos respecto a la masa, como se muestra en la figura 2 así como grabar los datos en un archivo tipo `.csv` con el fin de no perder los datos y poder usarlos en futuro.

```
total <- data.frame(iteraciones,distancia,masa)
total$masa <- as.factor(total$masa)
library('ggplot2')
png(paste("totalR.png", sep=""), width=700, height=700)
ggplot(data=total,aes(x=masa,y=distancia,fill=iteraciones))+
geom_boxplot()#stat_summary(fun.y=mean,geom="smooth",
aes(group=Tipo,col=Tipo))
graphics.off()

write.csv(total,file="TotalReto.csv")
```

La última modificación que se realizó fue la generación de los grafos respetando el tamaño de la partícula es decir respecto a la masa asignada, a continuación solo se muestra la instrucción para generarla de una sola posición de partículas, así como la figura 1 muestra los diferentes radios de las partículas y de forma similar son las instrucciones para el resto de las imágenes.

```
  png("p9radios.png")
  grafica <- ggplot(p, aes(x=p$x, y=p$y))
  grafica+geom_point(aes(size= p$m,col=colores[p$g+6]))+
  xlab("x")+ ylab("y") +
  labs(color= "carga", size="masa")+
  scale_color_manual(labels=seq(5,-5,-1),values=colores)+
  guides(col= guide_legend(override.aes = list(size=3, stroke=1.5))) +
  scale_size_continuous(breaks=seq(0,0.1,0.01),labels=seq(0,0.1,0.01))
  graphics.off()
```

4. Resultados

La figura 2 nos muestra la práctica original y es un poco de esperar la relación que muestran las distancias respecto a las cargas, mientras es más fuerte la carga que posee la partícula es más el desplazamiento que se realiza en ella.

En cambio observemos la figura 3, en ella podemos observar de igual forma el tiempo pero ahora respecto a la cantidad de masa, la cual como ya mencionamos esta relacionada de cierta forma a la cantidad de carga proporcionada a ella. En la imagen podemos observar que mientras mayor es la masa del objeto, menor es su movimiento, en las partículas pequeñas los movimientos son sumamente grandes y conformo aumenta la masa se puede observar que los movimientos son mucho más pequeños.

5. Comentarios

Se incluyen tres GIF, el primero de ellos, masaIgual muestra el movimiento original de las partículas sin ninguna masa que los haga ir más rápido o menos. El segundo GIF que se incluye

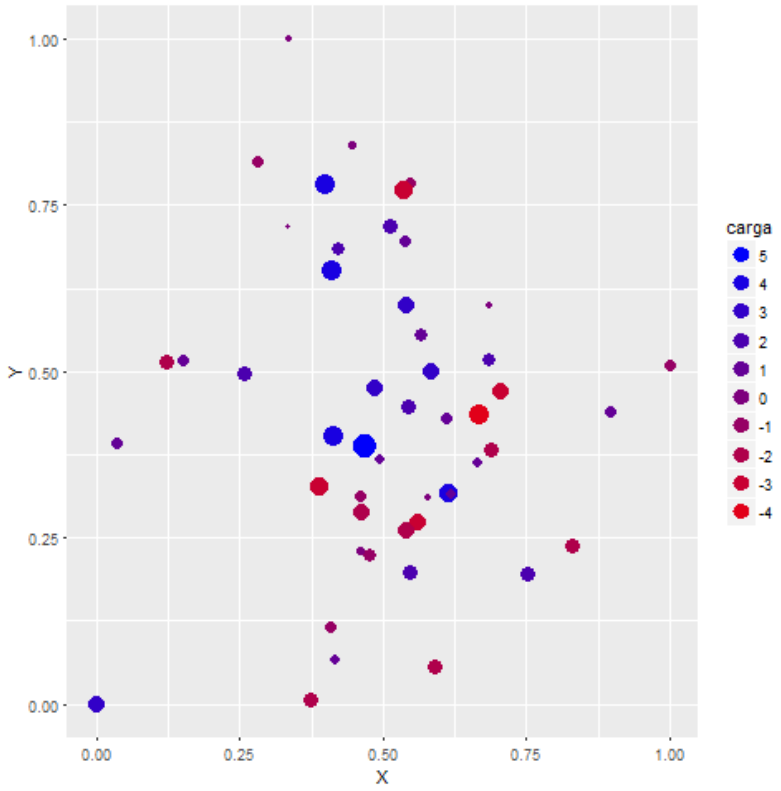

Figura 1: Partículas con masa y radio.

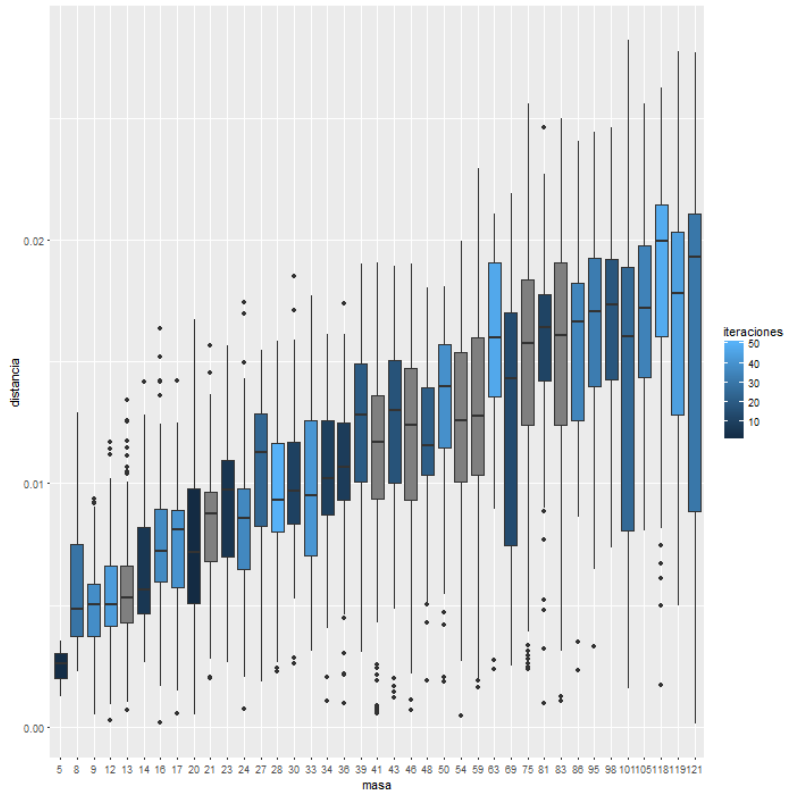

Figura 2: Distancias recorridas con respecto a la carga considerando que todas las masas son iguales.

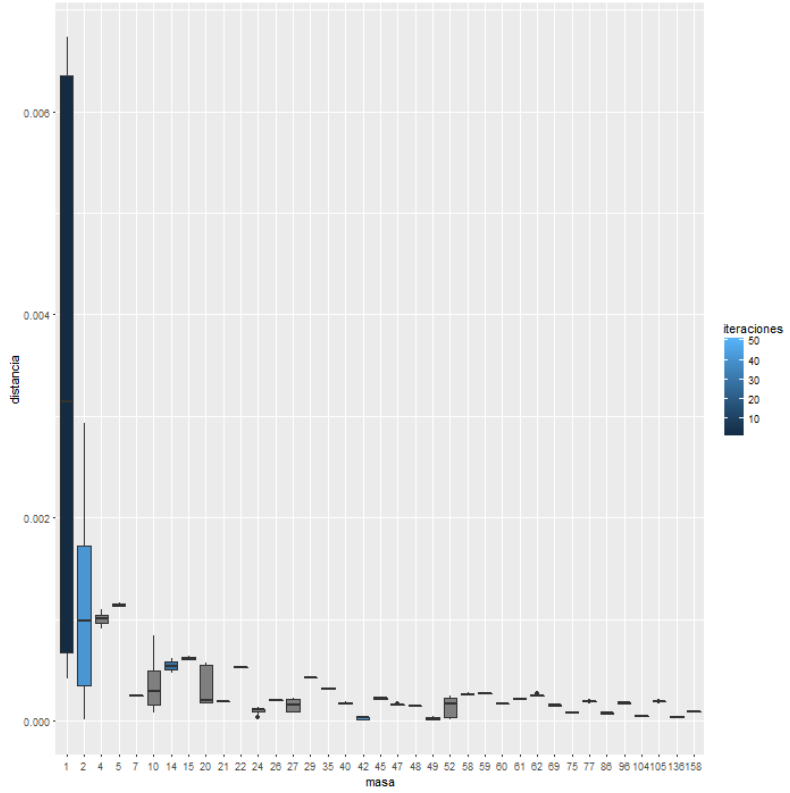

Figura 3: Distancias recorridas respecto a su cantidad de masa.

peso se muestran las partículas con sus respectivos pesos pero aún sin visualización de su radio. Y el último GIF radio las muestra ya con su respectivo radio, es decir en proporción al tamaño de su masa y su mismo movimiento.