"P4" Diagramas de Voronoi

NESTOR

Marzo 2022

1. Objetivo

El objetivo de la práctica consiste en un espacio bidimensional una zona con medidas conocidas que contiene puntos semilla , representados por sus coordenadas. Lo que se busca es dividir esa zona en regiones llamadas celdas de Voronoi de tal forma que todos los puntos que pertenecen a la región que estén más cerca de esa semilla que a cualquier otra.

2. Desarrollo

Basando el desarrollo en la codificación implementado por E. Schaeffer [2] y se encuentra todas las instrucciones en el repositorio repositorio de Nestor en GitHub.

Para comenzar se hace primero generar el cuadro de las semillas con las funciones.

Código 1: Función con las semillas

```
def celda(pos):
   if pos in semillas:
   return semillas.index(pos)
   x, y = pos % n, pos // n
   cercano = None
   menor = n * sqrt(2)
   for i in range(k):
   (xs, ys) = semillas[i]
   dx, dy = x - xs, y - ys
   dist = sqrt(dx**2 + dy**2)
   if dist < menor:
   cercano, menor = i, dist
return cercano
```

Posteriormente al tener las celdas se crea generar el cuadro con las funciones. Lo siguiente es definir en una función para las semillas y las grietas implementados por E. Schaeffer, como se muestra a continuación en el código 2.

Código 2: Semillas para encontrar grietas

```
def inicio():
    direccion = randint(0, 3)
    if direccion == 0: # vertical abajo -> arriba
        return (0, randint(0, n - 1))
    elif direccion == 1: # izq. -> der
        return (randint(0, n - 1), 0)
    elif direccion == 2: # der. -> izq.
        return (randint(0, n - 1), n - 1)
    else:
        return (n - 1, randint(0, n - 1))


def propaga(replica,rupturas):
    grieta = voronoi.copy()
    for f in range(rupturas):
        prob, dificil = 0.9, 0.8
        #grieta = voronoi.copy()
        g = grieta.load()
        (x, y) = inicio()
```

```
        largo = 0
        cnt=0
        if f == 0:
            color=(0,0,0)# grieta color negro
        if f == 1:
            color = (0, 0, 255)# color azul de grieta para el segundo ciclo
        while True:
            g[x, y] = color
            largo += 1
            frontera, interior = [], []
```

Para saber el programa de como se genera el cuadro que se llama las funciones y se pueda generar, grietas implementados por E. Schaeffer, se muestra a continuación en el código 3.

Código 3: Ejecución de códigos

```
for k in 5, 80, 250:
   ciclos=ciclos+1
   print("########## semillas",k,"###############")
   for s in range(k):
    while True:
    x, y = randint(0, n - 1), randint(0, n - 1)
    if (x, y) not in semillas:
    semillas.append((x, y))
    break

    celdas = [celda(i) for i in range(n * n)]
    voronoi = Image.new('RGB', (n, n))
    vor = voronoi.load()
    c = sns.color_palette("Set3", k).as_hex()
    for i in range(n * n):
    vor[i % n, i // n] = ImageColor.getrgb(c[celdas.pop(0)])
    limite, vecinos = 1, []
```

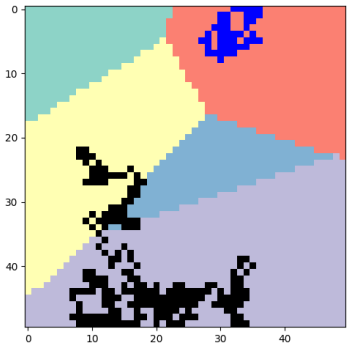
Se realiza para saber las réplicas y en la función de la grieta, se agrega aparte de que haga los ciclos de cincuenta veces, se agrega variables que se conoce como: rupturas. a continuación se muestra el siguiente código.

Código 4: Rupturas

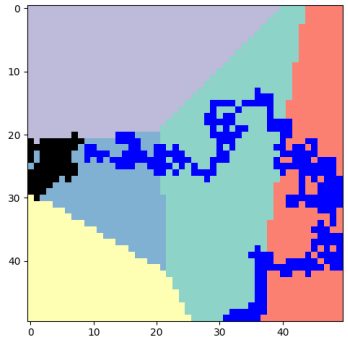
```
rupturas=2# cantidad de grietas
        contacto=[]
        for r in range(300): # replicas
            atraveso = propaga(r,rupturas)
            #print("Por replica:",r, "resultado:",atraveso)
        print("cuantos atravesaron",len(atraveso))
        if ciclos == 1:
            prob["min_sem"].append(((len(atraveso))/300)*100)
        if ciclos == 2:
            prob["med_sem"].append(((len(atraveso))/300)*100)
        if ciclos == 3:
            prob["max_sem"].append(((len(atraveso))/300)*100)
    print("final de probabilidades",prob)
```

3. Resultados

A continuación se muestra las semillas y las grietas, en la figura 1. Donde se da a demostrar que las grietas no se tocan figura 1a, y otro donde las grietas se tocan figura 1b. En la figura 2 se muestra las gietas de las semillas de mayor valor 2a y 2b. Posteriormente, en la figura 3 se muestran el porcentaje de las probabilidades que se calcularon de las grietas de las semillas con las dimensiones. Y finalmente una analisis de los datos multivariados en la figura 4.

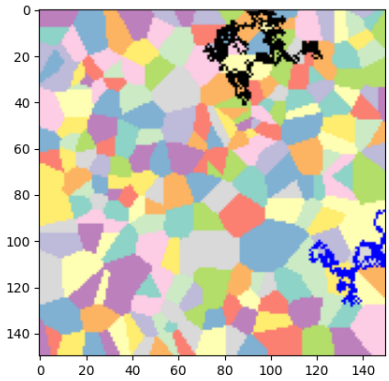


(a) Las grietas no se tocan.

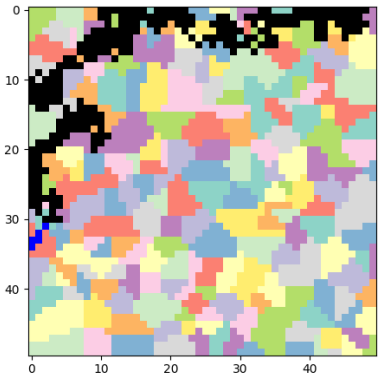


(b) Las grietas se tocan.

Figura 1: Grietas.



(a) Las grietas no se tocan.



(b) Las grietas se tocan.

Figura 2: Semillas de grietas de mayor valor.

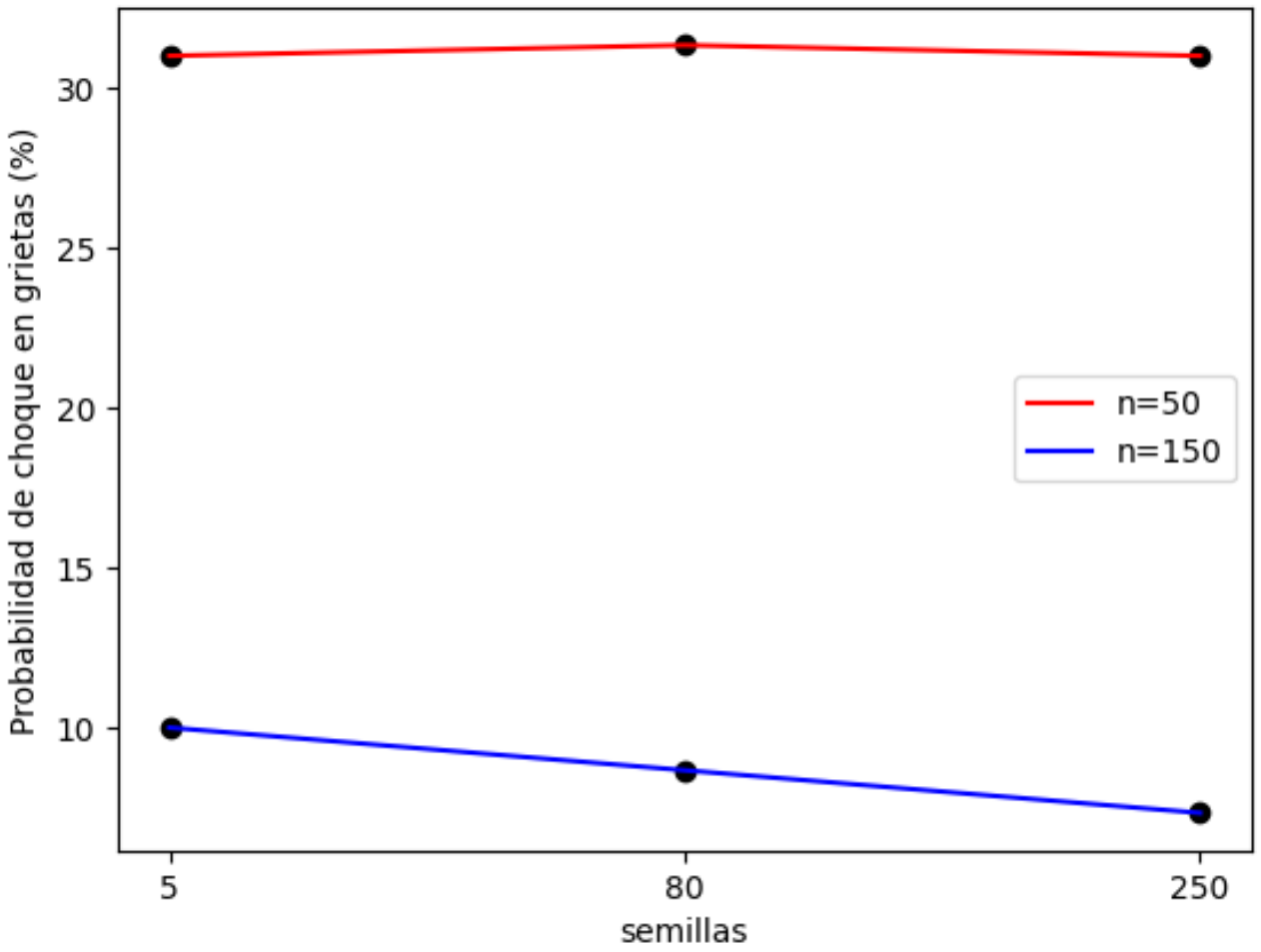


Figura 3: Probabilidad de las grietas

3.1. Radar Chart

Una Gráfica de Radar, también conocida como un diagrama de araña, es una herramienta muy útil para mostrar visualmente los gaps entre el estado actual y el estado ideal. [1]. A continuación se presenta visualmente los gaps existentes entre el estado actual y el estado ideal.

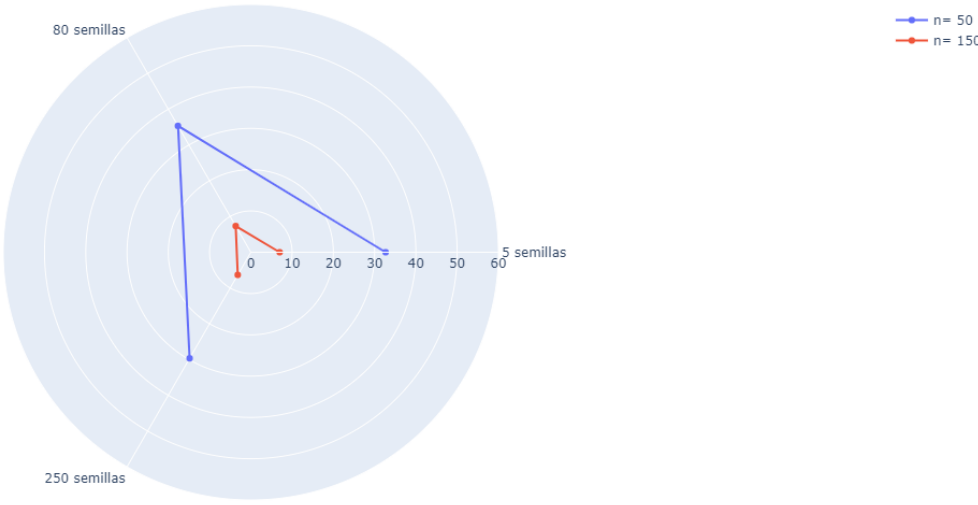


Figura 4: Analisis de los datos multivariados

4. Conclusiones

Se concluye que las fronteras va aumentando con respecto a la cantidad de semillas, ya que las grietas no se aminoran. Sin embargo en la gráfica de radar chart nos muestra como se comporta las semillas dando valores multivariados.

5. Reto 1

En este reto consiste en crecer las celdas dinámicamente alrededor de semillas de tal forma que las semillas aparecen al azar en distintas iteraciones y crecen con una tasa exponencialmente distribuida (variable entre núcleos pero constante para un núcleo específico) hasta toparse con las demás celdas, así como se muestra en la animación. Examina los cambios producidos en el fenómeno de propagación de grietas que esta nueva forma de crear las celdas provoca, ya que las semillas resultan en celdas de tamaños distintos según su edad y su tasa, además del efecto de la posición relativa a las demas semillas [2].

Para esto se crea una función para la visualización y creación de las semillas de como van creciendo ya que la práctica es lo mismo, este reto consiste si las grietas chocan solo que se modifican la creación de las semillas como se muestra en las siguientes instrucciones:

Código 5: Función con las semillas

```
def union(seeds,im,incr):
  for a in range(len(seeds)):
    x=seeds[a][0]
    y=seeds[a][1]
    r = incr[a]
    color=im.getpixel((x, y))
    img=im.copy()
    image = img
    draw = ImageDraw.Draw(image)
    draw.ellipse((x-r, y-r, x+r, y+r), fill=color)
    for Y in range(n):
```

Luego creamos la posición y color de las semillas como se mjuestra en la siguiente instrucción:

Código 6: Posición y Color de las semillas

```
for s in range(k):
            while True:
                x, y = randint(0, n - 1), randint(0, n - 1)
                if (x, y) not in semillas:
                    semillas.append((x, y))
                    break
        col=[]
        for dato in paleta:
            col.append([(int(i * 255)) for i in dato])
```

Para saber el comportamiento de las semmillas y el crecimiento, se muestra en la siguiente instrucción:

Código 7: Comportamiento y Crecimiento de las semillas

```
for i in range(len(semillas)):
  voronoi.putpixel(semillas[i],(tuple(col[i])))
    #plt.imshow(voronoi)###### IMPRIMIR
    #plt.show()
    #vo=voronoi.copy()
    #visual = vo.resize((10 * n,10 * n))
    #visual.save("ciclo_0_ini.png")
    p=0.4
    aumento=[]
    semi=[]
    for s in range(n):
    #fondo=[]
    #print("######## ciclo:",s)
    if s == 0:
      semi.append(semillas[0])
      semillas.pop(s)
      aumento.append(0)
```

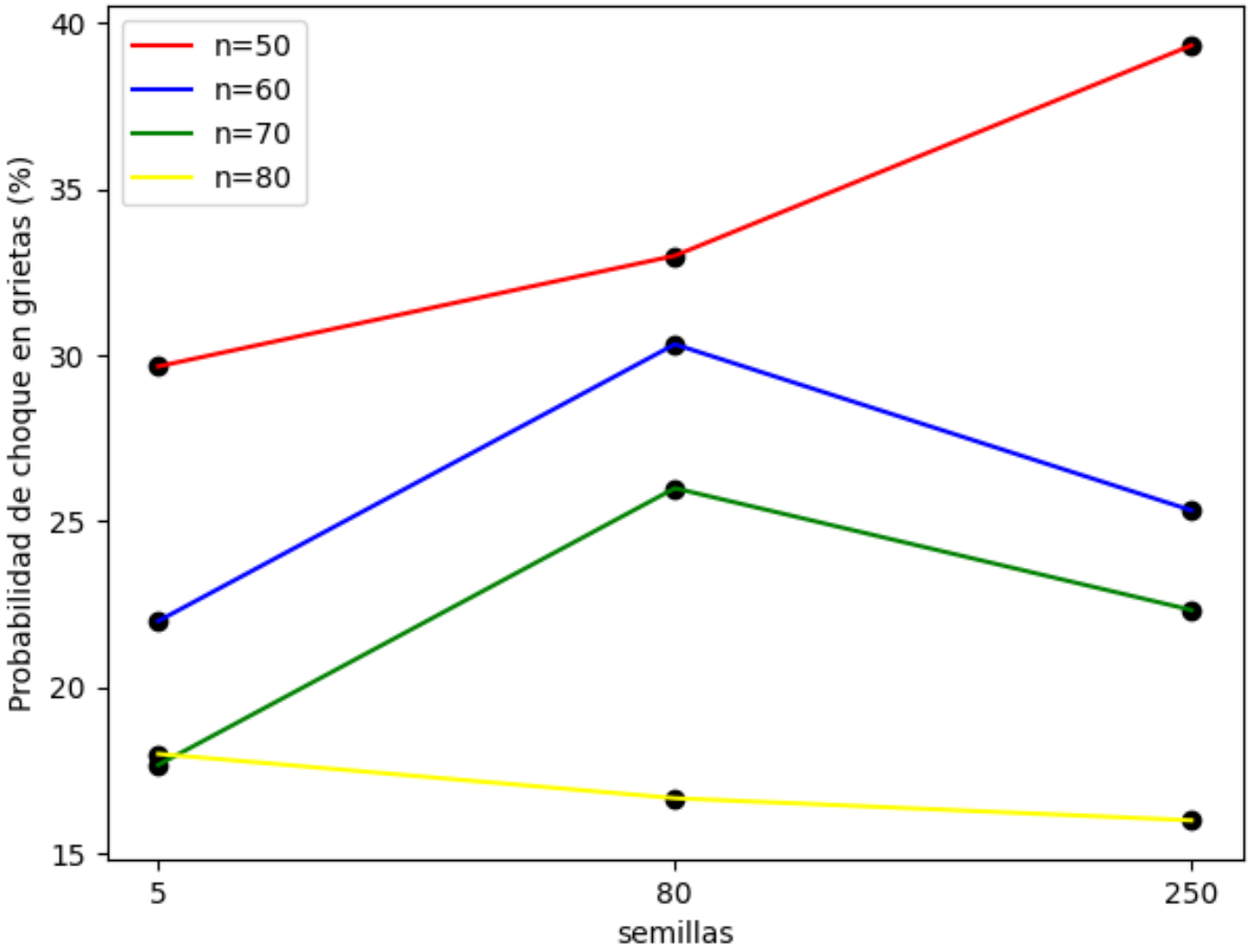
Para obtener los resultados se muestra para las semillas que se crearon, fueron creciendo hasta que se cristalizó con el aumento de más semillas como se muestra a continuación:

Código 8: Obtención del resultado comportamiento del crecimiento de las semillas

```
if s != 0 and ((random.uniform(0, 1)) > p) and len(semillas)>0:
                rnd=random.choice(semillas)
                semi.append(rnd)
                semillas.remove(rnd)
                aumento.append(0)
            aumento=[s+1 for s in aumento]
            voronoi= union(semi,voronoi,aumento)
            #[[fondo.append(voronoi.getpixel((x,y))) for x in range(5)]for y in range(5)]
            #vis=voronoi.copy()
            #visual = vis.resize((10 * n,10 * n))
            #visual.save("ciclo_{:d}.png".format(s))
            #plt.imshow(voronoi)
            #plt.show()
        #plt.imshow(voronoi)####### IMPRIMIR
        #plt.show()################
```

6. Reto 1 resultados

Con el aumento de más semillas se cristalizó y se analizó lo mismo del comportamiento de las grietas. Se encuentra todos los resultados y gifs en el repositorio repositorio de Nestor en GitHub.

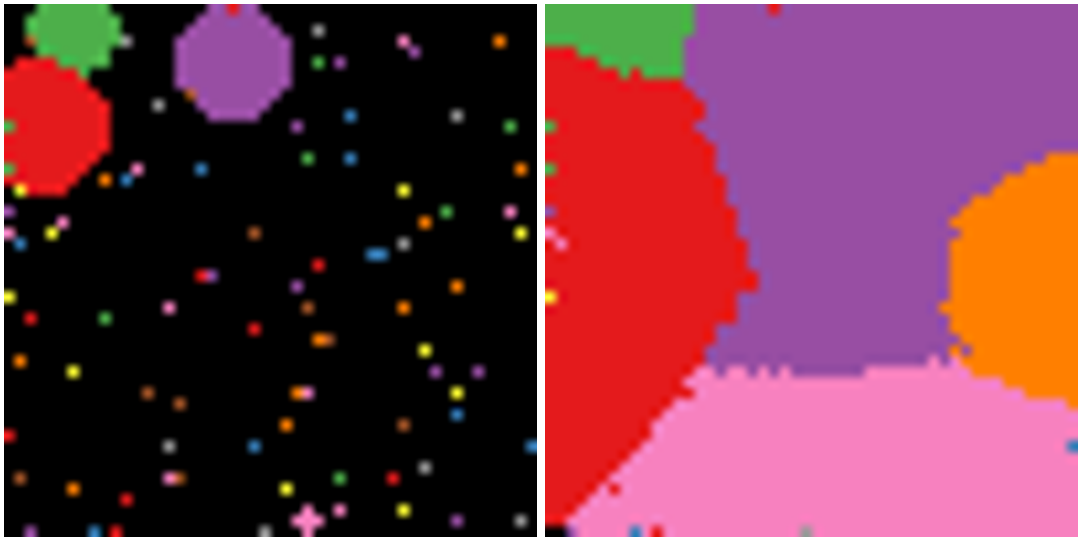

n=50
n=60
n=70
n=80
Probabilidad de choque en grietas (%)
40
35
30
25
20
15
5
80
250
semillas

(a) Las fronteras no se tocan. (b) Las fronteras se tocan.

Figura 3: Intervalo de 10 Semillas de las Grietas

(a) Las fronteras no se tocan. (b) Las fronteras se tocan.

Figura 4: Intervalo de 80 Semillas de las Grietas



(a) Las fronteras no se tocan. (b) Las fronteras se tocan.

Figura 5: Intervalo de 100 Semillas de las Grietas

Referencias

[1] J. Hunter. Lines, bars and markers. *Repositorio, GitHub*, 2021.

[2] E. Schaeffer. Autómata celular. *Repositorio, GitHub*, 2022.

[1]