

Castillo Pruneda Rogelio Gabriel

López Hernández Oscar

Rosales Valderrama Brian Axel


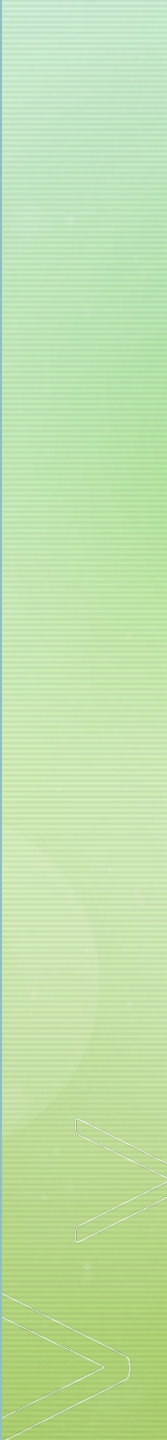



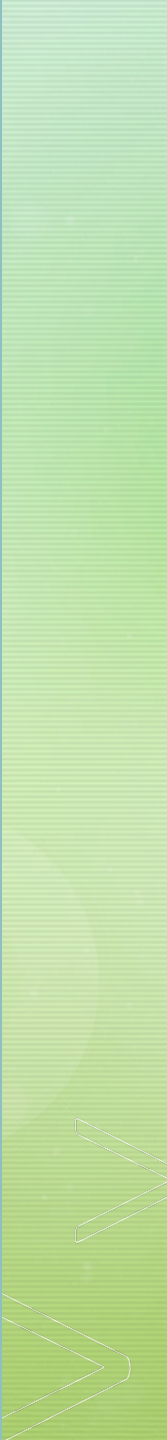
PyCX



¿Qué es PyCX?

- El proyecto PyCX tiene como objetivo desarrollar un repositorio en línea de códigos de muestra de Python simples, crudos pero fáciles de entender para el modelado y simulación de sistemas dinámicos complejos, incluidos mapas iterativos, ecuaciones diferenciales ordinarias y parciales, autómatas celulares, análisis de redes, redes dinámicas, y modelos basados en agentes

- 
- La filosofía central de PyCX se basa en la simplicidad, legibilidad, generalización y valores pedagógicos de los códigos de simulación. PyCX se ha utilizado en instrucciones de modelado de sistemas complejos en varios lugares con resultados exitosos.
- 

- 
- Hasta finales del siglo pasado, las simulaciones dinámicas de sistemas complejos estaban disponibles solo para investigadores con habilidades técnicas avanzadas. Existían pocos paquetes de software de simulación de propósito general y eran difíciles de usar sin experiencia en informática, lo que limitaba el crecimiento de la ciencia de sistemas complejos.
- 

- En la última década, se han desarrollado varios paquetes de software de simulación fáciles de usar, como NetLogo, Repast, Mason y Golly, que han facilitado el acceso al modelado y simulación de sistemas complejos para investigadores de diversas disciplinas. Estos paquetes han sido fundamentales para ampliar el uso de la simulación en la investigación científica, con abundantes tutoriales y modelos de simulación disponibles.

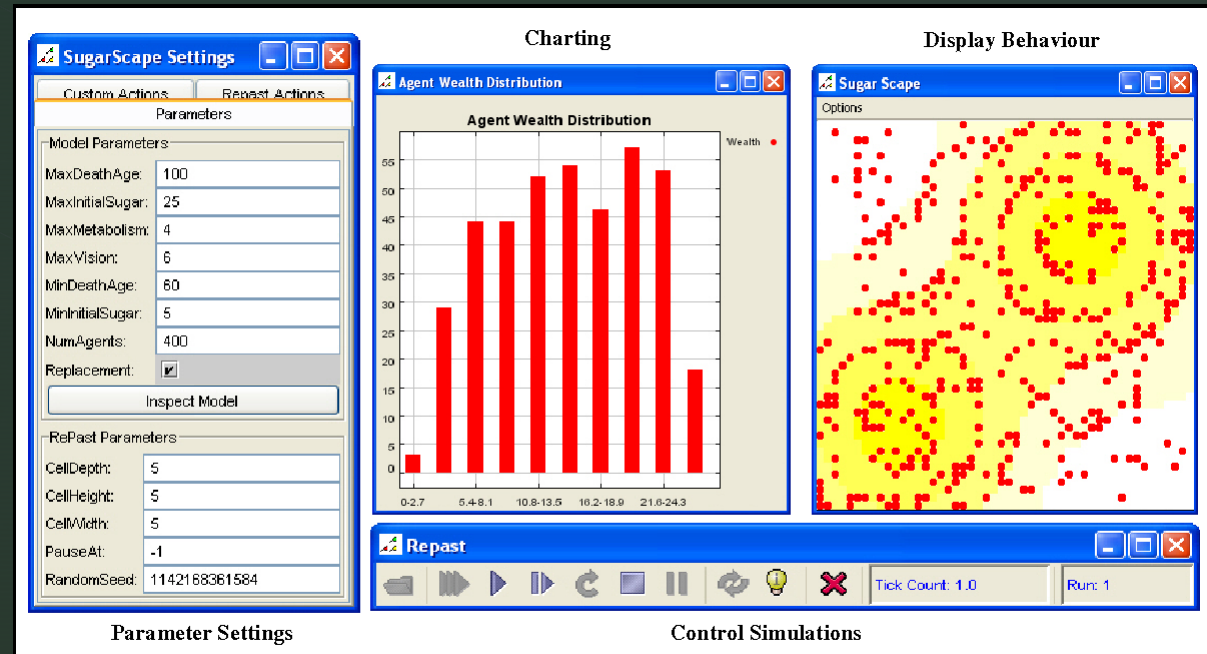

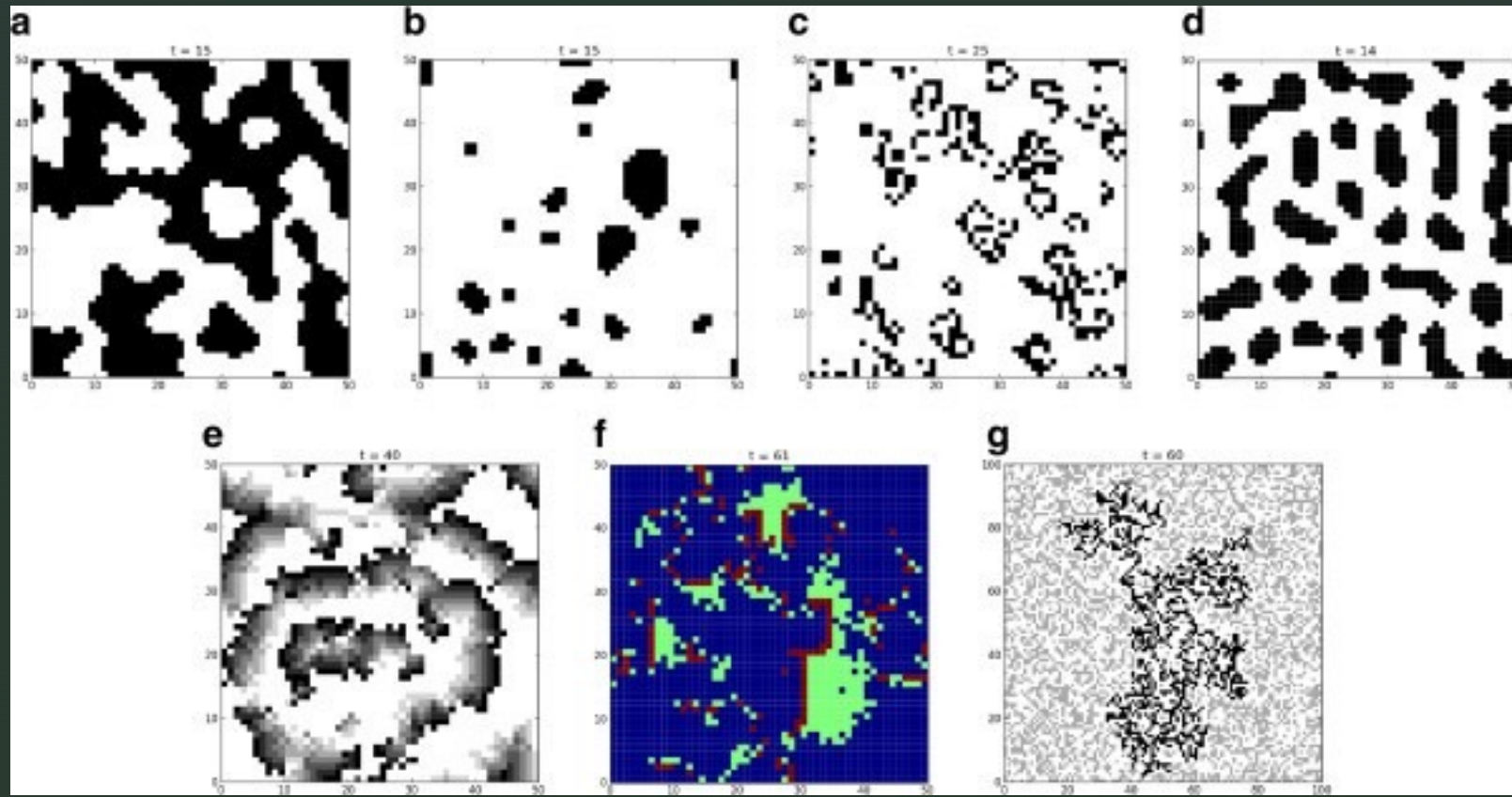


Figure 1: Typical Repast features

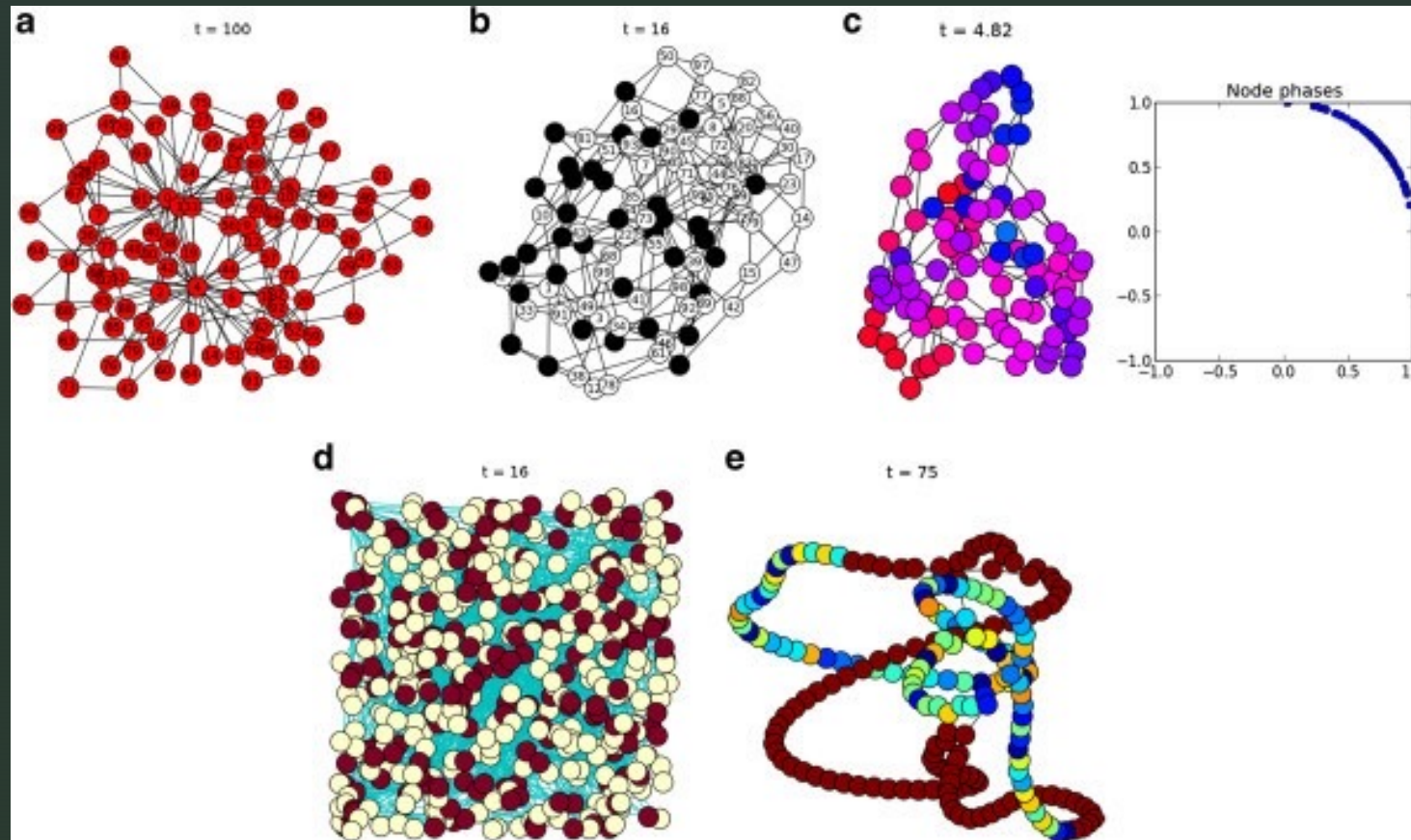
- 
- No obstante, estos programas presentan problemas en la educación superior. Primero, aprender a usar software específico no mejora las habilidades técnicas generales de los estudiantes, lo cual es importante ya que muchos trabajarán fuera de la ciencia de sistemas complejos. Segundo, las preferencias de software varían entre disciplinas, dificultando encontrar una única herramienta útil para todos los estudiantes. Tercero, los detalles de las suposiciones e implementaciones de los modelos a menudo están ocultos, lo que puede influir en los resultados de las simulaciones. Finalmente, el uso de software existente puede limitar la creatividad en la investigación debido a las restricciones impuestas por las herramientas disponibles.

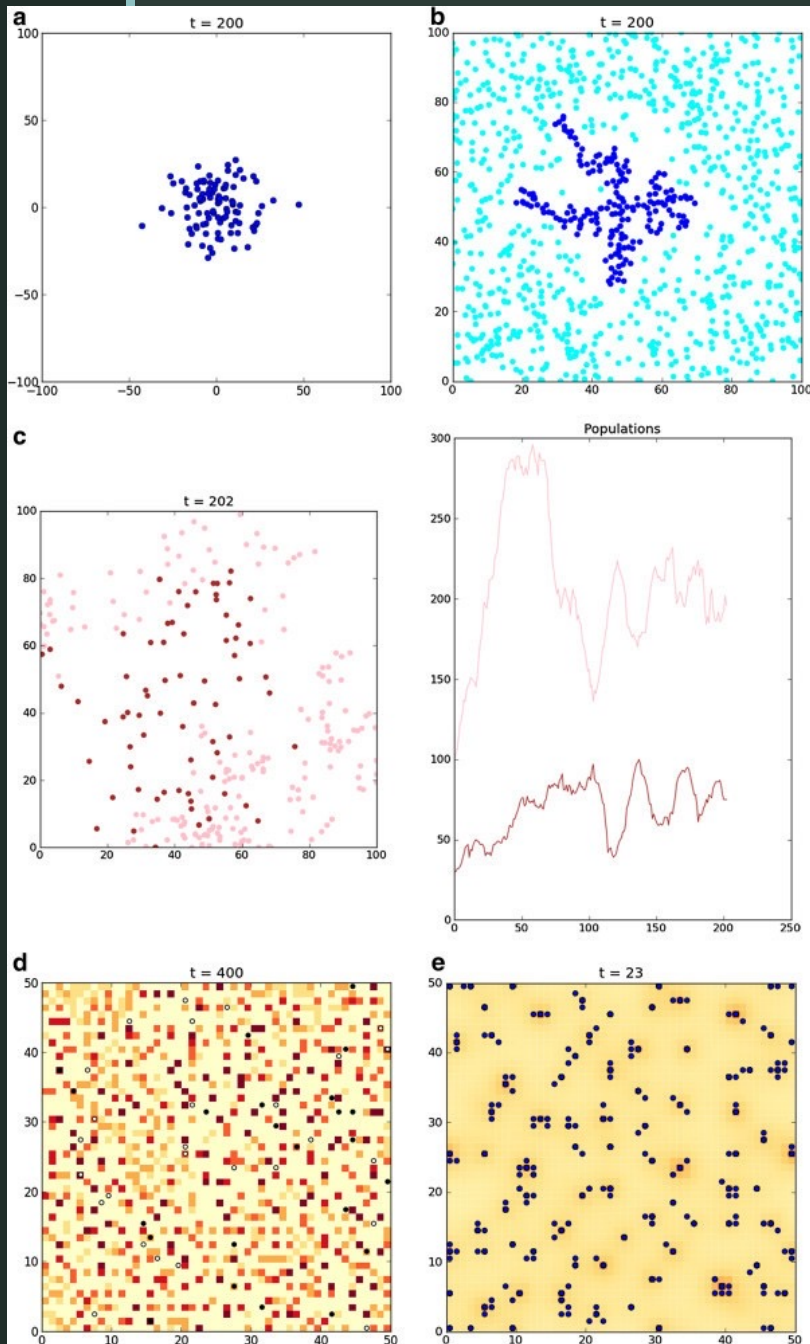


- Autómatas celulares (regla de la mayoría local, regla de las gotas, juego de la vida, formación de patrones de Turing, medios excitables, interacción huésped-patógeno, incendio forestal; Figura 3)



- Redes dinámicas (construcción y análisis básicos de la red, crecimiento de la red mediante vínculo preferencial, gobierno de la mayoría local en una red, sincronización en una red basada en el modelo de Kuramoto (Strogatz 2000), propagación de enfermedades en una red, cascada de fallas en una red; Figura 4)

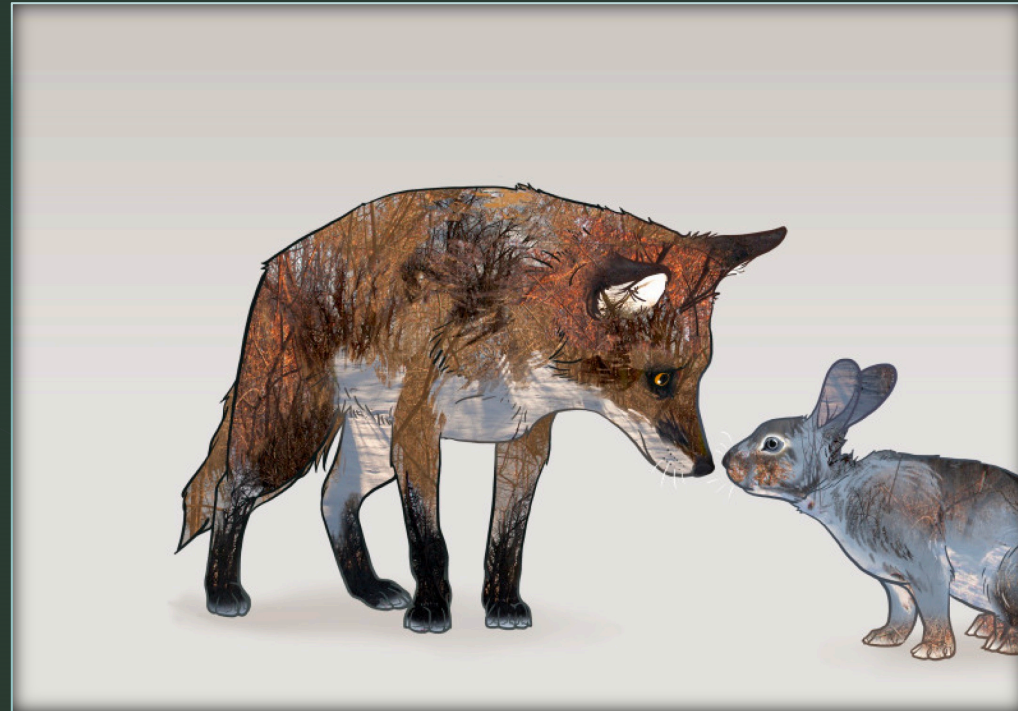




- Modelos basados en agentes (paseo aleatorio de partículas, agregación limitada por difusión, ecosistema depredador-presa, recolección de basura por hormigas (Resnick 1997), agregación de hormigas mediante comunicación basada en feromonas; Figura 5)

Modelo presa depredador

El modelo presa-depredador es un sistema dinámico que describe las interacciones entre dos especies: la presa, que es consumida por el depredador. Se formula comúnmente usando ecuaciones diferenciales, como el modelo de Lotka-Volterra, que describe cómo cambian las poblaciones de presa y depredador con el tiempo.



Método de Runge Kutta orden 4

Los métodos de Runge-Kutta son una familia de métodos iterativos usados para resolver ecuaciones diferenciales ordinarias (EDOs).

Consideremos el siguiente sistema de ecuaciones ordinarias

$$\begin{cases} \frac{dx}{dt} = f(x, y, t) \\ \frac{dy}{dt} = g(x, y, t) \end{cases}$$
$$x(t_0) = x_0, \quad y(t_0) = y_0$$

Formulas

$$\frac{dx}{dt} = f(x, y, t)$$

$$k_1 = h * f(x, y, t)$$

$$k_2 = h * f\left(x + \frac{k_1}{2}, y + \frac{l_1}{2}, t + \frac{h}{2}\right)$$

$$k_3 = h * f\left(x + \frac{k_2}{2}, y + \frac{l_2}{2}, t + \frac{h}{2}\right)$$

$$k_4 = h * f(x + k_3, y + l_3, t + h)$$

$$x(t + h) = x(t) + \frac{1}{6}(k_1 + 2k_2 + 2k_3 + k_4)$$

$$\frac{dy}{dt} = g(x, y, t)$$

$$l_1 = h * g(x, y, t)$$

$$l_2 = h * g\left(x + \frac{k_1}{2}, y + \frac{l_1}{2}, t + \frac{h}{2}\right)$$

$$l_3 = h * g\left(x + \frac{k_2}{2}, y + \frac{l_2}{2}, t + \frac{h}{2}\right)$$

$$l_4 = h * g(x + k_3, y + l_3, t + h)$$

$$y(t + h) = y(t) + \frac{1}{6}(l_1 + 2l_2 + 2l_3 + l_4)$$

Módulo analítico

$x(t)$: Representa a la presa

$y(t)$: Representa al depredador

Ecuaciones:

$$x'(t) = r_1x(t) - d_1x(t)y(t)$$

$$y'(t) = -r_2y(t) + d_2x(t)y(t)$$

Donde:

r_1 : Taza de aumento de presas en ausencia de depredadores

r_2 : Taza de disminución de depredadores en ausencia de presa

d_1 : Suceptibilidad de las presas a ser cazadas

d_2 : Capacidad de depredación de los depredadores

Ejemplo

$$r_1 = 0.4$$

$$r_2 = 0.3$$

$$d_1 = 0.37$$

$$d_2 = 0.05$$

Parametros iniciales del sistema:

$$x(0) = 3 \quad y(0) = 1$$

CONCLUSIONES

- Hemos presentado PyCX, un repositorio en línea de códigos de simulaciones de sistemas complejos, diseñado principalmente para la educación superior. PyCX utiliza Python como plataforma de modelado y simulación, superando varias limitaciones del software existente. PyCX combina novedades y tradiciones. Su aspecto técnico es novedoso porque se basa en Python y sus módulos complementarios, como NetworkX. Su filosofía central es antigua: todos deberían escribir sus propios códigos de simulación para explorar métodos de modelado y análisis únicos, como hacían los investigadores en el siglo XX. Al mejorar la alfabetización en programación a través de PyCX, esperamos que la comunidad científica de sistemas complejos mantenga su creatividad e innovación. El proyecto PyCX sigue evolucionando y agradecemos comentarios y aportes de investigadores y educadores interesados en participar.