INSTITUTO TECNOLÓGICO DE BUENOS AIRES
72.25 Simulación de Sistemas

TRABAJO PRÁCTICO Nro. 2

Autómatas Celulares

AUTORES Ail, Brian Ezequiel Leg. N° 49254
Fuster, Marina Leg. N° 57613

BUENOS AIRES
2020

1. INTRODUCCIÓN

Los autómatas celulares son modelos matemáticos que nos permiten encontrar o estudiar patrones complejos. Esto nos puede resultar sumamente interesante ya que, en la naturaleza, es común encontrar sistemas cuyo comportamiento es complejo, pero cuyas unidades constituyentes se comportan de manera simple, siguiendo siempre las mismas reglas.
El autómata celular suele ser representado como una grilla de celdas de lado α que discretiza el espacio de dimensión δ. Cada celda x_i posee un estado $s_t(x_i)$ (los estados son discretos) y luego se definen reglas de transición para precisar cómo el sistema evoluciona de un estado a otro. Las celdas son todas del mismo tipo y las reglas (que suelen ser deterministas) se mantienen uniformes en todo el espacio, a través del tiempo.
Para actualizar el estado de una celda x_i, se debe tomar en cuenta los estados anteriores, es decir, si $x_1, x_2, \ldots, x_n$ son las celdas del espacio, ϕ el conjunto de reglas, y $s_t(x_1), s_t(x_2), \ldots, s_t(x_n)$ los estados de las celdas en el tiempo t

$$s_{t+1}(x_i) = \phi(s_t(x_1), s_t(x_2), \ldots, s_t(x_n)) \quad \textbf{(1)}$$

En la materia fueron presentados autómatas celulares correspondientes a 1D, Game of Life (juego de la vida), FHP para simular fluidos y, finalmente, un autómata que permite simular bandadas de partículas. Nuestro trabajo se realiza a partir del autómata Game of Life.

1.1 Game of Life

Como previamente definimos, este autómata 2D posee estados discretos y un conjunto de reglas que permite la transición entre estados a través del tiempo. El espacio de estados puede ser representado como $S = \{0, 1\}$, siendo 0 una celda muerta y 1 una celda viva. En el caso de Game of Life, la aplicación de las reglas implica definir cuáles son los vecinos de cada celda x_i, ya que el nuevo estado dependerá de ellos. Para todos los sets de reglas definidos y teniendo en cuenta que podemos definir nuestras celdas son de la forma $x_{11}, x_{12}, \ldots, x_{nn}$ utilizamos la fórmula de vecindario de Moore de radio r

$$N_{i,j}^{(M)} = \{(k,l) \in L / |k-i| \leq r \text{ and } |l-j| \leq r\} \quad \textbf{(2)}$$

Como nuestro trabajo requería presentar tres autómatas diferentes en 2D y tres autómatas en 3D, fue necesario extender la ec. (2) para poder utilizarla con celdas de la forma $x_{111}, x_{112}, \ldots, x_{nnn}$, con lo cual la fórmula para vecindad utilizada en 3D es

$$N_{i,j,k}^{(M)} = \{(l,m,n) \in L / |l-i| \leq r \text{ and } |m-j| \leq r \text{ and } |n-k| \leq r\} \quad \textbf{(3)}$$

En el caso de Game of Life y del resto de los sistemas que estudiaremos, el resultado de aplicar el conjunto de reglas ϕ a la celda $x_{i,j}$ en el tiempo t dependerá de los parámetros $a_t(x_{i,j})$ y c_t donde

$$a_t(x_{i,j}) = \left| N_{i,j}^{(M)} \right| \quad \textbf{(4)}$$

para dos dimensiones (fácilmente extensible a tres dimensiones) y, siendo γ la cantidad de celdas vivas en el tiempo t, definimos

$$c_t = \gamma / \alpha^{\delta} \qquad \textbf{(5)}$$

Estos sistemas deben ser inicializados con un porcentaje inicial de celdas vivas ρ, en un espacio acotado al centro y alejado de los bordes. El porcentaje del espacio que representa este centro acotado lo representamos mediante ϱ.

2. IMPLEMENTACIÓN

Para traducir los seis modelos requeridos por el trabajo práctico, dividimos la implementación en dos secciones principales. En primer lugar, la simulación fue programada en Java y, en segundo lugar, el post procesamiento fue implementado a partir de una combinación de Python y Ovito. Utilizamos Python para las transformaciones de archivos, obtención de métricas y realización de gráficos estáticos. Ovito nos fue útil para la realización de animaciones.

2.1. Simulación

La simulación se compone, a grandes rasgos de una `App` que maneja el flujo principal del programa, un `package game`, que contiene el código correspondiente al juego, y un `package config`, que se encarga de guardar y unificar la configuración del sistema al momento de ejecución.

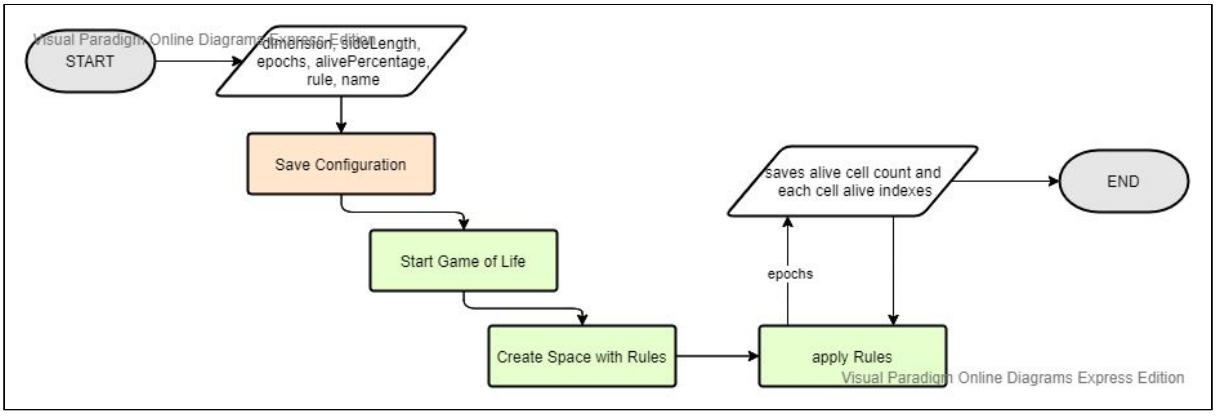

Figura 1: Diagrama de flujo de la simulación

En Fig. 1 podemos ver que la simulación comienza pidiendo determinados inputs al usuario, luego guarda la configuración y se procede a crear un nuevo Game of Life con sus respectivas reglas. Los resultados de dicha simulación se guardan en un archivo de nombre `name` (el cual también es input del usuario).

El producto final de esta primera instancia es un archivo que posee, para cada t de duración de la simulación, la cantidad de celdas vivas y las coordenadas de las mismas. Dependiendo del tipo de simulación que ejecutemos, puede ser uno o varios archivos (lo cual nos resulta útil para gráficos comparativos).

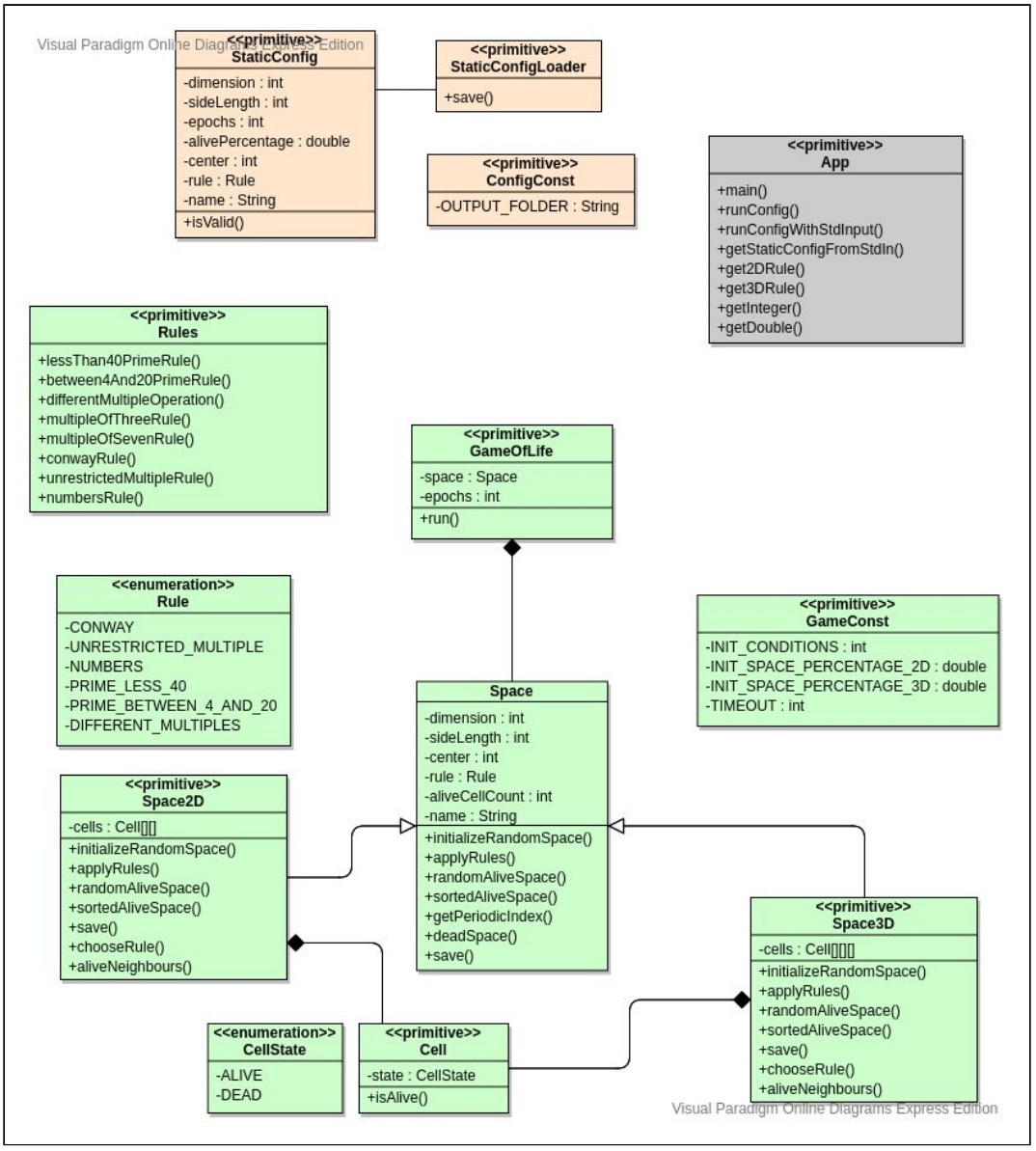

Figura 2: diagrama de clases UML (en naranja `package config`, en verde el `package game` y en gris el flujo general de la aplicación, `App`)

El espacio creado puede ser tanto 2D como 3D y, para permitir al sistema evolucionar en el tiempo, se aplica el siguiente pseudocódigo:

```
(1) desde t=1 hasta epochs:
(2)    C(t) <- nuevo espacio de celdas
(3)    n(t) <- nueva cantidad de celdas vivas
(4)    para cada celda x(t-1) en C(t-1):
(5)         x(t) <- aplicarReglas(x(t-1))
(6)         guardar x(t) en C(t)
(7)         si x(t) está viva: n(t)++
```

Para aplicar las reglas del paso (5) del pseudocódigo se deberá calcular los vecinos de la celda con el método `aliveNeighbours()` correspondiente a cada clase `Space` (pues dependiendo del

espacio, se implementa ec. (2) para 2D o ec. (3) para 3D) y elegir uno de los métodos de la clase `Rules`. Esto puede corroborarse con las clases y métodos que se muestran en Fig. 2.

2.2 Post procesamiento

El post procesamiento tiene dos utilidades principales. En primer lugar, busca transformar los archivos para que su salida pueda ser utilizada en Ovito. A esto le llamaremos `PostProcessor for visualization`. En segundo lugar, podemos obtener las estadísticas necesarias para nuestros resultados. Este segundo proceso lo denominamos `Statistics Extractor`.

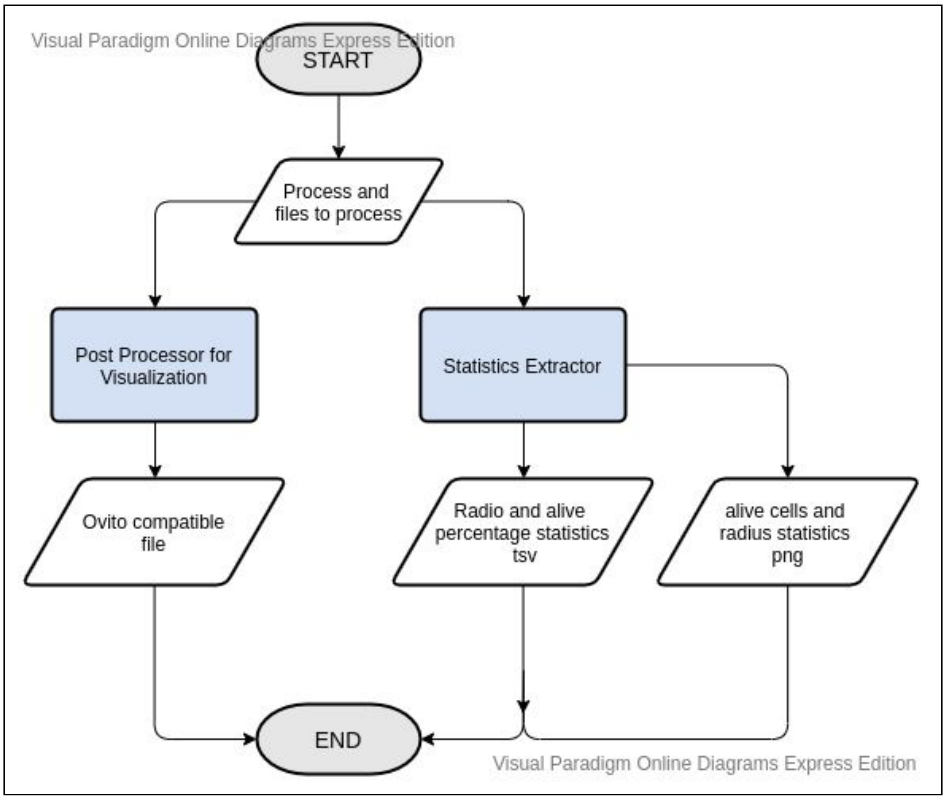

Figura 3: diagrama de flujo del post procesamiento

Como podemos ver en Fig. 3, el usuario puede elegir el proceso a ejecutar y con qué archivo ejecutarlo. Para archivos individuales se utiliza el `PostProcessor for visualization`, ya que permite obtener el archivo correspondiente a una animación para Ovito y, para procesos que buscan sintetizar lo que ocurre en varias simulaciones al mismo tiempo se ejecuta `Statistics Extractor`. Los productos posibles de este proceso son:

1. Archivo .xyz con terminación en "_visualization", para luego configurar en Ovito. Contiene radio de las partículas, ubicación y color correspondiente de acuerdo a la distancia.
2. Archivos .png para mostrar la evolución en el tiempo de porcentaje de celdas vivas y de radio del sistema.
3. Archivos .tsv con las métricas correspondientes a la imagen png descrita en el punto 2.

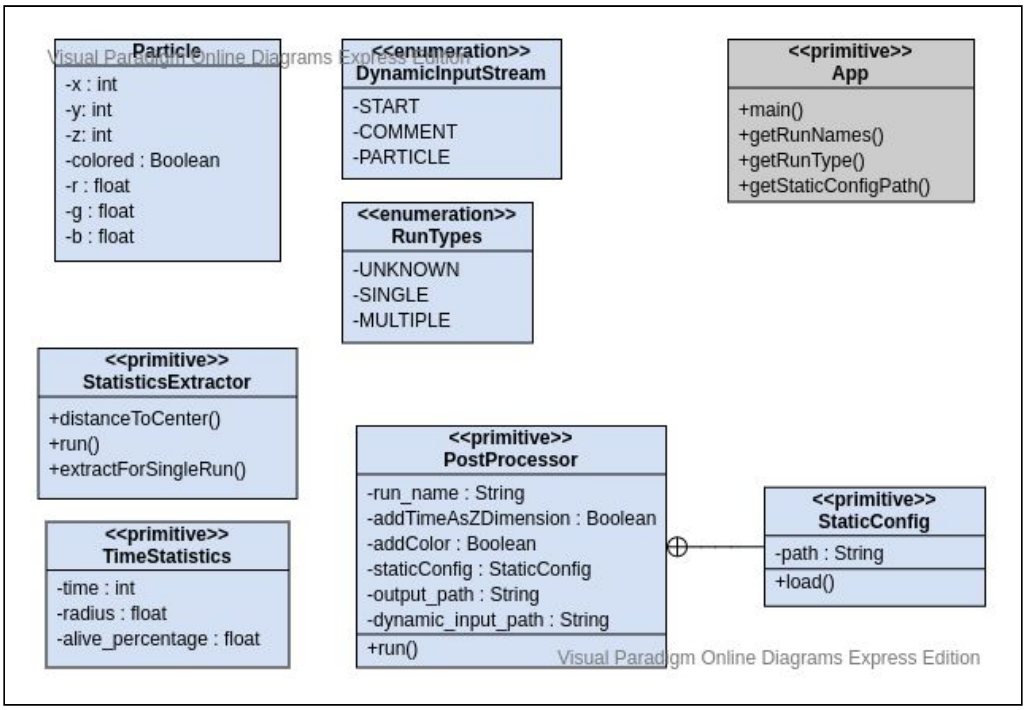

Figura 4: diagrama de clases UML (en azul lo necesario para realizar el procesamiento y en gris el flujo general de la aplicación)

3. SIMULACIONES

Para realizar las simulaciones fue necesario definir los valores de los parámetros que acompañarán las reglas para autómatas celulares 2D y 3D. Esto fue realizado a partir del modelo teórico que se presentó en la sección introductoria.

3.1. Simulaciones 2D

Para todas las simulaciones 2D, $\delta = 2$, $\alpha = 100$, y $\varrho = 0.1$ la cantidad de iteraciones para permitirle al sistema evolucionar es 250 y se realizaron 10 ejecuciones por cada regla, a excepción de Conway Rules, donde el número de ejecuciones fue de 100.

3.1.1 Conway Rules

Este conjunto de reglas es el correspondiente al autómata celular propuesto en 1970 por el matemático británico John Horton Conway. A partir de ec. (4), podemos definir $s_{t+1}(x_i)$ como

$$s_{t+1}(x_i) = s_t(x_i) \lfloor 1/\left|2 - a_t(x_i) + 1\right| \rfloor + \lfloor 1/\left|3 - a_t(x_i) + 1\right| \rfloor \qquad \textbf{(6)}$$

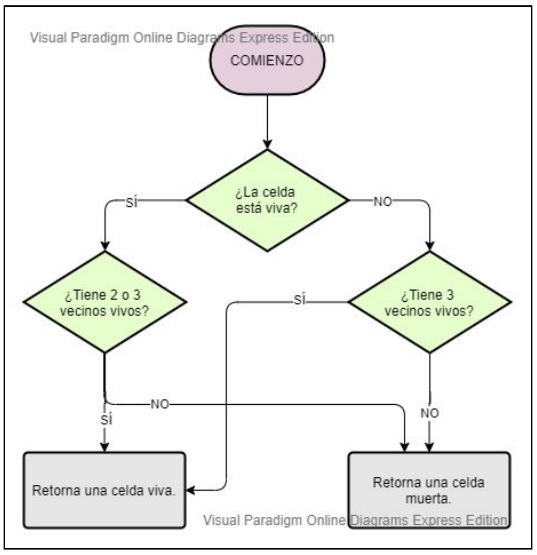

Figura 5: flujo de decisión para aplicar la regla Conway

En Fig. 5 podemos ver el flujo representado por las reglas de Conway correspondientes a ec. (6), de manera tal que sea más sencilla la interpretación de las mismas. Este tipo de esquemas será repetido para todos los conjuntos de reglas, tanto 2D y 3D, cuyo objetivo será, así como en este caso, representar en palabras el significado de la fórmula.

Luego de pasar por el post procesamiento y obtener la evolución del radio de crecimiento y del porcentaje de celdas vivas en función del tiempo, decidimos estudiar, en base a la evolución del radio de crecimiento, la probabilidad de que el sistema se mantenga estacionario a cierto radio o de que el sistema crezca hasta tocar las paredes del espacio. Primero, definimos

$$f_k(x) = \begin{cases} 1 \; si \; max(\, r(\, x)\,) \; \leq \; 50 \\ 0 \; si \; max(\, r(\, x)\,) \; > \; 50 \end{cases} \qquad \textbf{(7)}$$

como la fórmula que nos permite saber si el sistema en la ejecución k se estancó, la cual devuelve 1 si lo hizo, 0 sino. Luego, definimos el observable correspondiente a Conway Rules como

$$O \; = \; (\sum_{i=1}^{100} f_k(x)\,)\,/100 \qquad \textbf{(8)}$$

Este observable nos llevó a preguntarnos sobre el radio de crecimiento al cual convergen aquellas ejecuciones $k \,/\, f_k(x) = 1$. Para ello, nos quedamos con las curvas que veíamos que convergían sin llegar a la pared. Luego, definimos un segundo observable O' definido como

$$O' \; = \; r(250) \qquad \textbf{(9)}$$

donde r es el radio del sistema en $t \; = 250$.

3.1.2 Unrestricted Multiple Rules

En este caso presentamos un nuevo conjunto de reglas, donde no hay restricciones sobre el valor de los vecinos en cuanto a magnitud, sino de acuerdo a su multiplicidad. A partir de ec. (4), definimos la transición de un estado a otro, $s_{t+1}(x_i)$, como

$$s_{t+i}(x_i) = \lceil a_t(x_i)/(a_t(x_i)+1) \rceil (s_t(x_i) \lfloor 1/(a_t(x_i)\%3+1) \rfloor + (1-s_t(x_i)) \lfloor 1/(a_t(x_i)\%2+1) \rfloor) \quad \textbf{(10)}$$

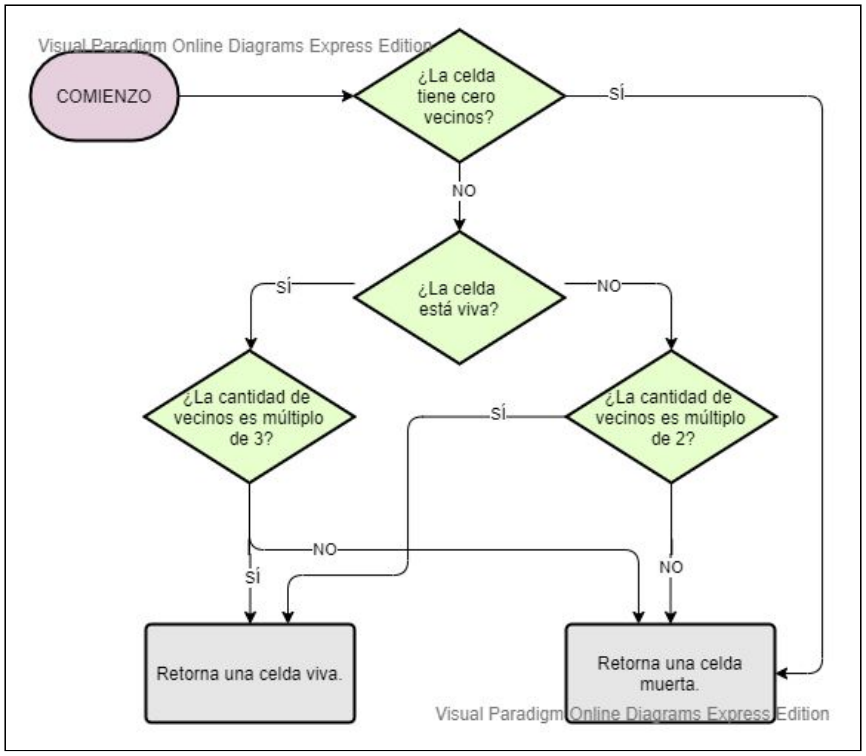

Figura 6: flujo de decisión para aplicar
la regla Unrestricted Multiples

En Fig. 6 vemos las reglas correspondientes a la ec. (10). En este caso, tomaremos como observable la velocidad de aumento del radio de crecimiento, representado según la pendiente de la recta de dicho radio. Sea la función numpy.polyfit cuya fórmula matemática y explicación se encuentra en [2], t el tiempo en que el sistema toca la pared, definimos el observable O según

$$O = polyfit(x,\ r(x),\ 1)\ con\ 0 \leq x \leq t \quad \textbf{(11)}$$

Donde $r(x)$ es el radio del sistema en el tiempo x, y el 1 representa el grado del polinomio al cual queremos ajustar nuestros resultados.

3.1.3 Numbers Rules

Nuestro tercer, y último, conjunto de reglas 2D fue obtenido por prueba y error. Quisimos ver si surgían estructuras interesantes al permitir que las células se mantuvieran vivas con una cantidad de vecinos múltiplo de 2 (2,4,..etc). Definimos $s_{t+1}(x_i)$ según la fórmula

$$s_{t+i}(x_i) = s_t(x_i) \lfloor 1/(a_t(x_i)\%2 + 1) \rfloor + (1 - s_t(x_i)) \lfloor 1/\left|3 - a_t(x_i) + 1\right| \rfloor \quad \textbf{(12)}$$

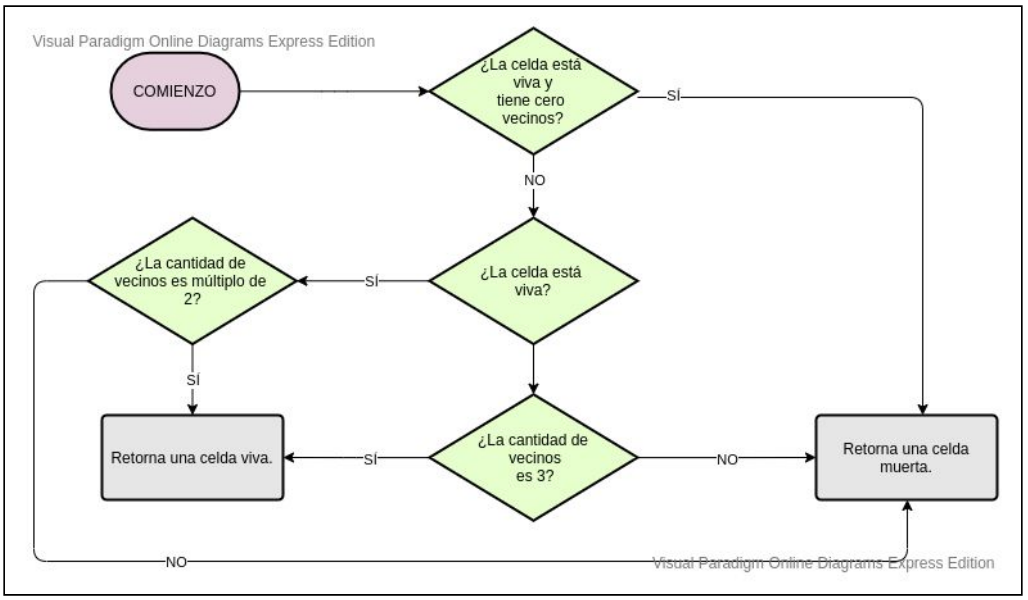

Figura 7: flujo de decisión para aplicar la regla Numbers

En Fig. 7 podemos ver representadas de manera visual las reglas correspondientes a la ec. (12). En nuestro último conjunto de reglas 2D, tomaremos como observable el t al cual el radio de crecimiento se estabiliza, lo cual se define según

$$O = t / r(t) = r(t+1) = r(t+2) = \ldots = r(t+5) \quad \textbf{(13)}$$

donde $r(t)$ es el radio de crecimiento del sistema en el tiempo t.

3.2. Simulaciones 3D

En el caso de las simulaciones 3D, tenemos que $\delta = 3$, $\alpha = 40$, y $\varrho = 0.1$. Por otra parte, la cantidad de iteraciones para permitirle al sistema evolucionar es 150 y, al igual que para la mayoría de los sistemas 2D, se realizaron 10 ejecuciones por cada regla. Esto se mantiene uniforme a lo largo de los tres sistemas descritos debajo.

A diferencia de los sistemas 2D, donde los observables variaron dependiendo de la regla utilizada, en este caso se calculará el mismo observable para las reglas descritas en 3.2.1, 3.2.2 y 3.2.3. Este observable es, al igual que en la regla 3.1.3, la pendiente de la recta correspondiente al radio de crecimiento a medida que evoluciona con el tiempo, con lo cual, la ecuación correspondiente al observable O de cada regla, es ec. (11).

3.2.1 Primes Rules

Sabiendo que p_j es el j-ésimo primo y que comenzamos contando a los primos desde p_1 y que $\pi(n)$ es la función contadora de números primos [1], entonces calculamos $s_{t+1}(x_i)$ como

$$s_{t+1}(x_i) = \sum_{j=1}^{\pi(26)} \lfloor 1/\left|p_j - a_t(x_i) + 1\right| \rfloor \quad \textbf{(14)}$$

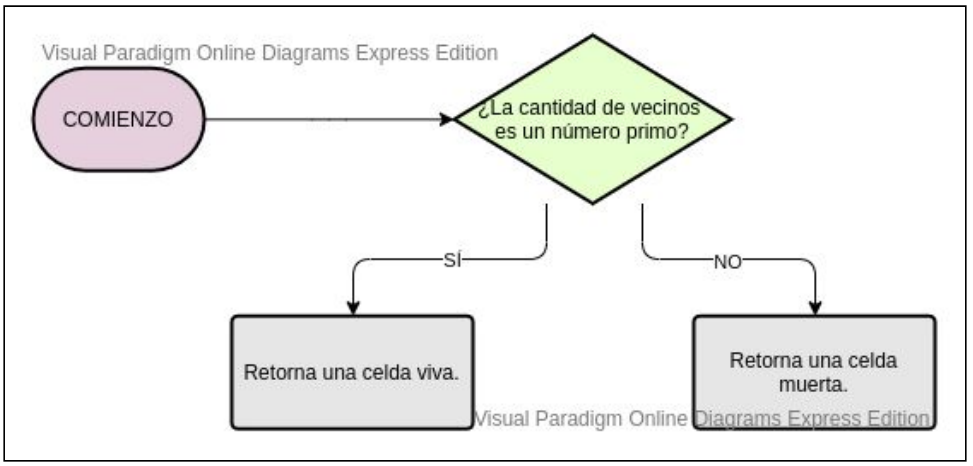

Figura 8: flujo de decisión para aplicar la regla Primes

En este nuevo sistema, podemos ver cómo Fig. 8 representa de manera sencilla el cambio de estado formulado en ec. (14).

3.2.2 Between 3 and 20 Prime Rules

Tomando las mismas aclaraciones sobre los primos realizadas en la subsección 3.1.1, definimos el cambio de estado $s_{t+1}(x_i)$ de la forma

$$s_{t+1}(x_i) = s_t(x_i) \sum_{j=2}^{\pi(20)} \lfloor 1/\left|p_j - a_t(x_i) + 1\right| \rfloor + (1 - s_t(x_i)) \sum_{j=2}^{\pi(10)} \lfloor 1/\left|p_j - a_t(x_i) + 1\right| \rfloor \quad \textbf{(15)}$$

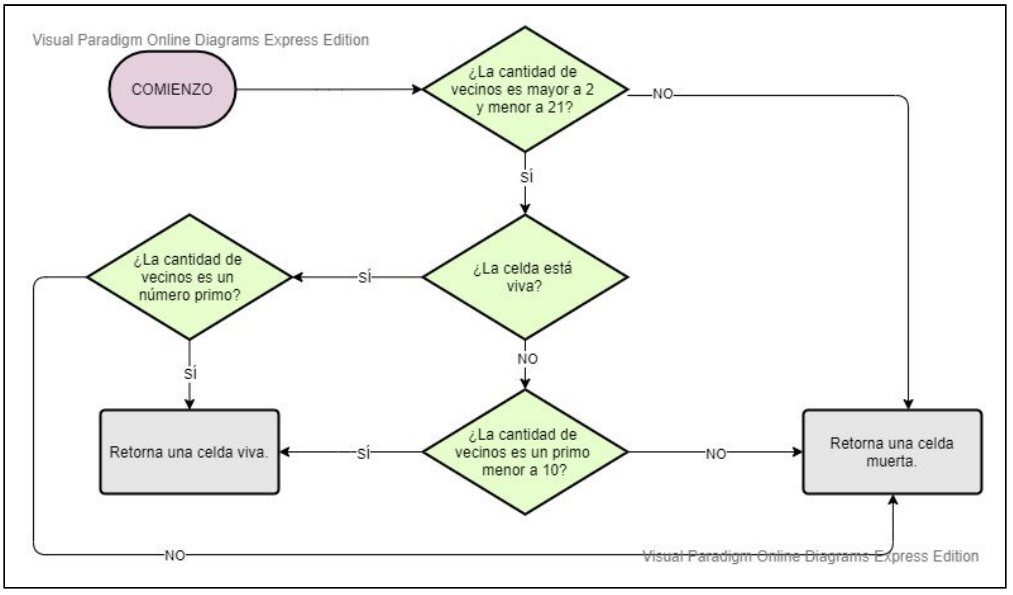

Figura 9: flujo de decisión para aplicar la regla Between 3 and 20 Primes

Este sistema es un poco más restringido que el anterior. Podemos ver esto analizando el diagrama de flujo en Fig. 9, que se corresponde con el cambio de estado que se presenta en la ec. (15).

3.2.3 Different Multiples Rules

En este caso, para la evolución del estado de una celda particular, no sólo van a influir la cantidad de vecinas determinadas por el radio de Moore especificado en ec. (4), sino también la cantidad de celdas vivas en el sistema como conjunto. Sean $m_{3,i}$ los i-ésimos múltiplos de 3, donde $m_{3,0} = 0$, $m_{3,1} = 3$ y así sucesivamente. Por otro lado, $m_{7,i}$ son los i-ésimos múltiplos de 7, con $m_{7,0} = 0$, $m_{7,1} = 7$, etc. Bajo estas condiciones, definimos el cambio de estado de la celda x_i como

$$s_{t+1} = \lceil c_t - 0.5 \rceil \sum_{i=1}^{6} \lfloor 1/\left| m_{3,i} - a_t(x_i) + 1 \right| \rfloor + \lceil 0.6 - c_t \rceil \sum_{i=1}^{2} \lfloor 1/\left| m_{7,i} - a_t(x_i) + 1 \right| \rfloor \quad \textbf{(16)}$$

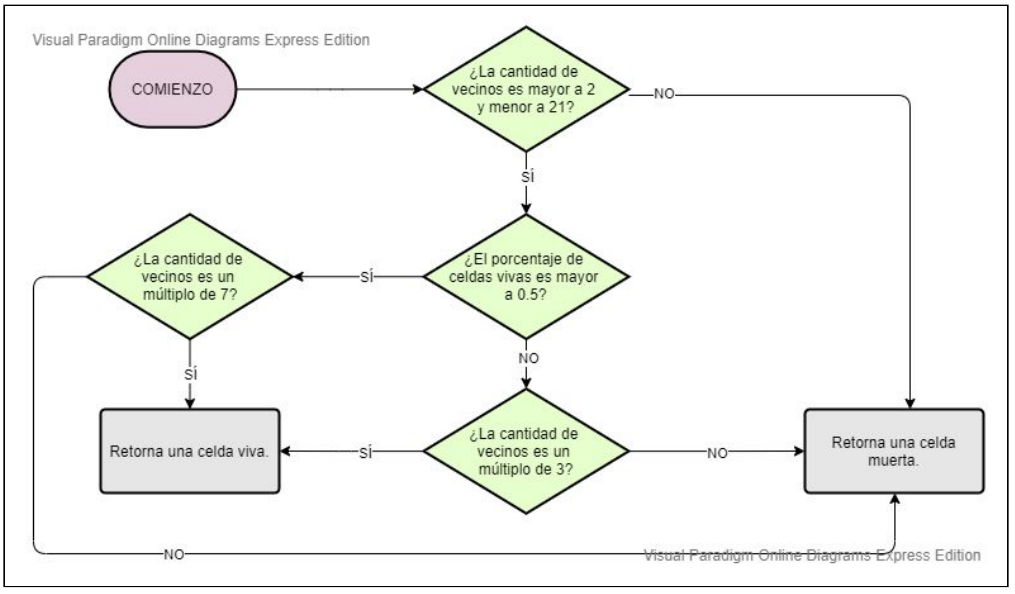

Figura 10: flujo de decisión para aplicar la regla Different Multiples.

En este caso, Fig. 10 nos muestra cómo interpretar la fórmula para el cambio de estado de las celdas que se expresa en ec. (16).

4. RESULTADOS

En esta sección procederemos a analizar cada uno de los sistemas presentados en la sección 3. Presentaremos una o, en caso de ser necesario, dos animaciones por sistema, el post procesamiento para analizar tanto su radio de crecimiento como el porcentaje de celdas vivas a través del tiempo y, por último, resultados sobre el observable correspondiente (también definido para cada regla en su subsección correspondiente).

4.1 Sistemas 2D

Para los sistemas 2D, los parámetros que se mantienen constantes a lo largo de todas las ejecuciones y para todos los resultados son $\delta = 2$, $\alpha = 100$, $\varrho = 0.1$ y 250 iteraciones por ejecución del sistema.

4.1.1 Conway Rules

En este primer caso, mostraremos dos animaciones ya que, según nuestro observable, consideramos dos sistemas de interés: aquellos que estabilizan y aquellos que no lo hacen. Llamaremos Conway Glider a la simulación cuyas celdas vivas alcanzan la pared y Conway Stagnant a aquél que logra estabilizar. Nótese que por cada valor de porcentaje inicial de células vivas, realizamos 100 ejecuciones.

Conway Glider: https://youtu.be/hRCMJAPGx1Q
Conway Stagnant: https://youtu.be/hX2leFcI2d8

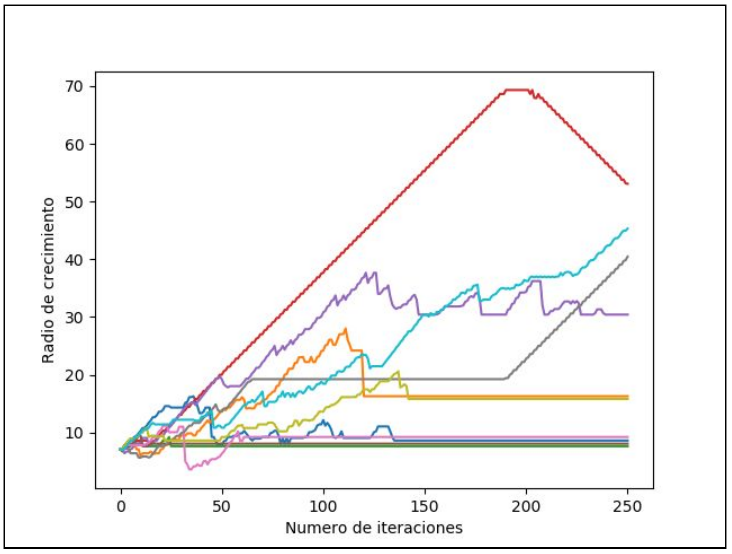

Figura 11: Radio de crecimiento a través del tiempo. Para cada ejecución, $\rho = 0.5$. Se encuentran representadas 10 ejecuciones.

En Fig. 11 podemos distinguir estas dos situaciones comentadas al inicio de la sección. Por una parte tenemos trazas como la roja o la celeste que notamos que se van disparando hasta que el radio de crecimiento alcanza una de las paredes (esto sucede cuando el radio de crecimiento es un poco menor a 50). Por otra parte, rectas como la verde oliva quedan estancadas en un radio de crecimiento menor. Por ejemplo, en estos casos, el radio de crecimiento es aproximadamente 18.

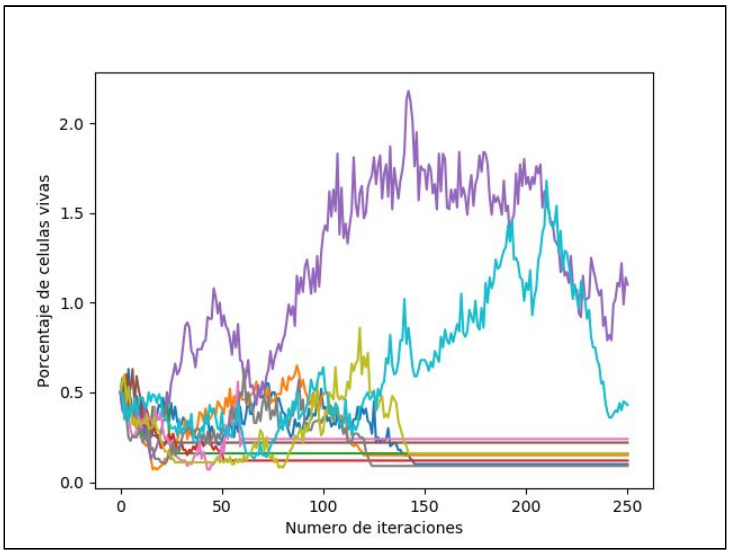

Figura 12: Porcentaje de células vivas a través del tiempo. Para cada ejecución $\rho = 0.5$. Se encuentran representadas 10 ejecuciones.

En Fig. 12 podemos ver como se estanca el porcentaje de células vivas a través del tiempo en varios de los casos.

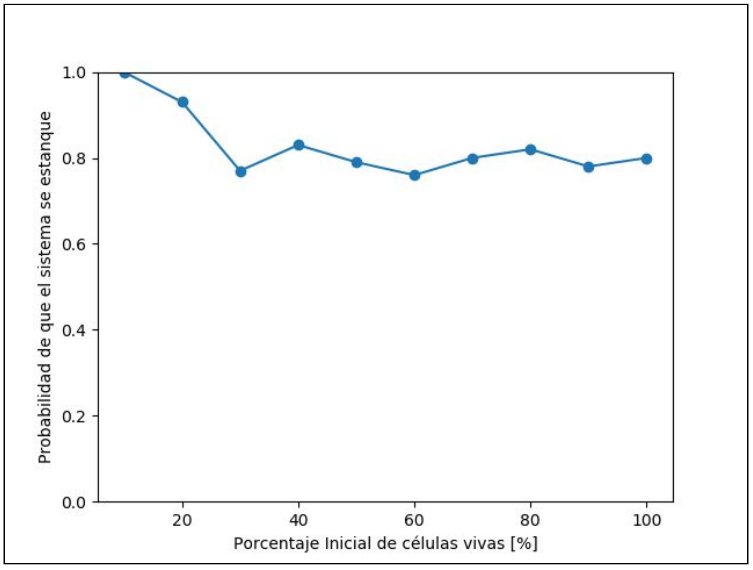

Figura 13: Observable O para Conway Rules. ρ variable entre 10 y 100 a intervalos de 10. Se realizaron 100 corridas por valor de ρ

Este primer observable, cuya gráfica es la correspondiente a Fig. 13 nos muestra cómo cambia la probabilidad de que el sistema se estanque al tiempo que vamos variando el porcentaje de células vivas. En este caso, nos interesa aquellos sistemas que se estancan, pues obtuvimos un segundo observable. Este segundo observable *O'* presenta cómo varía el radio al cual se estancan en función del porcentaje de células vivas iniciales. Podemos observar el gráfico correspondiente en Fig. 14.

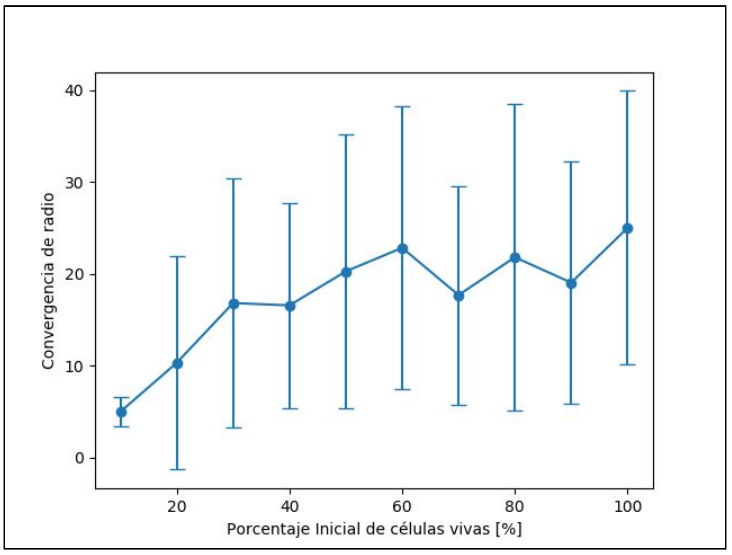

Figura 14: Observable O' para Conway Rules. Calculado a partir de aquellas ejecuciones $I_k / I_k = 1$, con ρ variable entre 10 y 100 a intervalos de 10.

4.1.2 Unrestricted Multiple Rules

Presentaremos una animación de esta regla y luego los diferentes gráficos correspondientes. La animación se presenta debajo de acuerdo al nombre Different Multiples. Recordemos que nuestro observable O para Unrestricted Multiples era la velocidad de crecimiento del radio de crecimiento del sistema.

Unrestricted Multiples: https://youtu.be/9IEWA45-PR0

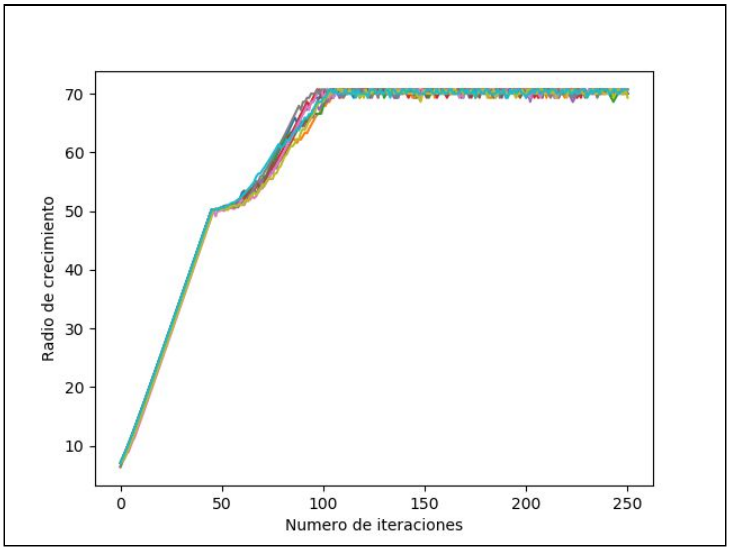

Figura 15: Radio de crecimiento a través del tiempo. Para cada ejecución $\rho = 0.5$. Se encuentran representadas 10 ejecuciones.

En el sistema Unrestricted Multiples, la Fig. 15 nos muestra que, para un porcentaje de células vivas iniciales de 0.5, podríamos sospechar que casi todos los casos alcanzarán la pared. Es clara la

uniformidad con la cual las distintas ejecuciones alcanzan la pared (a un radio de crecimiento de valor un poco menor a 50) con una velocidad que pareciera crecer de manera lineal.

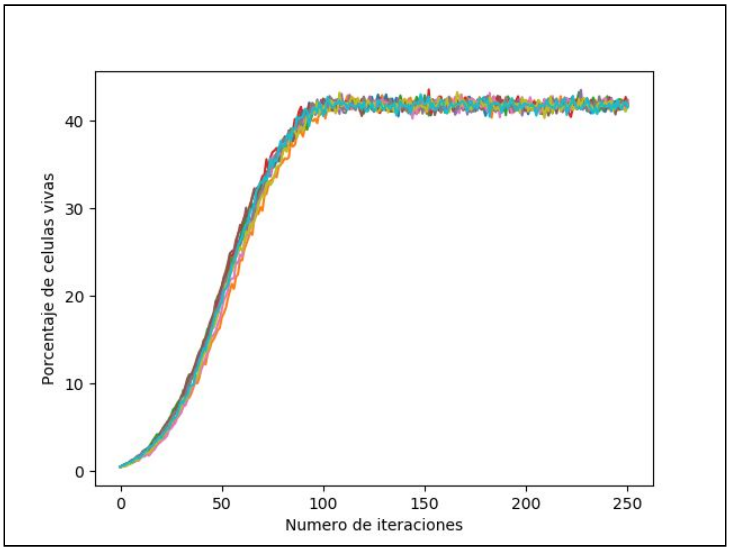

Figura 16: Porcentaje de células vivas a través del tiempo. Para cada ejecución $\rho = 0.5$. Se encuentran representadas 10 ejecuciones.

Por otro lado, la evolución en el porcentaje de células vivas vs. el número de iteraciones que se presenta en Fig. 16 es bastante uniforme para todas las ejecuciones, de la misma manera que las ejecuciones son similares en Fig. 15.

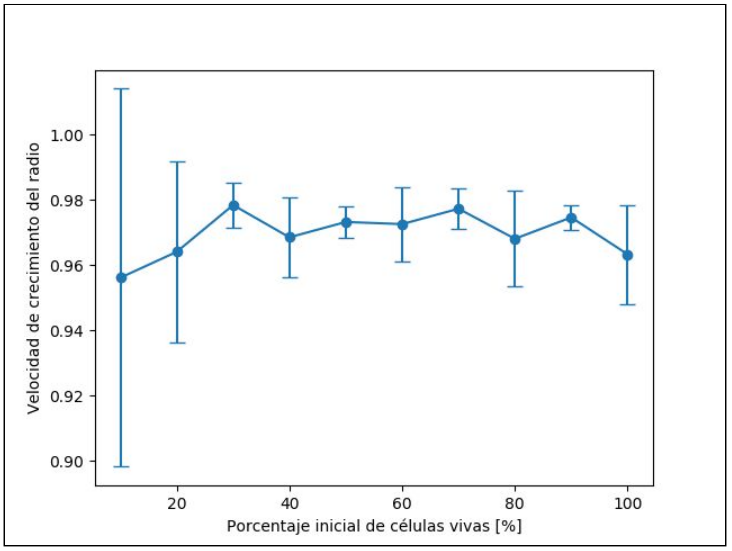

Figura 17: Observable O para Unrestricted Multiple Rules. Calculado a partir de 10 ejecuciones para cada valor de ρ, el cual varía entre 10 y 100.

En Fig. 17 podemos observar cómo va variando la velocidad de crecimiento del radio en función del porcentaje inicial de células vivas. Nótese que el caso en que $\rho = 0.1$ es bastante inestable, pues su desvío estándar alcanza todos los valores del eje y.

4.1.2 Numbers Rules

En el caso de Numbers, el tercero de los sistemas 2D, el observable O elegido era el tiempo t en el cual el sistema se volvía estacionario. La animación, al igual que en los casos anteriores, se encuentra en el link debajo.

Numbers: https://youtu.be/xD-WN6RQkIo

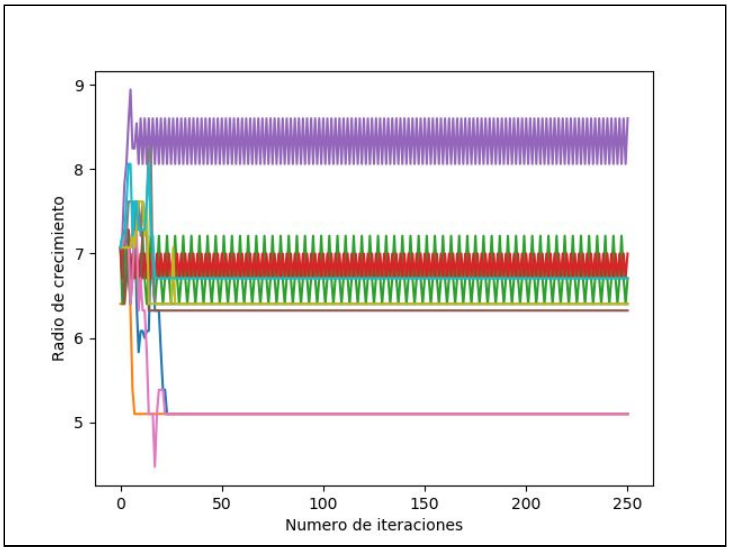

Figura 18: Radio de crecimiento a través del tiempo. Para cada ejecución $\rho = 0.5$. Se encuentran representadas 10 ejecuciones.

Fig. 18, podemos notar que el sistema suele llegar a un estado estacionario, ya sea completamente constante o oscilatorio. Las ejecuciones corresponden a un porcentaje inicial de celdas vivas de 0.5, pero luego en el observable buscaremos analizar si esta tendencia se repite.

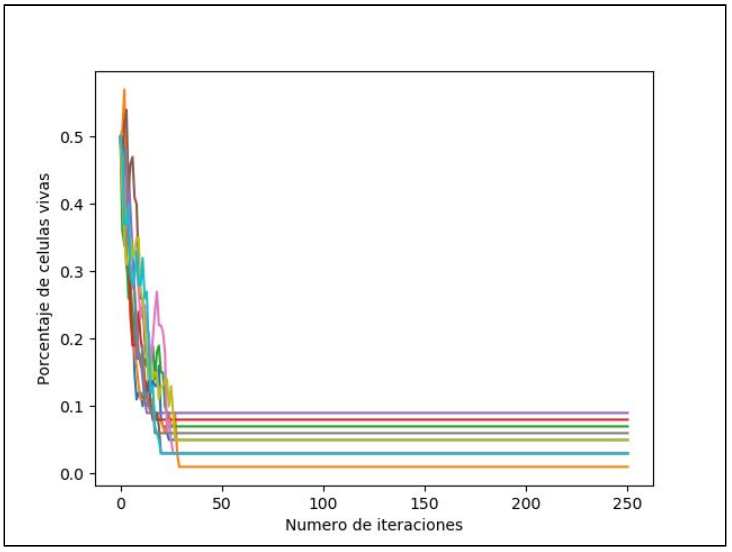

Figura 19: Porcentaje de células vivas a través del tiempo. Para cada ejecución $\rho = 0.5$. Se encuentran representadas 10 ejecuciones.

En Fig. 19 notamos que casi todas las ejecuciones para estos sistemas particulares caen en cuanto a porcentaje de células vivas vs. número de iteraciones. Por otro lado, es necesario resaltar que el porcentaje (que se mueve entre valores 0 y 100) es extremadamente pequeño tanto para su punto máximo como para el valor estable.

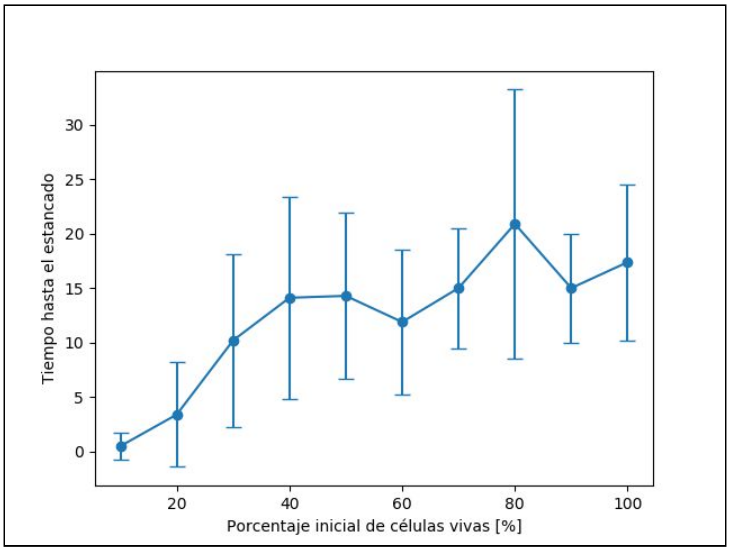

Figura 20: Observable O para Numbers Rules. Calculado a partir de 10 ejecuciones para cada valor de ρ, el cual varía entre 10 y 100 a intervalos de 10.

En Fig. 20 vemos el observable *O* que buscamos estudiar para la Numbers Rules. En este caso, buscamos medir el tiempo *t* al cual se estanca el sistema. Notar que el valor máximo no supera los 40, siendo cada ejecución de 250 iteraciones.

4.2 Sistemas 3D

En los sistemas 3D también tenemos valores que se mantienen constantes. Por un lado, los parámetros estáticos son $\delta = 3$, $\alpha = 40$, $\varrho = 0.1$ y 150 iteraciones por ejecución del sistema. Por otra parte, para cada valor de ρ entre 10 y 100 (a intervalos de 10), se realizaron 10 simulaciones.

4.2.1 Primes Rules

Esta regla es bastante sencilla, ya que una célula se mantiene viva o nace una nueva si la cantidad de vecinos es un número primo. Debajo, se encuentra el link correspondiente a la animación de este sistema, con los parámetros descritos en la sección 4.2.

Primes: https://youtu.be/IOFBuV55tzs

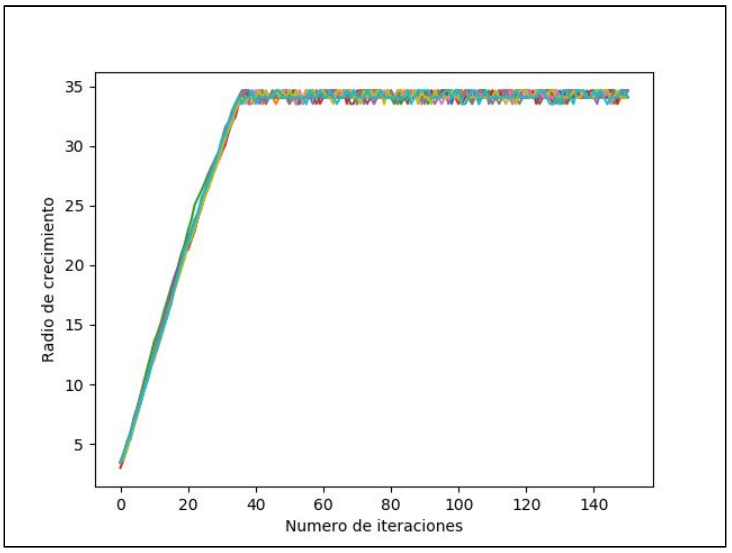

Figura 21: Radio de crecimiento a través del tiempo. Para cada ejecución $\rho = 0.5$. Se encuentran representadas 10 ejecuciones.

En Fig. 21 podemos ver cómo casi todas las ejecuciones se comportan de manera similar: el radio de crecimiento crece de manera lineal hasta alcanzar la pared del sistema.

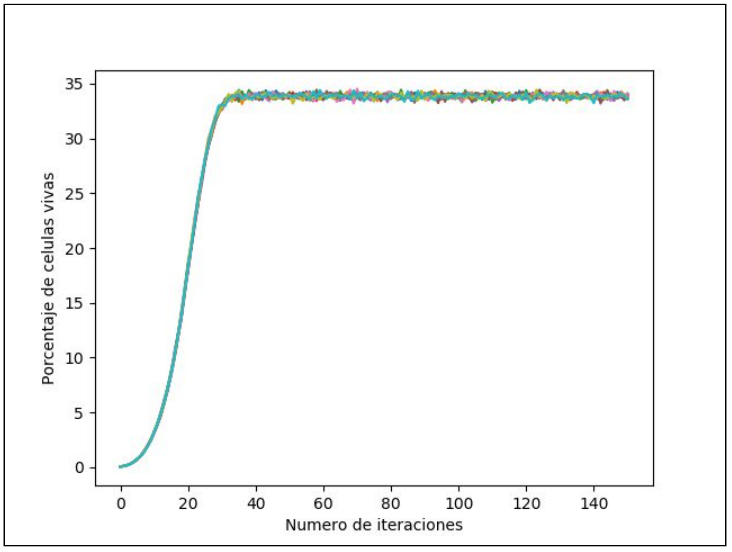

Figura 22: Porcentaje de células vivas a través del tiempo. Para cada ejecución $\rho = 0.5$. Se encuentran representadas 10 ejecuciones.

En Fig. 22, analizando la evolución del porcentaje de células vivas vs. el número de iteraciones, podemos ver que hay muy poca diferencia entre las ejecuciones, lo cual se asemeja a la situación observada en Fig. 21.

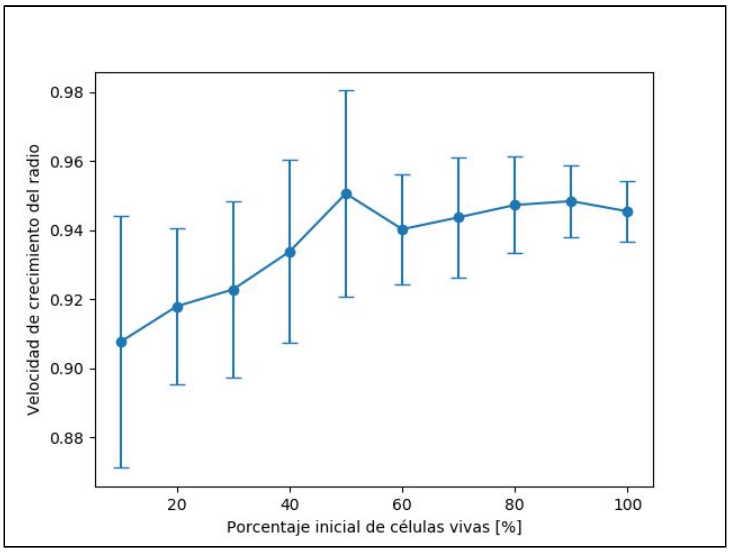

Figura 23: Observable O para Less than 41 Primes Rules. Calculado a partir de 10 ejecuciones para cada valor de ρ, el cual varía entre 10 y 100 a intervalos de 10.

Como se expresó en la sección 3.2, para todos los sistemas 3D fue tomado el mismo observable: la velocidad de crecimiento del radio del sistema en función del porcentaje inicial de células vivas.

4.2.2 Between 3 and 20 Primes Rules

Este sistema busca ser un poco más restrictivo que el anterior ya que acota a los primos posibles tanto para celdas vivas como muertas (para muertas lo hace un poco más). A continuación, se encuentra el link de la animación del sistema.

Between 3 and 20 Primes: https://youtu.be/hR4hVGmbvnQ

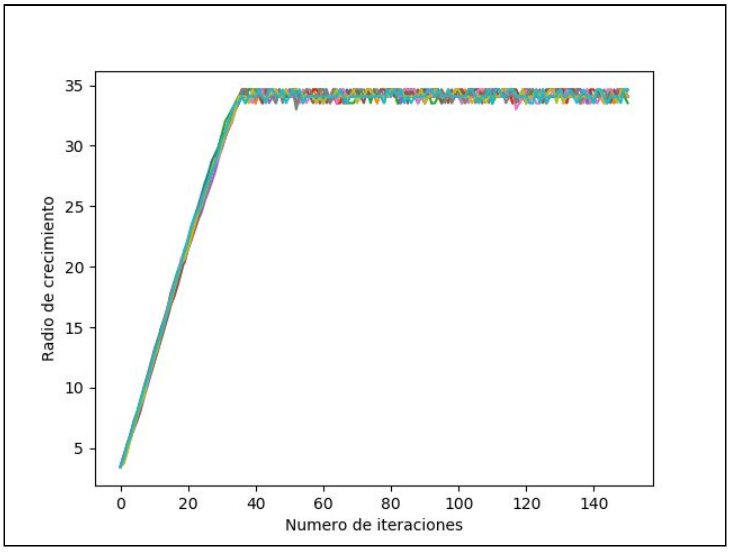

Figura 24: Radio de crecimiento a través del tiempo. Para cada ejecución $\rho = 0.5$. Se encuentran representadas 10 ejecuciones.

En Fig. 24 podemos ver un gráfico bastante similar a su semejante en la sección 4.2.1. El radio de crecimiento evoluciona de manera lineal con respecto al número de iteraciones.

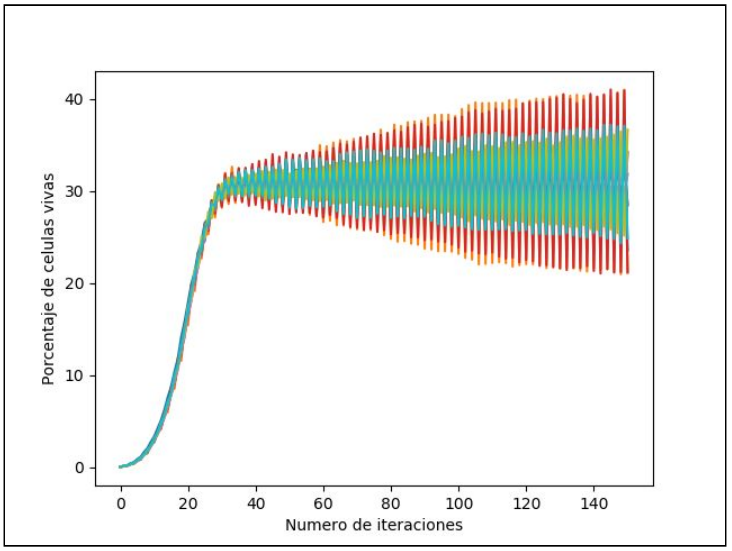

Figura 25: Porcentaje de células vivas a través del tiempo. Para cada ejecución $\rho = 0.5$. Se encuentran representadas 10 ejecuciones.

Por otro lado, en Fig. 25 vemos que, hasta que la primera célula toca la pared, el porcentaje de células vivas vs. el número de iteraciones crece de manera exponencial.

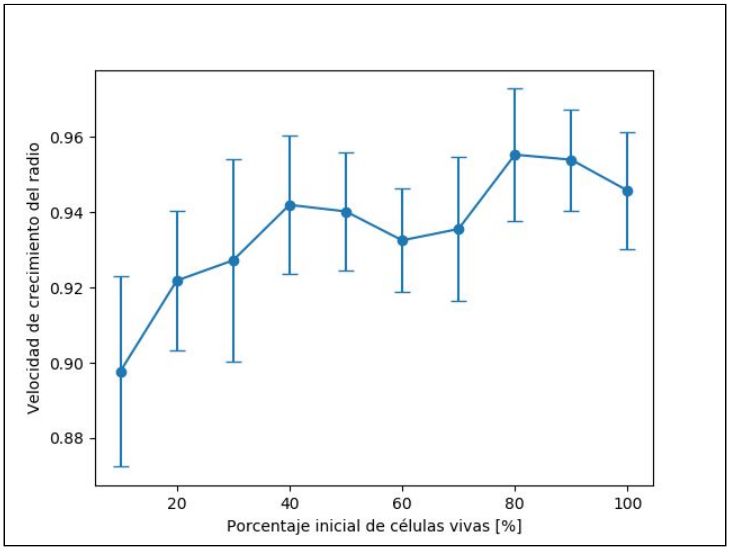

Figura 26: Observable O para Between 3 and 20 Primes Rules. Calculado a partir de 10 ejecuciones para cada valor de ρ, el cual varía entre 10 y 100 a intervalos de 10.

Por último, en Fig. 26, vemos la representación del observable *O*, el cual, nuevamente es la velocidad de crecimiento del radio en función del porcentaje inicial de células vivas.

4.2.2 Different Multiples Rules

Finalmente presentamos nuestro último sistema, el cual no sólo tiene en cuenta los vecinos de acuerdo a la ec. (3), sino también la evolución del sistema en su totalidad. A continuación, una animación del mismo:

Different Multiples: https://youtu.be/RXcC9WAmVs4

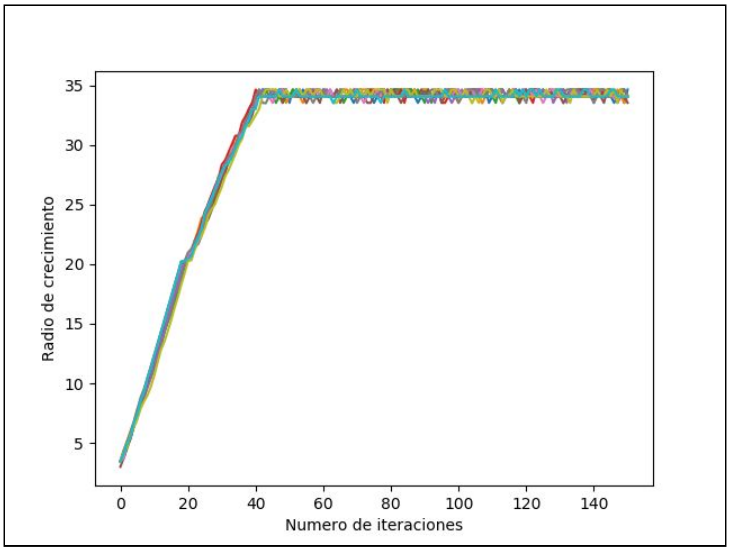

Figura 27: Radio de crecimiento a través del tiempo. Para cada ejecución $\rho = 0.5$. Se encuentran representadas 10 ejecuciones.

Según Fig. 27, este sistema, cuando su porcentaje inicial de celdas vivas es de 0.5, tiene un radio de crecimiento que evoluciona de manera lineal a través del tiempo. Podemos ver que, a pesar de que las inicializaciones fueron realizadas de manera random, no hay demasiada diferencia en cómo estas simulaciones se comportan.

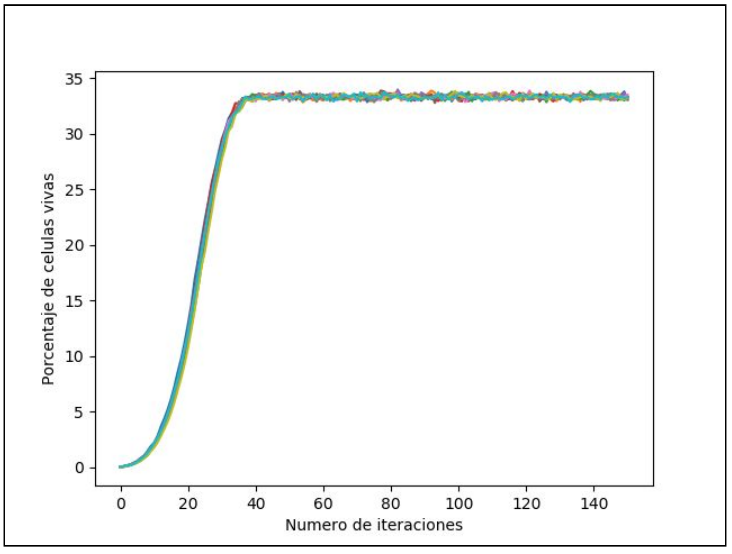

Figura 28: Porcentaje de células vivas a través del tiempo. Para cada ejecución $\rho = 0.5$. Se encuentran representadas 10 ejecuciones.

Podemos ver en Fig. 28, cómo la uniformidad de las distintas ejecuciones se corresponde con la presente en Fig. 27. En este caso notamos que hay un crecimiento exponencial del porcentaje de células vivas en función del número de iteraciones realizadas.

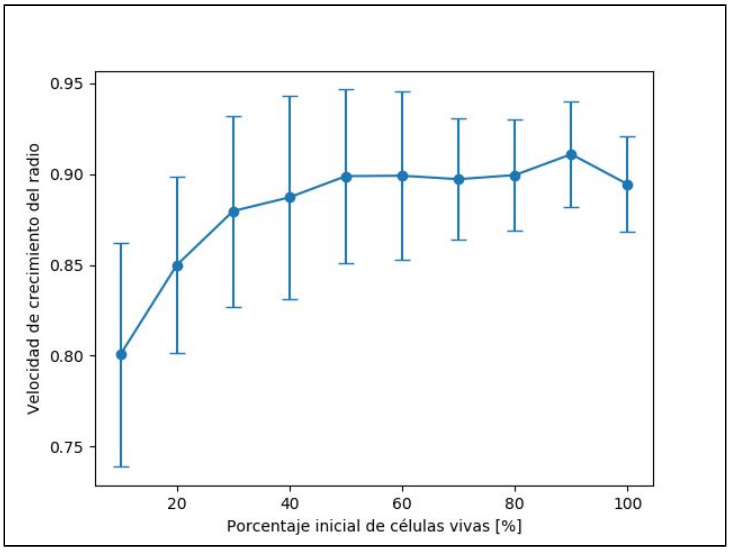

Figura 29: Observable O para Different Multiples Rules. Calculado a partir de 10 ejecuciones para cada valor de ρ, el cual varía entre 10 y 100 a intervalos de 10.

Por último, en Fig. 29 vemos la representación gráfica del observable elegido para este sistema. Este observable, recordemos, es la velocidad de crecimiento del radio del sistema (pendiente de la recta del radio de crecimiento del sistema).

5. CONCLUSIONES

- Los sistemas 3D tuvieron comportamientos similares, independientemente de la regla utilizada. En todos ellos, el radio de crecimiento del sistema creció de forma lineal hasta alcanzar la pared, y sus masas crecieron exponencialmente hasta el tiempo t en el cual la primera celda alcanzó la pared del sistema.
- En los sistemas 3D, el valor medio de radio de crecimiento (observable) era proporcional al porcentaje de células vivas iniciales (input). Pero el desvío era lo suficientemente grande como para creer que esta relación no existe.
- En la regla de Conway (2d), se generaron los patrones descritos en las referencias [3], y en particular, se detectaron los gliders que generaban un crecimiento del radio hasta el borde del sistema.
- En los sistemas de la regla de Conway, los radios de crecimiento del sistema al cual dicho sistema alcanzaba convergencia (para los casos en que la misma se producía) tienen un valor medio proporcional al porcentaje de células vivas inicial, pero el desvío es lo suficientemente grande como para creer que esta relación no existe.

REFERENCIAS

[1] Función contador de primos,
https://es.wikipedia.org/wiki/Funci%C3%B3n_contador_de_n%C3%BAmeros_primos

[2] numpy.polyfit https://numpy.org/doc/stable/reference/generated/numpy.polyfit.html
[3] Berlekamp, E. R.; Conway, John Horton; Guy, Richard K. (2004). Winning ways for your mathematical plays (2nd ed.). Natick, Mass: A K Peters. ISBN 156881142X. OCLC 560267317