Universidad Nacional Autónoma de México

POSGRADO EN CIENCIA E INGENIERÍA DE LA COMPUTACIÓN

Modelo de Conexión de Redes entre Sitios Arqueológicos

T E S I S

QUE PARA OPTAR POR EL TÍTULO DE:

Maestro en Ciencias e Ingeniería de la Computación

P R E S E N T A :

Paris Alejandro Dávalos Bravo

TUTOR

Dr. Alfonso Gastélum Strozzi

Ciudad Universitaria, Cd. Mx., 2024

Universidad Nacional Autónoma de México
UNAM
Biblioteca Central
Dirección General de Bibliotecas de la UNAM

UNAM – Dirección General de Bibliotecas
Tesis Digitales
Restricciones de uso

DERECHOS RESERVADOS ©
PROHIBIDA SU REPRODUCCIÓN TOTAL O PARCIAL

Todo el material contenido en esta tesis esta protegido por la Ley Federal del Derecho de Autor (LFDA) de los Estados Unidos Mexicanos (México).

El uso de imágenes, fragmentos de videos, y demás material que sea objeto de protección de los derechos de autor, será exclusivamente para fines educativos e informativos y deberá citar la fuente donde la obtuvo mencionando el autor o autores. Cualquier uso distinto como el lucro, reproducción, edición o modificación, será perseguido y sancionado por el respectivo titular de los Derechos de Autor.

Dedicatoria a ti.

Agradecimientos

Agradecimientos a mi madre y a mi tía que siempre me han apoyado. Al doctor Alfonso y su paciencia y su actitud que me demostró. Al CONACyT. A aquellas personas que conocí gracias a este trabajo y a los viejos amigos insuperables en esta travesía. Gracias totales.

Índice general

Índice de figuras

Capítulo 1

Introducción

El análisis de redes como herramienta para distintas áreas ha sido de gran utilidad y a ido en aumento en los últimos años. Para las ciencias sociales no han sido la excepción, en efecto el concepto de pensar que los individuos están incrustados en densas redes de interacciones y relaciones sociales ya es viejo. Sin embargo, no fue hasta 1932 que Jacob Moreno fundó la sociometría un proceso usado para medir las relaciones sociales entre los individuos. Desde entonces el uso de teoría de grafos y álgebra matricial en las ciencias sociales ha ido en aumento y la arqueología no es la excepción a esto de acuerdo con (Barnes and Harary (1983)).

Este trabajo tiene un enfoque multidisciplinario en el cual se presenta un análisis de redes para un conjunto de datos arqueológicos. Este conjunto de datos fue obtenido gracias al INAH (Instituto Nacional de Antropología e Historia) Michoacán y está conformado por objetos metálicos del periodo posclásico hallados en mesoamérica, en una sección posterior se hablará más a fondo de la base de datos y los problemas enfrentados con esta. Al ser un trabajo para obtener el título de maestro en ciencias e ingenieria de la computación las conclusiones arqueológicas de la red se omitieron; sin embargo, en esta introducción se hablará un poco del contexto arqueológico de sus datos y la importancia de los mismos.

El aspecto del metal en las sociedades mesoamericanas es un tema aún abierto e interesante que se estudia aún al día de hoy. El metal, específicamente el cobre, surge en el occidente de México, los arqueólogos sostienen que fue debido al contacto

marítimo con el norte de Sudamérica y el Occidente de México. La metalurgia en el occidente se puede dividir en dos fases (600 d.C. - 1200 d.C.) y (1200 d.C-1510 d.C) diferenciadas por tecnologías diferentes para trabajar el metal. La primera etapa de occidente está caracterizada por la presencia de sociedades pequeñas a diferencia de otros sitios de la época como Teotihuacán. Otra característica importante del surgimiento del metal es la ausencia de propagación en Teotihuacan (100-750 d.C.) o en la civilización clásica maya (150-900 d.C.) cuando el primer objeto de metal se encontró en el occidente de México alrededor de 600 d.C. y estos dos lugares fueron los más desarrollados, influyentes e importantes en su tiempo en Mesoamérica según el trabajo de (Hosler (2005)). También estas sociedades son de las más estudiadas en la actualidad junto a la sociedad azteca por lo que resalta la importancia de estos trabajos pues el metal no se encontraba ahí. La segunda etapa está caracterizada porque se dio en el estado Tarasco. Ambas introducciones de las técnicas metalúrgicas se cree que vinieron de rutas marítimas con el sudamerica. La importancia del imperio tarasco no solo recae en la producción del cobre; a pesar de su poca publicidad actual comparados con otra cultura para el año de 1522 eran el segundo imperio más grande de Mesoamérica. Llegando incluso a contrarrestar las acometidas de los aztecas por incorporarlos a su imperio en varias ocasiones (Perlsteinpollard (2004)). La importancia de los tarascos no se limita a lo bélico, los contactos comerciales fueron diversos, se ha encontrado que obtenían bienes como la turquesa y la pirita del sur occidente de Estados Unidos donde no solo era el ámbito del comercio, si no también culturalmente por ejemplo en la adopción de tipos cerámicos parecidos a los del resto del occidente (Albiez-Wieck (2017)). Sin embargo, el caso del metal es distinto, se han encontrado cascabeles de estilo tarasco por ejemplo en Molino pero también muchos sitios donde se han encontrado objetos con un diseño único (Hosler (2005)). Por lo que es un tema aún abierto en la arqueología, el cual este trabajo busca proporcionar herramientas para la solución. Uno de los objetivos arqueológicos es poder identificar si se pueden apreciar estas relaciones independientes de los grandes centros del occidente o no, con lo cual este trabajo de tesis busca proporcionar una herramienta de análisis.

Se utilizó una base de datos facilitada por el INAH Michoacán en colaboración con la Universidad Estatal de Arizona, en esta base de datos se utilizó la clasificación tipológica creada por (Pendergast (1962)). Pendergast divide su clasificación de la siguiente manera: La primera es la más general donde los objetos son divididos por su uso: objetos utilitarios, objetos ceremoniales y objetos de adorno personal. Dentro de estás categorías existe una división para cada artefacto de tipo designada por números romanos, una división más específica de los subtipos están conformados por letras arábicas mayúsculas y subsubtipos designados por númerales romanos y letras arábicas minúsculas. A pesar de las límitaciones que el mismo autor menciona en su tipología se sigue usando a día de hoy en los estudios metalúrgicos de Mesoamérica.

El análisis de redes es una técnica multidisciplinaria la cual se encarga de estudiar sistemas complejos, los cuales se definen como sistemas donde los miembros tienen conexiones unos con otros y además es difícil obtener su comportamiento colectivo a partir de sus miembros individuales. Estos sistemas pueden representar una gran variedad de fenómenos desde las redes neuronales que tenemos en el cerebro, hasta las infraestructuras de comunicaciones relacionando tanto satélites como computadoras y teléfonos como lo demuestran los artículos de (De et al. (2016)) y (Choi et al. (2021)).

A pesar de que las redes se formen de distintas maneras desde la creación biológica, social o tecnológica para nombrar algunas, su arquitectura suele ser muy similar por lo que se pueden utilizar un conjunto común de herramientas matemáticas para explorar estos sistemas (Barabási and Pósfai (2016)). Estas herramientas matemáticas siguen en desarrollo por eso uno de los objetivos de este trabajo es aportar otra herramienta matemática para el análisis de redes. Una de las razones por las cuales se requieren de nuevas herramientas matemáticas es que paradójicamente aunque las redes se puedan encontrar en cualquier lado como se ha mencionado, este campo de la ciencia surge apenas a finales del siglo XIX. Esto se debe principalmente a dos motivos: el primero es que se requiere de el acceso a la información; hecho posible gracias a las computadoras, y una vez teniendo mapeada con exactitud la red es el poder distinguir que a pesar de sus diversas formaciones comparten cierta universalidad y se puede

generalizar su estudio (Barabási and Pósfai (2016)).

Una red se puede definir como un conjunto de elementos los cuales están conectados por vínculos. Cualquier red puede ser modelada por una estructura de grafos $G(V, E)$ donde G es el grafo, V es el conjunto de vértices o nodos que son elementos del grafo y E son las conexiones entre los elementos (Erciyes (2017)). Dentro del tipo de redes existen unas denominadas redes complejas las cuales son redes que no tienen características topológicas simples. También suelen ser dinámicas es decir que cambian en el tiempo y por consecuencia también su topología. Los modelos de este tipo de redes se pueden clasificar en tres: Redes Clásicas Aleatorias, Redes de Mundos Pequeños y Redes con Escala Libre. Las Redes Clásicas Aleatorias se caracterizan porque cada conexión es dada al azar por una probabilidad para todos los nodos. Las redes de mundos pequeños se caracterizan por tener un promedio de distancias pequeño en comparación con el tamaño de la red usualmente grandes, permitiendo ir de un vértice a otro usando un número pequeño de vértices. Tienen principal relación con el análisis de redes sociales ya que de ahí se origina su término. Un problema con esta única definición es que no permite la creación de clústers por lo que Watts y Strogatz proponen un modelo con diámetro pequeño y creaciones de clústers locales al mismo tiempo, esto se ejemplifica en la imagen 1.1. Las Redes con Escala Libre se caracterizan por poder incluir vértices nuevos a la red y conectándolos a los nodos con altos grados de concentración (Erciyes (2017)).

Las redes sociales son aquellas que consisten de individuos o grupos de personas que tienen alguna relación entre ellos. Para efectos de este trabajo se estudian asentamientos prehispánicos y su relación comercial por medio de los artefactos de metal encontrados en los asentamientos y con un clasificación específica para los artefactos. En las ciencias sociales se utiliza mucho el modelo de Redes de Mundos Pequeños esto se debe a que por las dos propiedades mencionadas pueden representar bien conceptos sociales; uno de los más comunes es la amistad. Sin embargo, también presentan problemas como el asumir homogeneidad en la probabilidades de conexiones, dependencia de sus condiciones iniciales y la evolución temporal suele ser limitada.

En este trabajo se busca presentar una alternativa para la creación de redes com-

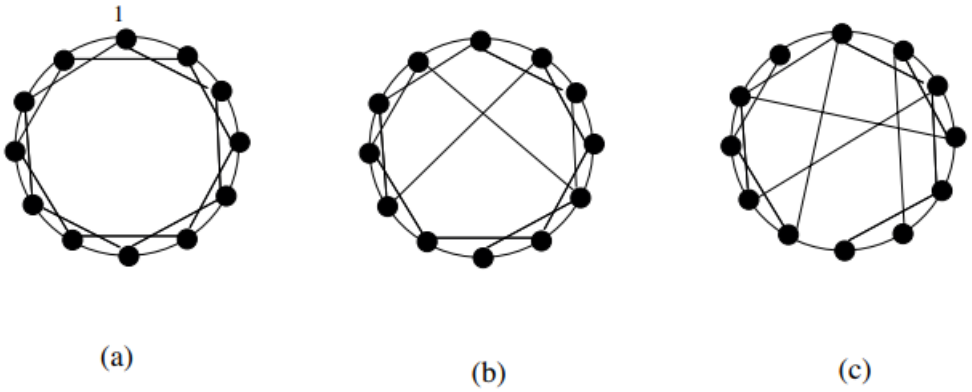

Figura 1.1: Ejemplificación del modelo de Watts y Strogatz. a) Estado inicial siendo una red regular original. b) Después de ciertos reacomodos se preservan los clústers y además en un promedio de distancia pequeño entre los vértices. c) Si se realiza por mucho tiempo este proceso se obtiene una Red Clásica Aleatoria. (Erciyes (2017)).

plejas, en lugar de trabajar al principio con un grafo la red se formará en un espacio de similitud creando ahí el modelo y la simulación. Este proceso está basado en la creación de redes dirigidas por fuerzas. Las redes dirigidas por fuerzas son aquellas redes creadas por dos tipos de fuerzas una de repulsión que separa los nodos del grafo mientras los vértices atraen a los nodos (Jacomy et al. (2014)). Entonces mientras una fuerza trata de separar el sistema otra lo mantiene junto. Estas fuerzas son inspiradas por fuerzas de la vida real como lo muestra (Eades (1984)) donde la fuerza de repulsión está basada en la fuerza de repulsión eléctrica ($F_r = \frac{k}{d^2}$) y la fuerza de atracción en la de los resortes ($F_a = -kd$) donde d es la distancia entre los nodos y k una constante. Las fuerzas físicas no son las únicas que se utilizan para la creación de redes como lo muestra (Fruchterman and Reingold (1991)) donde la fuerza de atracción es $F_a = -\frac{d^2}{k}$ y la de repulsión $F_r = -\frac{k^2}{d}$ esto se debe a que pesar de que simule un sistema físico no es en si mismo uno. Este proceso depende únicamente de las conexiones entre los nodos; sin embargo, también tiene sus puntos malos como lo son la dependencia en el estado inicial del grafo, no es determinista y las coordenadas de cada punto no representan una variable en específico (Jacomy et al. (2014)). Es por esto que en este trabajo se propone un acercamiento distinto donde no van a ser

grafos dinámicos, si no que al terminar la simulación es donde se crea el grafo para visualización.

En el primer capítulo se trata el tema de la base de datos, explicar que datos se usaron y su importancia. En el siguiente capítulo se presenta el método basado en las interacciones de campos físicos como la gravedad o el campo eléctrico este capítulo es el central ya que muestra varios modelos y sus comparaciones, a continuación se explica el porque se utilizan grafos para presentar los resultados y como se obtuvieron estos grafos. Para finalizar con las conclusiones del trabajo.

Capítulo 2

Base de Datos

El suministro de una base de datos para poder realizar las experimentaciones con los distintos método es una parte crucial para el desarrollo y validación de este trabajo. La base de datos sobre la cual se trabaja es un proyecto muy importante que inició en colaboración del INAH Michoacán con la Universidad estatal de Arizona el cual se reitera el agradecimiento. No basta con tener solamente una base de datos, esta debe de ser bien estructurada y confiable para poder poder usarse como punto de partida en cualquier modelo que se requiera. Por esta razón uno de los trabajos que se realizaron de forma no planeada fue la revisión de una parte de esta base de datos y corrección de datos que se hayan encontrado erróneos. El esfuerzo de revisar y corroborar se concentró en los todos aquellos objetos de metal que se encuentran en la base de datos.

En total se revisaron 1,336 entradas de la base de datos de las cuales cada una presentaba 11 características como se observa en la tabla 2.1. Es importante notar que hay algunas características que no se tienen en la mayoría de los objetos esto es por su naturaleza difícil de obtener como es el caso de las columnas de *Earliest* y *Latest* que corresponden a la temporalidad del objeto, algo difícil de obtener sin estudios exhaustivos y pruebas de laboratorio. El $SiteName$ representa el asentamiento donde fue encontrado el objeto, la *Latitude* y *Longitude* son las coordenadas del sitio arqueológico, $MarkerCount$ son la cantidad de objetos de ese registro; es importante mencionar que son objetos encontrados en diversas temporadas de campo

de distintas personas por lo que es común que haya distintos tipos de registros algunos por objeto individual y otros como conjunto del mismo tipo de objeto. Después se tiene $ArtifactTypeName$ que obedece a la clasificación de Pendergast para ese objeto, $UltimateForm$ indica el objeto que es, $Period$ es el periodo histórico al que pertenece el objeto y finalmente $remarks$ son anotaciones del objeto particulares y sus referencia de donde fue localizado.

ID	**SiteName**
Latitude	**Longitude**
MarkerCount	**Earliest**
Latest	**ArtifactTypeName**
UltimateForm	**Period**
Remarks	
1489	Livingstone
33.483888	-114.108333
1	
	IA1a-i
Bell	
Pendergast, 1962	
1329	Homestead Site
32.667859	-113.227509
1	
	IA1a-i
Bell	
Victoria Vargas Tesis	
1426	Pinnacle Peak
33.920103	-112.783421
1	
	IC7a
Bell	
Victoria Vargas Tesis	
353	Gillespie Dam
33.224246	-112.758994
6	1125
1450	IA1a-i
Bell	1,2
Six complete Bells found during previous excavation ...	

Tabla 2.1: Ejemplo de cuatro entradas de la base de datos

2.1. Clasificación de Pendergast

Una de las características más importantes de los objetos y la cual es utilizada en este trabajo es la clasificación de Pendergast. David M. Pendergast es un arqueólogo americano especializado en los estudios de mesoamerica el cual en 1962 presenta una tipología para clasificar los objetos de metal en mesoamérica. También en el artículo presenta un mapa con todos los sitios con objetos de metal en México y Estados Unidos, en este artículo se presentan sitios arqueológicos a pesar de no haber información descriptiva sobre los artefactos de metal en las excavaciones (Pendergast (1962)). Esto es importante ya que existe la posibilidad de un error en el fechado y los artefactos de metal correspondan a otra temporalidad. Por lo mismo si no se encontraba alguna referencia o descripción detallada del objeto no se agregaba a la base de datos.

Como se observa en la imagen 2.1, la distribución de objetos de metal están principalmente concentrados en mesoamerica (Oaxaca principalmente) y en el suroeste de Estados Unidos; esto se debe a varias razones una de ellas es un mayor financiamiento por parte de las autoridades estadounidenses por lo cual era posible realizar mayores trabajos y por el lado mexicano una desatención a los lugares de aridoamerica que es el norte de México. Si se compara con la imagen 2.2, se tienen más sitios actuales; sin embargo, se decidió omitir para sitios del sur de México esto se debe a que se tiene conocimiento de dos corrientes de metalurgia que llegaron a México por un lado la de área maya y por otro lado por el imperio tarasco. Como se está enfocando el estudio en la relación con el suroeste de Estados Unidos se decidió omitir eso sitios para no meter ruido al estudio. Se reconoce la falta de divulgación a otras áreas del conocimiento de las varias herramientas que existen en el área de la computación para mantener una base de datos ordenada, estructurada y sin errores algo sumamente importante.

Respecto a clasificación tipológica de artefactos metálicos Pendergast presenta un esfuerzo para aumentar la claridad al estudiar estos objetos al proponer tres categorías fundamentales: objetos utilitarios, objetos de adorno personal y objetos ceremoniales.

Figura 2.1: Mapa con el contenido de los sitios arqueológicos con objetos de metal en 1962 (Pendergast (1962)).

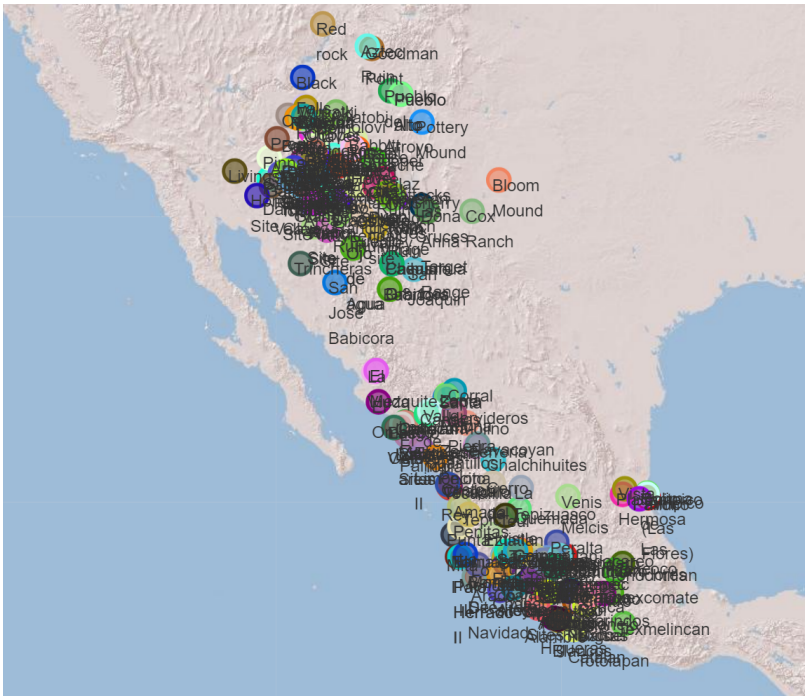

Figura 2.2: Mapa de la base de datos de los sitios arqueológicos con objetos de metal.

La primer categoría divide a los artefactos dependiendo de su uso aparente al ser objetos como hachas y para las otras dos clasificaciones se basa principalmente en formas y otra subdivisión de decoración.

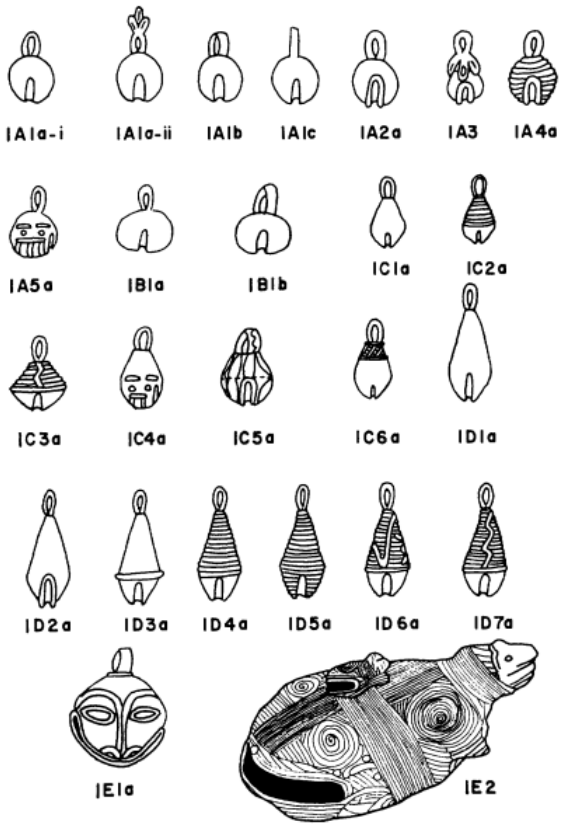

Figura 2.3: Clasificacion de Pendergast para los cascabeles (Pendergast (1962)).

Con la primer separación en conjuntos enumera con números romanos los objetos por lo que se tiene I para representar la agujas siendo el primer objeto utilitario y a su vez I a los cascabeles, primer objeto de adorno personal. Para utilizar las siguientes divisiones se utiliza las letras del alfabeto en mayúsculas donde se diferencia por formas como ejemplo los cascabeles IA e IB donde los IA son cascabeles globulares y IB son globulares del tipo aplanados esta diferencia se aprecia mejor en la figura 2.3. Continúa subdividiendo ahora por decoración utilizando número aravicos, si se requiere continúa dividiendo en formas pero ahora usando letras en minúsculas y así sucesivamente. Se puede apreciar de una mejor manera en la imagen 2.3 que muestra todas las clasificaciones para los cascabeles (Pendergast (1962)), también en la tabla

2.2 como poder pasar de una descripción de un objeto a su clasificación.

Objeto	**Forma**	**Decoración**	**Subforma**	**Tipología**
Cascabel	Globular	Borde dividido	Anillo para suspensión	IA2a

Tabla 2.2: Ejemplo de la descripción de un objeto a su tipología de Pendergast

Los siguientes análisis de los grafos que se realizaron utilizaron únicamente los cascabeles esto debido a su simbolismo ornamental probable que presentaron en aquella época (Hosler (2005)) y también a su distribución y clasificación donde son el tipo de artefacto en los cuales se tienen más registros como se observa en la figura 2.4.

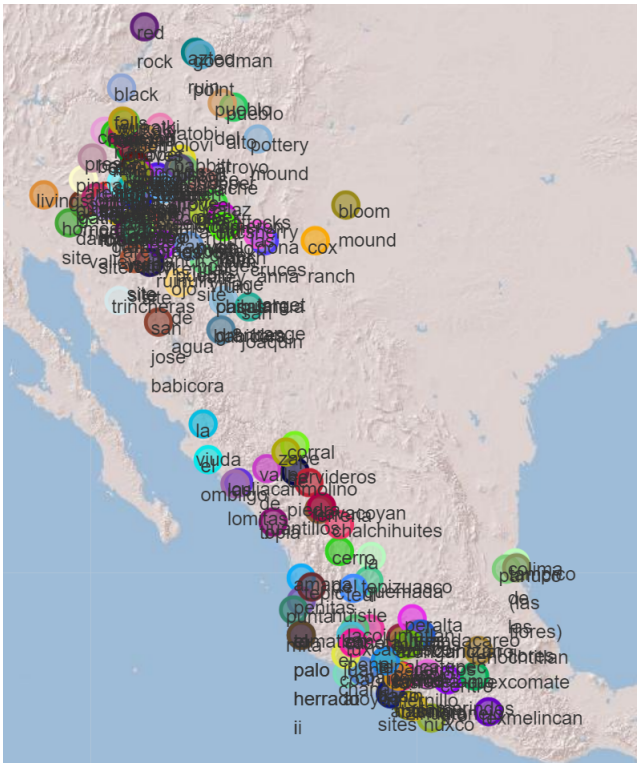

Figura 2.4: Localización de al menos un tipo de cascabel en sitios arqueológicos del base de datos.

Capítulo 3

Método de Fuerzas

3.1. Ejemplo con tres sitios de prueba

Antes de proceder al cálculo utilizando la base de datos de sitios arqueológicos, que se presentó con anterioridad, se realiza un modelo de prueba con tres partículas para verificar que el código funcione y los resultados sean congruentes con lo que se espera de propuesta. Esto se debe en parte a la facilidad de corroborar la solución en un caso menor de elementos para después generalizarla y además que a pesar de ser una solución analítica este análisis, no se tienen referencias de trabajos para las relaciones de estos sitios arqueológicos en específico por lo que la interpretación de los resultados finales y si son factibles o no dependerán de un estudio con un enfoque arqueológico posterior por lo que no presenta utilidad para saber si los cálculos son incorrectos.

Este método de fuerzas propone un análisis de datos inspirado en las fuerzas eléctricas de la física, más específicamente en la electroestática en donde las fuerzas son independientes del tiempo por lo que las partículas sienten atracción o repulsión basadas en la propiedad de la carga y la distancia entre ellas (Jackson (1999)). En este caso se desea que los sitios arqueológicos sientan repulsión o acercamiento basados en la similitud de los mismos, esta propiedad de similitud se calcula entre un par de sitios arqueológicos, y se vayan agrupando o alejando debido a esta fuerza diferenciándose así del caso de la electroestática. Además después de un tiempo se espera que el

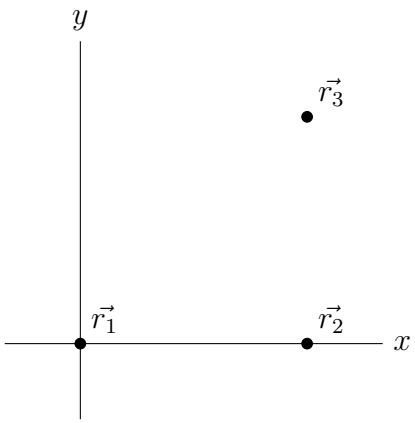

Figura 3.1: Espacio de la simulación a tiempo 0 de las partículas de prueba con una posición incial en (0,0), (1,0) y (1,1).

sistema entre en equilibrio y así presentar una red a los arqueólogos para su posterior análisis.

Para el movimiento de las partículas se basa en la segunda ley de Newton donde cada partícula tendrá fuerzas que accionen sobre sí que determinan su aceleración y por consecuencia el movimiento. Es importante mencionar que a pesar de que se hablen de fuerzas no se esta trabajando en un espacio físico propiamente por lo que no cuenta con unidades físicas como la masa.

Sean $\vec{r_1}$, $\vec{r_2}$ y $\vec{r_3}$ el nombre de las partículas en un espacio de $\mathbb{R}^2$ cada una va a sentir una fuerza por los campos de las demás partículas además de ejercer la suya a las otras partículas. Por lo que la fuerza total en la partícula $\vec{r_1}$ es igual a:

$$\vec{F_1} = \vec{F_{12}} + \vec{F_{13}}. \tag{3.1}$$

Donde $\vec{F_{12}}$ y $\vec{F_{13}}$ son las fuerzas que siente la partícula $\vec{r_1}$ debido a la partícula $\vec{r_2}$ y $\vec{r_3}$ respectivamente, calculadas de la siguiente manera:

$$\vec{F}_{12} = \lambda_{12}\frac{\vec{r}_{12}}{|\vec{r}_{12}|} \tag{3.2}$$

$$\vec{F}_{13} = \lambda_{13}\frac{\vec{r}_{13}}{|\vec{r}_{13}|}. \tag{3.3}$$

Siendo λ_{12} y λ_{13} las constantes calculadas en base a como se toma la similitud de

los sitios arqueológicos, 1, 2 y 1, 3, respecto a los objetos encontrados y $\vec{r}_{12}$, $\vec{r}_{13}$ los vectores de posición que los unen. Los vectores posición son de la forma $\vec{r}_{12}(x, y)$ por lo que no dependen del tiempo. Es importante mencionar que tienen la propiedad que $\vec{r}_{12} = -\vec{r}_{21}$ y $\vec{r}_{13} = -\vec{r}_{31}$. Esta constante λ va a ser muy importante ya que va a dictar el movimiento en la simulación por lo que en el trabajo se utilizaron distintas $\lambda's$ cambiando así los resultados de las simulaciones. Sustituyendo las ecuaciones (3.2) y (3.3) se puede obtener la ecuación para calcular el movimiento de la partícula 1:

$$\vec{F}_1 = \lambda_{12}\frac{\vec{r}_{12}}{|\vec{r}_{12}|} + \lambda_{13}\frac{\vec{r}_{13}}{|\vec{r}_{13}|}. \tag{3.4}$$

Integrando la ecuación (3.4) respecto al tiempo se puede obtener las posición de $\vec{r_1}$, $\vec{r_2}$ y $\vec{r_3}$ en el tiempo t:

$$\vec{r}_1(t) = \lambda_{12}\frac{\vec{r}_{12}}{2|\vec{r}_{12}|}t^2 + \lambda_{13}\frac{\vec{r}_{13}}{2|\vec{r}_{13}|}t^2 + At + B. \tag{3.5}$$

Donde en A y B son las constantes de integración las cuales se obtendrán con las siguientes condiciones iniciales para $t = 0$, $\vec{r}_1(t = 0) = (x_0, y_0)$ y $\vec{v}_1(t = 0) = 0$, donde $\vec{v}_1$ es la velocidad de $\vec{r_1}$. Obteniendo los siguientes valores para A y B

$$A = 0 \tag{3.6}$$

$$B = (x_0, y_0). \tag{3.7}$$

Sustituyendo en la ecuación (3.5) las ecuaciones (3.6) y (3.7) resulta en la ecuación para la posición de la partícula :

$$\vec{r}_1(t) = \lambda_{12}\frac{\vec{r}_{12}}{2|\vec{r}_{12}|}t^2 + \lambda_{13}\frac{\vec{r}_{13}}{2|\vec{r}_{13}|}t^2 + (x_0, y_0). \tag{3.8}$$

Con esto ya se tiene casi todo para poder calcular el ejemplo, sólo hace falta el valor de λ. Esta función que va de $\lambda : \mathfrak{B}^m \rightarrow \{-0.1\} \cup (0, 1]$ donde $\mathfrak{B} = \{0, 1\}$ y m en este caso va a ser la dimensión de los tipos de artefactos de la base de datos. Un ejemplo en el caso de los cascabeles sería IA1a-i,IA1a-ii y así consecuentemente. En

otras palabras de una sucesión de ceros y unos manda a un subespacio de $\mathbb{R}$. Para calcular en este ejemplo la función λ respecto a dos sitios arqueológicos es igual a:

$$\lambda(\vec{v}_1, \vec{v}_2) = \begin{cases} -0.1 & \text{if } \forall i, j : v_1(i) \neq v_2(j) \\ \frac{\sum_{i=1}^{n} \delta_{v_1(i), v_2(i)}}{n} & \text{de otra manera.} \end{cases} \tag{3.9}$$

Donde δ es la delta de Kronecker por lo que $\delta_{v_1(i),v_2(i)}$ es igual a 1 cuando $v_1(i) = v_2(i)$ en cualquier otro caso es 0. El valor de -0.1 representa una fuerza de alejamiento al considerar que son sitios no parecidos, este valor se obtuvo de manera experimental, al estar probando distintos escalares. El otro caso es el de la fuerza de acercamiento el cual si ambos tienen los tipos de objetos para todos los tipos sería de 1.

Al tener todas las ecuaciones ya se puede calcular la posición de los sitios analíticamente en cualquier tiempo y comparar con la solución del algoritmo para así corroborar sus resultados. Tomando el ejemplo de la imagen (3.2) para las condiciones iniciales del problema se puede calcular la posición de las tres partículas a un tiempo t. Tomando como partícula 1 el sitio arqueológico 76 Ranch, como partícula 2 Amapa y como partícula 3 Apache Creek. Al ser un caso de ejemplo se trabajó con la siguiente base de datos también hipotética para calcular la función λ en la cual se tienen tres sitios y tres tipos de cascabeles de ejemplo los IA1a-i, IA1a-ii e IA1b:

SiteName	**IA1a-i**	**IA1a-ii**	**IA1b**
76 ranch	0	0	0
Amapa	0	1	0
Apache creek	1	0	0

Tabla 3.1: Datos de ejemplos para verificar el código para el caso donde se tienen tres sitios para el cálculo de la función λ.

Ahora se puede calcular la función δ, para 76 ranch $v_1 = (0, 0, 0)$, para Amapa $v_2 = (0, 1, 0)$ y para Apache Creek $v_3 = (1, 0, 0)$, es importante recalcar que estos datos no pertenecen a la realidad y son solo para probar el método de fuerzas, en consecuencia el resultado es el mostrado en la siguiente Tabla:

Ahora ya es posible calcular la posición de los sitios en cualquier tiempo con un

SiteName	**76 ranch**	**Amapa**	**Apache creek**
76 ranch	0	-0.1	-0.1
Amapa	-0.1	0	-0.1
Apache creek	-0.1	-0.1	0

Tabla 3.2: Resultados de la función λ utilizando los datos de la tabla (3.1).

simple calculo, se realizó para distintos tiempos el cálculo para el tiempo $t = 1$, esto a su vez nos da los coeficientes para cada tiempo, la posición de los tres sitios es la siguiente:

$$\vec{r}_1(t=1) = -0.1\frac{(-1,0)}{2|1|} + -0.1\frac{(-1,-1)}{2|\sqrt{2}|} + (0,0) = (0.0853, 0.0353), \tag{3.10}$$

$$\vec{r}_2(t=1) = -0.1\frac{(1,0)}{2|1|} + -0.1\frac{(0,-1)}{2|1|} + (1,0) = (0.9500, 0.0500), \tag{3.11}$$

$$\vec{r}_3(t=1) = -0.1\frac{(1,0)}{2|1|} + -0.1\frac{(1,1)}{2|\sqrt{2}|} + (1,1) = (0.0853, 0.0353). \tag{3.12}$$

Al obtener la posición x y y en cualquier momento del tiempo se puede trazar el movimiento de los sitios en este espacio de similitud. Como ambas x y y dependen cuadráticamente del tiempo el movimiento va a ser lineal como se muestra en la figura (3.2) lo cual muestra ser un problema para los objetivos de este trabajo al ser todas soluciones lineales por lo que no se van a crear conjuntos de sitios similares y sitios que se alejen cuando sean únicos. El porqué se comporta de esta manera es debido a que para obtener la ecuación 3.5 se supuso que tanto $\vec{r}_{12}$ como $\vec{r}_{13}$ eran independientes del tiempo;sin embargo, el tomar como dependientes del tiempo resulta en famoso problema de los tres cuerpos que también significa un problema a nuestra solución como se muestra en la siguiente sección.

El sistema de tres cuerpos es un sistema caótico y su generalización para n cuerpos también en el cual las posiciones iniciales del problema si afectan a la solución que se obtiene, como se muestra en las siguientes tres imágenes donde se muestra como un ligero cambio en las posiciones iniciales representa un cambio grande en las soluciones finales. Otra desventaja de usar esta aproximación es el que no se cuenta con una

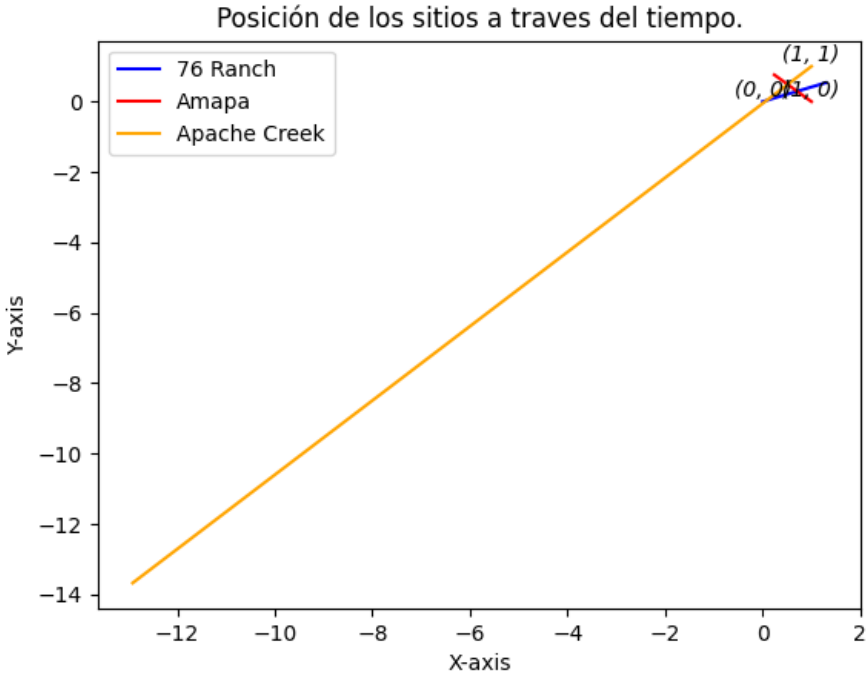

Figura 3.2: Solución exacta con el recorrido de las posiciones a través del tiempo y siendo independientes del tiempo para un tiempo de $t = 0$ a $t = 4$.

solución analítica por lo que se debe utilizarse métodos numéricos para su solución. En el caso de este trabajo se resolvió la ecuación diferencial por medio del método Runge-Kutta esto alarga el tiempo de cálculo, para tres sitios no es tan preocupante pero esto es el conjunto de prueba por lo que puede ser una complicación para los datos verdaderos, como se muestra en la siguiente sección.

Método de Runge-Kutta

A partir de la ecuación 3.4 se puede construir una ecuación diferencial para calcular la posición, por segunda ley de Newton se sabe que $\vec{F} = m\vec{a}$, en este caso la masa no se toma en cuenta. Con esto la ecuación 3.4 se transforma en:

$$\begin{aligned} \vec{a_1} &= \lambda_{12}\frac{\vec{r}_{12}}{|\vec{r}_{12}|} + \lambda_{13}\frac{\vec{r}_{13}}{|\vec{r}_{13}|} \Rightarrow \\ \frac{\partial^2 \vec{r_1}}{\partial t^2} &= \lambda_{12}\frac{\vec{r}_{12}}{|\vec{r}_{12}|} + \lambda_{13}\frac{\vec{r}_{13}}{|\vec{r}_{13}|}. \end{aligned} \tag{3.13}$$

La ecuación 3.13 no tiene solución exacta conocida para casos generales por lo que se deben usar métodos numéricos para resolverla (Jackson (1999)), se puede cambiar para tener un sistema de ecuaciones diferenciales ordinarias sabiendo que $\vec{v} = \frac{d\vec{r}}{dt}$:

$$\frac{\partial \vec{r_1}}{\partial t} = \vec{v_1} \tag{3.14}$$

$$\frac{\partial \vec{v_1}}{\partial dt} = \lambda_{12}\frac{\vec{r}_{12}}{|\vec{r}_{12}|} + \lambda_{13}\frac{\vec{r}_{13}}{|\vec{r}_{13}|}. \tag{3.15}$$

Estás ecuaciones cumplen la siguiente forma $\frac{\partial f}{\partial t} = rhs(x, f, \frac{\partial f}{\partial x}, \frac{\partial f}{\partial x'})$ con lo cual se pueden resolver con el método de líneas. El cual consiste en obtener ecuaciones diferenciales ordinarias para f en cada x_i en el dominio discreto que a su vez se utilizan para construir a f en el tiempo t^{n+1} a partir de f en el tiempo t^n (Press et al. (1992)) y usando el Runge-Kutta. Para este código se utilizó un Runge-Kutta de orden 3 que sigue la siguiente secuencia:

$$k_1 = \Delta t f(t_{i-1}, y_{i-1}) \tag{3.16}$$

$$k_2 = \Delta t f(t_{i-1} + \frac{1}{2}\Delta t, y_{i-1} + \frac{1}{2}k_1) \tag{3.17}$$

$$k_3 = \Delta t f(t_{i-1} + \frac{1}{2}\Delta t, y_{i-1} + \frac{1}{2}k_2) \tag{3.18}$$

$$y_i = y_{i-1} + \frac{1}{6}k_1 + \frac{2}{3}k_2 + \frac{1}{6}k_3. \tag{3.19}$$

Como se ve en las ecuaciones el método Runge-Kutta depende de un salto de tiempo Δt mientras más pequeño sea, más se va a reducir el error pero esto viene con un coste computacional. Los valores de las variables para la prueba son un tiempo de $t = 4$ y un $\delta t = 0.0004$.

En la tabla 3.1 se aprecia cómo ningún sitio arqueológico comparte algún tipo de cascabeles por lo que nuestro objetivo es observar como se alejan al no presentar elementos en común. Este objetivo se ve plasmado en la figura 3.3 como en el espacio de similitud los sitios se alejan entre sí. Ahora para probar el caso donde se acerquen se muestra esta tabla y sus resultados:

En la figura 3.4 se puede observar como los sitios se juntan en una especie de

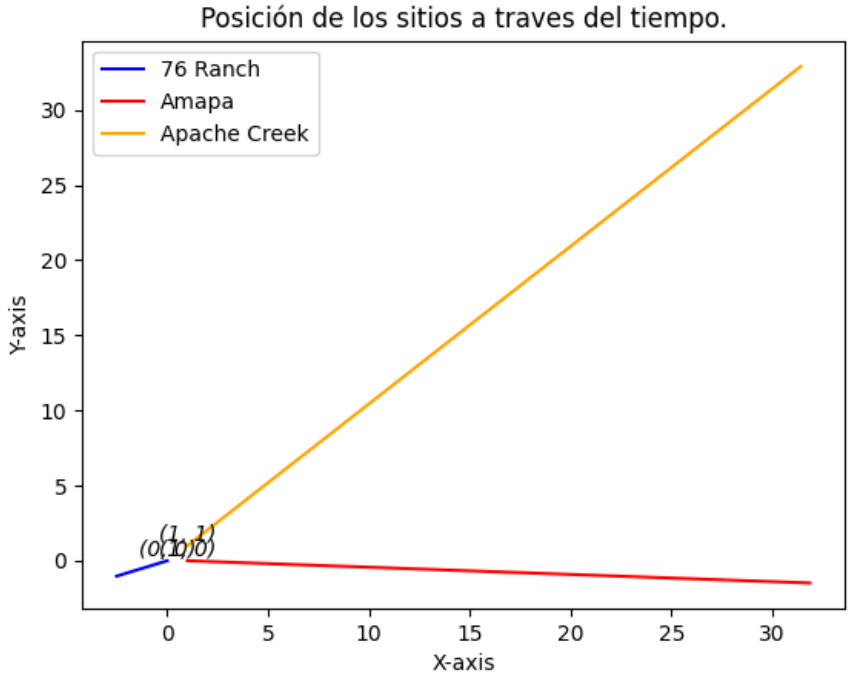

Figura 3.3: Movimiento de los sitios en el espacio de similitud usando el método Runge-Kutta y la tabla 3.1 para la función λ en un tiempo t=4.

SiteName	IA1a-i	IA1a-ii	IA1b
76 ranch	1	1	1
Amapa	1	1	1
Apache creek	1	1	1

Tabla 3.3: Segundo conjunto de datos de ejemplos para verificar el código para el caso donde se tienen tres sitios para el cálculo de la función λ.

entrelazamiento entre ellos y toman una dirección conjunta, esto se explica al tomar en cuenta que la fuerza no es la misma para los sitios arqueológicos porque las distancias no son iguales. También los resultados que importan para este trabajo no es la distancia de los sitios al punto de origen si no la distancia entre ellos por lo que aunque se estén moviendo como se muestra en la figura 3.4 lo que importa es que forman una especie de clúster por lo que son similares para este caso.

Como se observa en la figura 3.5 el sitio de 76 Ranch se aleja de los otros dos sitios y estos a pesar de tener una fuerza de atracción no es lo suficientemente fuerte para que formen un clúster como en la figura 3.4. Pero si se compara con la figura 3.3

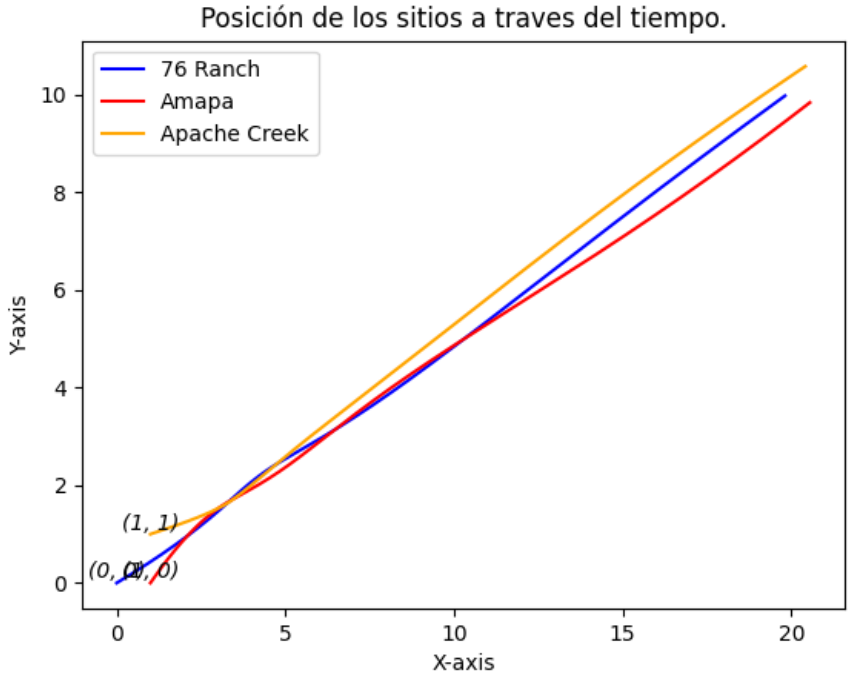

Figura 3.4: Movimiento de los sitios en el espacio de similitud usando el método Runge-Kutta y la tabla 3.3 para la función λ en un tiempo t=4.

se puede observar como el alejamiento no es tan pronunciado, esto se debe a que la fuerza de repulsión entre 76 Ranch fue mayor que la del acercamiento.

Los cálculos para las figuras 3.3, 3.4 y 3.5 fueron obtenidos en un tiempo de 228.9529, 239.2495 y 226.2354 segundos de ejecución respectivamente, lo cual es un tiempo corto de ejecución; sin embargo, se debe tomar en cuenta que fue el caso de prueba con solo 3 sitios y 3 tipos de objetos encontrados. La base de datos del estudio tiene 146 sitios con 41 tipos de cascabeles distintos por lo que el tiempo de ejecución es importante tomarlo en cuenta. Otro de los problemas es que este tipo de sistemas es generalmente caótico lo cual es un problema al tratar de obtener un resultado independiente de las posiciones iniciales del sistema (Křížek (1995)). Por estas dos razones se propone otro método en la siguiente sección.

SiteName	IA1a-i	IA1a-ii	IA1b
76 ranch	1	0	0
Amapa	0	1	1
Apache creek	0	1	1

Tabla 3.4: Tercer conjunto de datos de ejemplos para verificar el código para el caso donde se tienen tres sitios para el cálculo de la función λ. Donde 76 Ranch es único y los otros dos sitios comparten elementos similares.

Método discreto

La alternativa al observar los resultados y tiempo de cálculo fue modificar la ecuación 3.20 haciendo que las posiciones dependan del tiempo anterior y no sean estáticas. Esto es posible al realizar una discretización del tiempo donde el dominio temporal es el siguiente $T = [t_0, t_n]$ donde $t_i = i\Delta t$ para $i = 0, ..., N_t$ enteros y $\Delta t = \frac{t_n - t_0}{N_t}$. Por lo que tenemos la siguiente ecuación para el movimiento de $\vec{r}_1$ a través del tiempo:

$$\vec{r}_1(t_i) = \lambda_{12}\frac{\vec{r}_{12}(t_{i-1})}{2|\vec{r}_{12}(t_{i-1})|}(\Delta t)^2 + \lambda_{13}\frac{\vec{r}_{13}(t_{i-1})}{2|\vec{r}_{13}(t_{i-1})|}(\Delta t)^2 + (x_0, y_0). \tag{3.20}$$

Como se puede observar en la figura 3.6 los sitios presentan el comportamiento esperado al usar la tabla 3.1, donde se están alejando los unos de los otros. Ahora se va a probar con la tabla 3.3, es importante mencionar que para estos cálculos se usaron los mismos parámetros que para el método Runge-Kutta para después comparar su tiempo de ejecución.

En la figura 3.7 se observa cómo los sitios se van acercando pero se empiezan a acercar tan rápido que hacen una especie de efecto resorte. Donde se vuelven a regresar y a pasar, por lo que se puede decir que están en una especie de clúster indicando así su relación de similitud. Como se usó la tabla 3.3 el resultado es el esperado; sin embargo, es importante mencionar que los resultados aunque sean los esperados tanto con este método como con el método Runge-Kutta son distintos, mientras que el Runge-Kutta.

En la figura 3.8 el resultado muestra como a pesar de que los tres sitios se alejan,

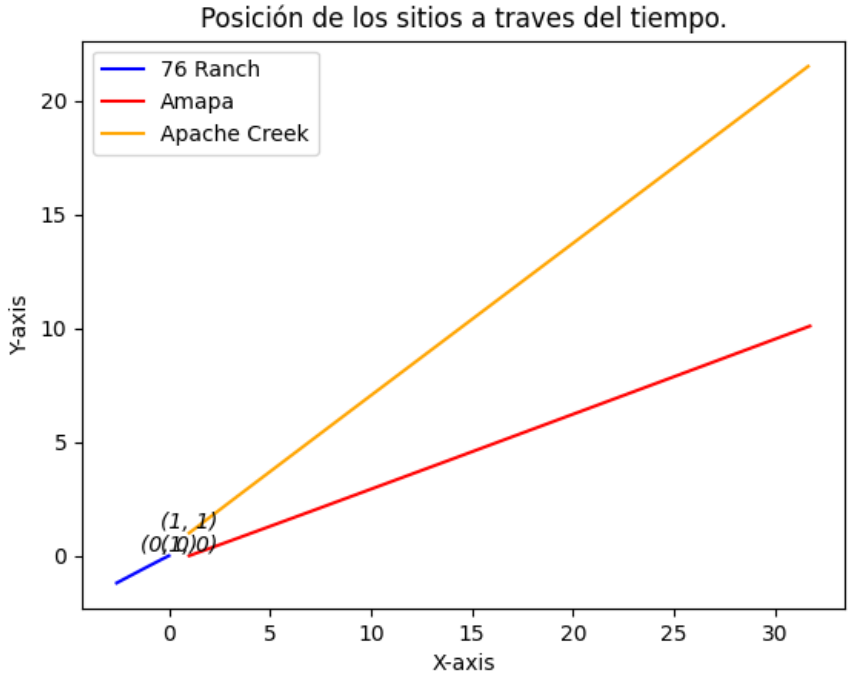

Figura 3.5: Movimiento de los sitios en el espacio de similitud usando el método Runge-Kutta y la tabla 3.4 para la función λ en un tiempo t=4.

a diferencia de la figura 3.6 Amapa y Apache Creed no se están alejando tan drásticamente. Esto muestra un resultado similar al obtenido con el método Runge-Kutta para la misma tabla.

Los cálculos para las figuras 3.6, 3.7 y 3.8 se obtuvieron en un tiempo de ejecución del código de 67.3032, 67.5929 y 65.7481 segundos respectivamente. Lo cual representa una disminución considerable de tiempo comparado con el Runge-Kutta tomando en cuenta la base de datos y junto con las acciones de las simulaciones similares propongo que es un buen sustituto al método físico usando de ventaja que es una simulación no física. A pesar de esto se va a mostrar un resultado con el método Runge-Kutta en la siguiente sección para comparar mejor los métodos.

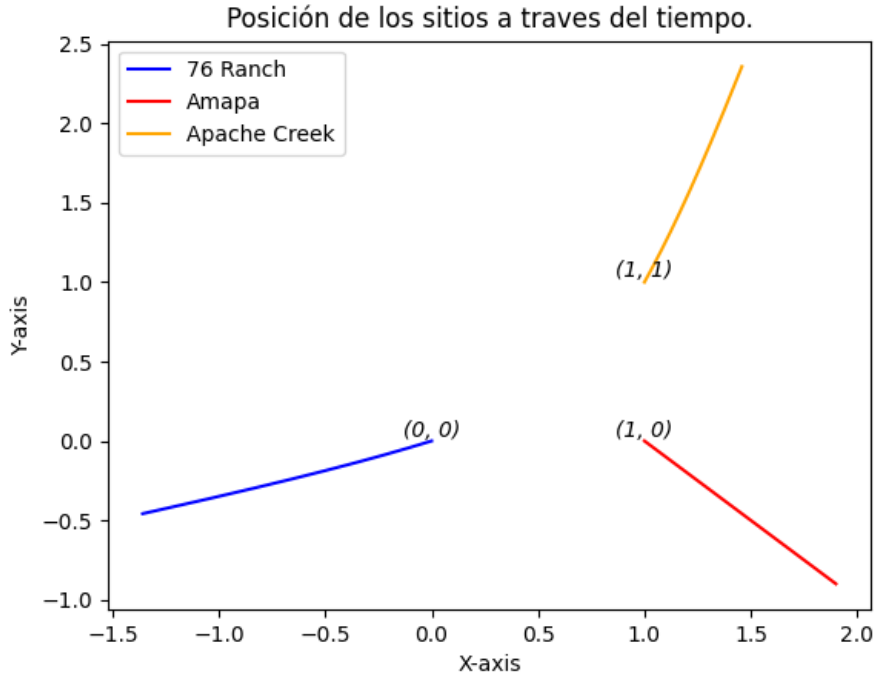

Figura 3.6: Movimiento de los sitios en el espacio de similitud usando el método de evolución creado para este trabajo y la tabla 3.1 para la función λ en un tiempo t=4.

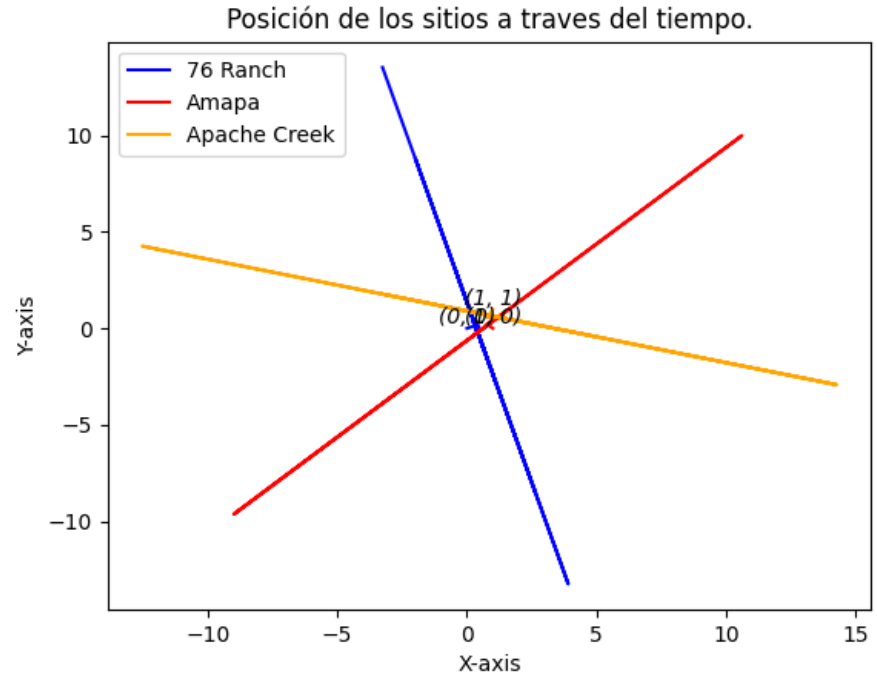

Figura 3.7: Movimiento de los sitios en el espacio de similitud usando el método de evolución creado para este trabajo y la tabla 3.3 para la función λ en un tiempo t=4.

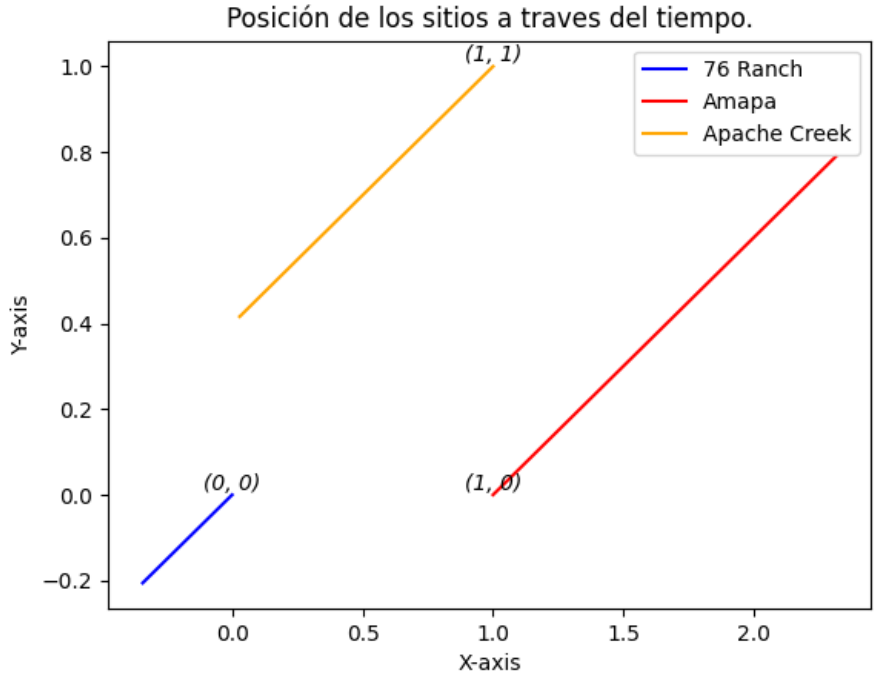

Figura 3.8: Movimiento de los sitios en el espacio de similitud usando el método de evolución creado para este trabajo y la tabla 3.4 para la función λ en un tiempo t=4.

3.1.1. Caso general

Una vez probado que el código funciona correctamente para el caso donde se tienen 3 partículas se presenta ahora la generalización para n número de partículas. Sea un conjunto de partículas $\vec{r_1}$, $\vec{r_2}$, ..., $\vec{r_n}$ en un espacio de $\mathbb{R}^2$ se va a calcular la fuerza para una partícula i debido al resto de las partículas. Como en el caso de las tres partículas las fuerzas que actúan son solamente aquellas debido al resto de partículas. Por lo que la ecuación 3.1 se generaliza como:

$$\vec{F_i} = \vec{F}_{12} + \vec{F}_{13} + ... + \vec{F}_{i,i-1} + \vec{F}_{i,i+1} + ... + \vec{F}_{in}. \tag{3.21}$$

De forma similar los valores de las fuerzas individuales son a como se muestran para 3.2 y 3.3, por lo que la ecuación 3.21 queda de la siguiente manera para un sitio i:

$$\vec{F_i} = \lambda_{i1}\frac{\vec{r}_{i1}}{|\vec{r}_{i1}|} + \lambda_{i2}\frac{\vec{r}_{i2}}{|\vec{r}_{i2}|} + ... + \lambda_{in}\frac{\vec{r}_{in}}{|\vec{r}_{in}|}. \tag{3.22}$$

Método Runge-Kutta

Como en el caso de las tres partículas se puede construir una ecuación diferencial para la posición usando la segunda ley de Newton. Por lo que sustituyendo en la ecuación 3.22 obtenemos la ecuación para la posición de un sitio i:

$$\begin{aligned} \vec{a_i} &= \lambda_{i1}\frac{\vec{r}_{i1}}{|\vec{r}_{i1}|} + \lambda_{i2}\frac{\vec{r}_{i2}}{|\vec{r}_{i2}|} + ... + \lambda_{in}\frac{\vec{r}_{in}}{|\vec{r}_{in}|} \Rightarrow \\ \frac{\partial^2\vec{r}_i}{\partial^2 t} &= \lambda_{i1}\frac{\vec{r}_{i1}}{|\vec{r}_{i1}|} + \lambda_{i2}\frac{\vec{r}_{i2}}{|\vec{r}_{i2}|} + ... + \lambda_{in}\frac{\vec{r}_{in}}{|\vec{r}_{in}|}. \end{aligned} \tag{3.23}$$

Como en el caso de tres cuerpos esta ecuación no tiene solución exacta conocida para casos generales por lo que se procede de la misma forma usando el método de lineas junto con el Runge-Kutta para obtener una solución numérica. Por lo que se reescribe la ecuación 3.25:

$$\begin{aligned} \frac{\partial\vec{r}_i}{\partial t} &= \vec{v}_i \\ \frac{\partial\vec{v}_i}{\partial t} &= \lambda_{i1}\frac{\vec{r}_{i1}}{|\vec{r}_{i1}|} + \lambda_{i2}\frac{\vec{r}_{i2}}{|\vec{r}_{i2}|} + ... + \lambda_{in}\frac{\vec{r}_{in}}{|\vec{r}_{in}|}. \end{aligned} \tag{3.24}$$

Estas dos ecuaciones cumplen con la forma $\frac{\partial f}{\partial t} = rhs(x, f, \frac{\partial f}{\partial x}, \frac{\partial f}{\partial x'})$ con lo que se pueden solucionar por el método de lineas con el método Runge-Kutta. Uno de los objetivos de proponer un nuevo método para crear redes sociales era que no dependiera de la posición inicial de los sitios arqueológicos por lo tanto cada sitio se inicializó de forma aleatoria en un cuadrado de 200X200 con centro en el $(0, 0)$ como se ejemplifica en la figura 3.9.

Esta forma de inicializar los sitios presenta un problema para este método ya que las ecuaciones 3.23 es un conjunto no lineal de ecuaciones diferenciales parciales de segundo orden por lo que inducen una naturaleza caótica al sistema (Trenti and Hut (2008)). En la siguientes imágenes se muestra la posición inicial de los sitios arqueológicos y su posterior evolución usando este método. Se utilizan dos posiciones iniciales distintas para ver el efecto de que sean distintas donde se espera que por ser un sistema caótico cambien los clústers de la simulación.

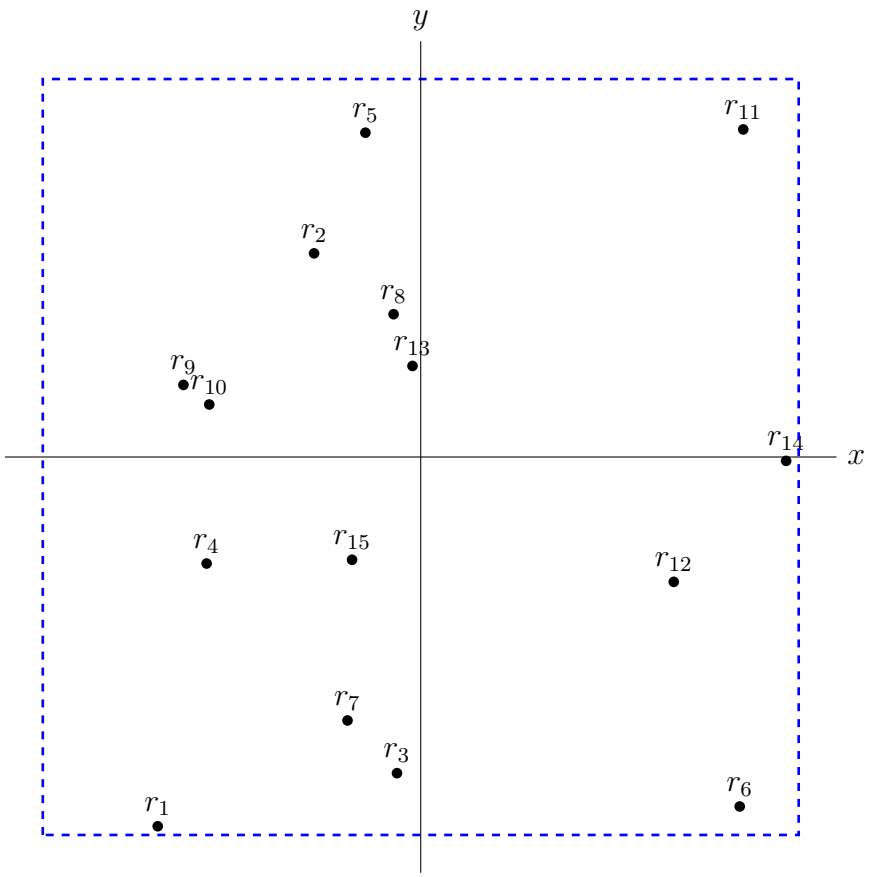

Figura 3.9: Para todas las simulaciones se utilizó esta forma de inicialización de los datos. Poner los sitios al azar en un cuadrado centrado en el (0,0)

Es importante mencionar que la ecuación 3.22 presenta una indeterminación cuando la distancia entre cualquiera dos sitios se vuelve cero. Por lo que aunque sean al azar las condiciones iniciales del problema se puso una restricción que nunca dos sitios ocuparan el mismo lugar. El efecto de la indeterminación no sólo se queda ahí, también por la literatura se sabe que si se acercan demasiado las partículas por la singularidad pueden presentar a velocidades relativas arbitrariamente grandes (Trenti and Hut (2008)). Esto se trato de varias maneras, primero los sitios no presentan colisiones entre ellos y segundo cuando se acerca demasiado un sitio a otro toda fuerza de acercamiento se vuelve cero con lo que únicamente queda una fuerza de repulsión. Estas condiciones no sólo se tomaron para este método si no de misma manera para todos los métodos de evolución.

Para la simulación con condiciones iniciales en la figura 3.10 y los resultados de

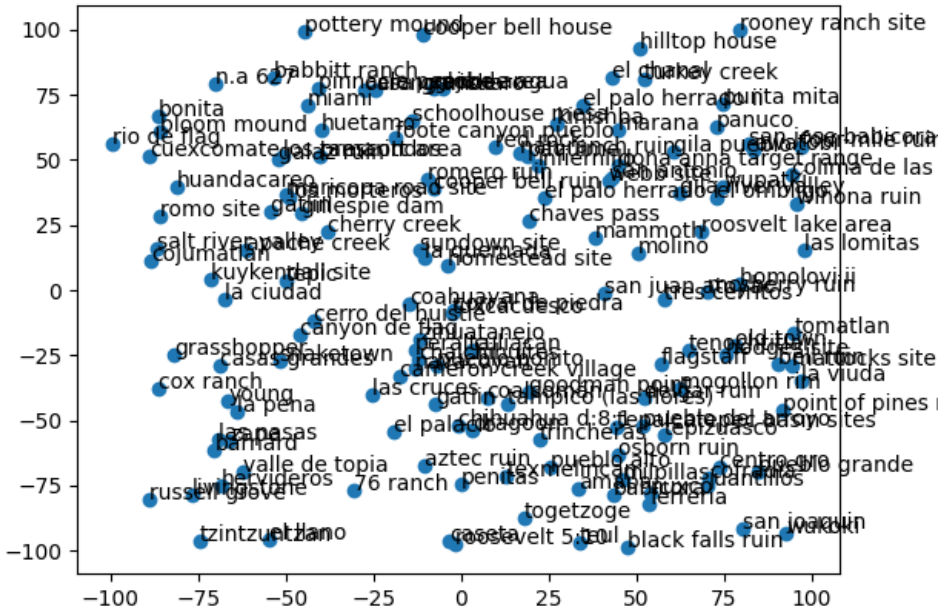

Figura 3.10: Estado inicial aleatorio de los sitios para la simulación en el cuadrado de 200 X 200 con los nombre de los sitios arqueológicos.

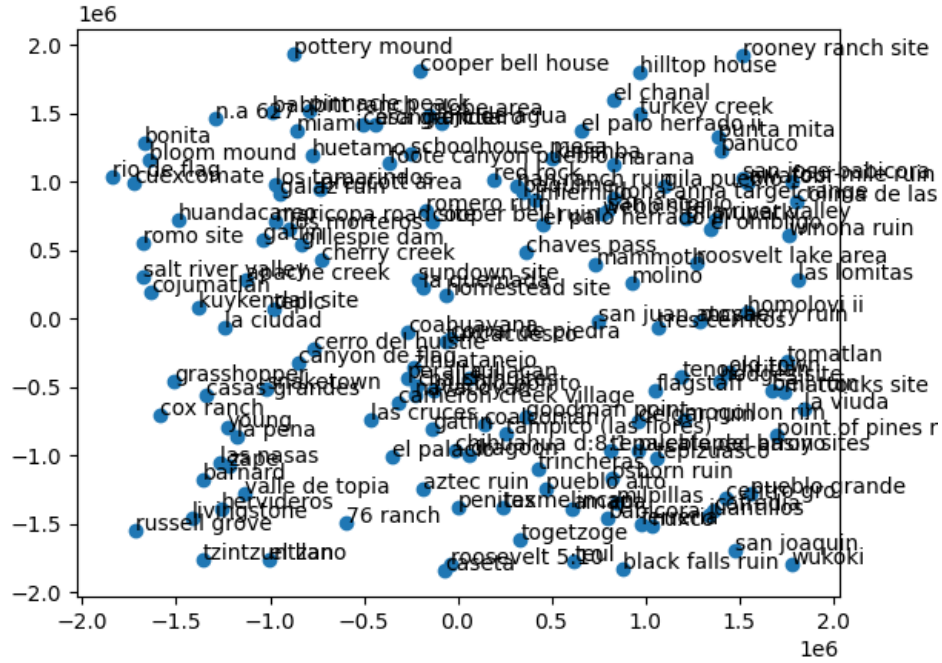

Figura 3.11: Estado final de la simulación después de un tiempo $t = 100$ y un valor $dt = \frac{100}{1000} = 0.1$ es decir se realizaron 1000 pasos temporales.

la figura 3.11 se corrió durante un tiempo de 77904.6792 segundos que es aproximadamente veintidós horas para 145 sitios donde hay cascabeles y con 1000 pasos temporales, lo cual es un tiempo grande para seguir haciendo pruebas esto se debe a dos cosas, en primera el método Runge-Kutta requiere un valor pequeño de dt para que el valor del error no sea grande por lo que se tienen que usar varios pasos temporales y además requiere tres iteraciones para su cálculo como se muestran en las ecuaciones de la sección anterior. Estos detalles se puede solucionar usando usando un integrador de Hermite (Yamamoto and Makino (2018)); sin embargo, para este trabajo no se realiza esa implementación por el tiempo que requiere la implementación. Se puede observar también al comparar ambas imágenes que las estructuras se mantienen y se van alejando una de otra de forma constante obteniendo la misma configuración inicial solo que a escalas de distancias mayores. Esto es un detalle importante para el objetivo de este trabajo puesto que el resultado es pésimo porque no representa ni siquiera los nodos comerciales ya conocidos y evidentes, mucho menos nuevos nodos posibles.

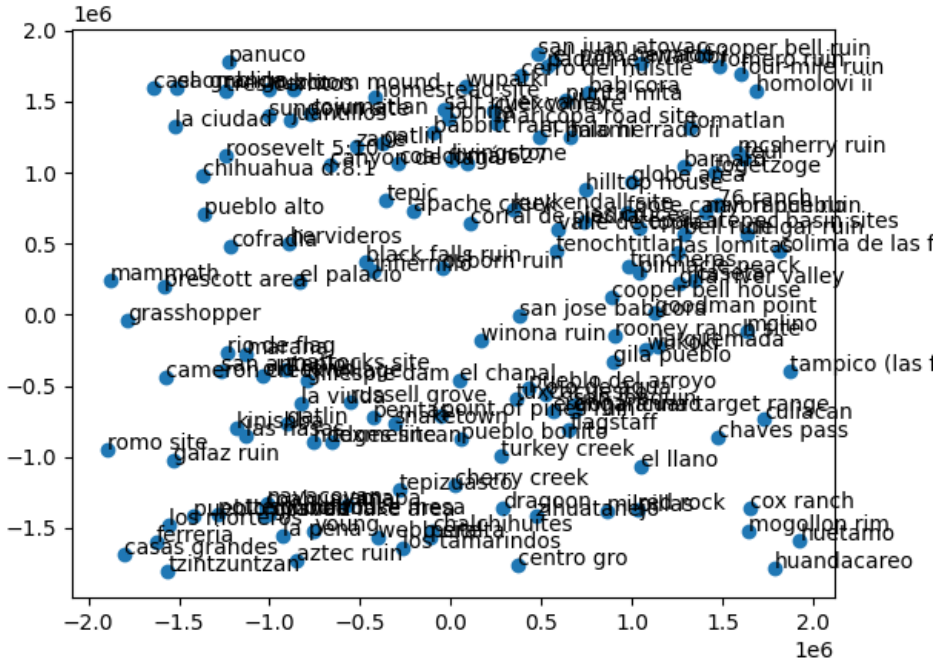

Figura 3.13: Estado final de la segunda simulación después de un tiempo $t = 100$ y un valor $dt = \frac{100}{1000} = 0.1$ es decir se realizaron 1000 pasos temporales.

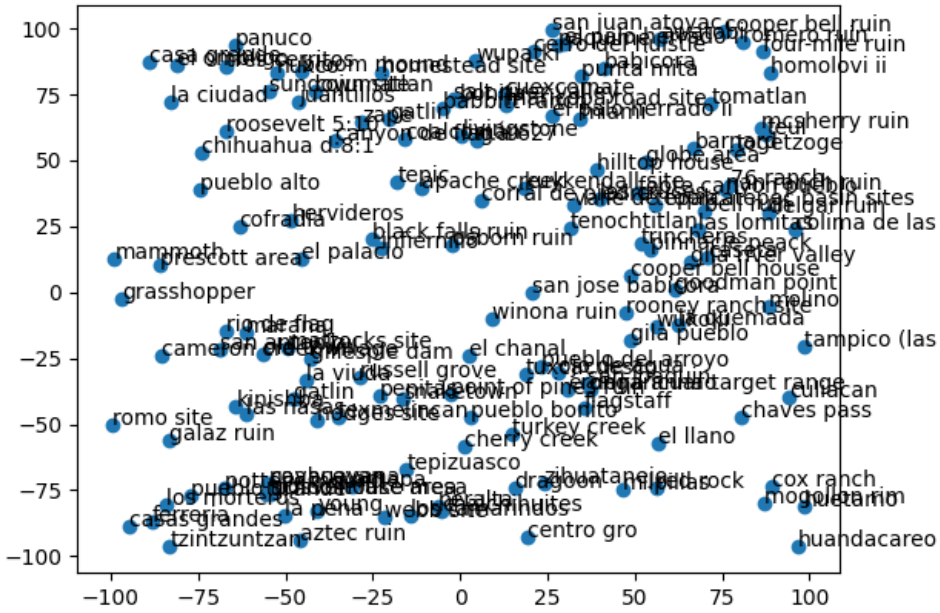

Figura 3.12: Estado inicial aleatorio de los sitios para la segunda simulación en el cuadrado de 200 X 200 con los nombre de los sitios arqueológicos.

Se vuelve a correr una simulación para observar si se creaban agrupaciones de sitios similares; sin embargo, como se observa en la figura 3.13 no se crea ninguna estructura. Se observa cómo en la primera simulación como los sitios se alejaron los unos de otros de manera constante resultando en la misma figura 3.12 pero a una escala de separación mayor. En este trabajo se propone que este alejamiento constante se debe a la función λ en específico ya que el valor que se propone de -0.1 para la fuerza de alejamiento es empírico obtenido de las simulaciones del método discreto. Entonces una hipótesis es que es tan grande el valor que la influencia de repulsión que no se pueden formar estructuras. Otra propuesta es que no se dejó correr la simulación a tiempos muy altos mayores a 100 por dos razones: en la primer simulación si se pueden apreciar ligeros cambios en contraste de las posiciones iniciales; sin embargo, la escala es muy grande impidiendo que se formaran grupos y porque para que el error sea pequeño si se aumenta el valor de t se tienen que aumentar los pasos temporales siendo un costo computacional mayor que si se observa la figura 3.5

ya es muy grande para este número de sitios arqueológicos. Esto contradice con el objetivo del trabajo donde lo mínimo sería poder replicar las conexiones ya conocidas como lo son la importancia de Tzintzuntzan o Paquime como centros de comercio importantes y se tendrían que ver rodeados de otros sitios que comparten tipos de objetos por lo que para el resto de las pruebas se utiliza el llamado método discreto explicado en la siguiente sección.

Método discreto

Para el método discreto el proceso para conseguir el valor de la posición en el espacio de similitud es muy similar al que se hizo en la ecuación 3.5, dando como resultado para un número N de puntos la siguiente ecuación:

$$\vec{r_i}(t_j) = \lambda_{i1}\frac{\vec{r}_{i1}(t_{j-1})}{|\vec{r}_{i1}(t_{j-1})|}\Delta t^2 + \lambda_{i2}\frac{\vec{r}_{i2}(t_{j-1})}{|\vec{r}_{i2}(t_{j-1})|}\Delta t^2 + ... + \lambda_{in}\frac{\vec{r}_{in}(t_{j-1})}{|\vec{r}_{in}(t_{j-1})|}\Delta t^2 + (x_0 + y_0). \tag{3.25}$$

Donde no se tiene que integrar ninguna ecuación por lo que el cálculo es más rápido que el método Runge-Kutta. De la misma manera que para el método Runge-Kutta se va a probar con dos condiciones iniciales distintas la primer condición inicial se ve en la figura 3.14 y como en los otros casos se iniciaron los sitios al azar en un cuadrado de 200X200. Se utilizan los mismos parámetros temporales es decir un tiempo $t = 100$, un valor $dt = \frac{100}{1000} = 0.1$ y con 1000 pasos temporales y la misma base de datos para cascabeles de metal que contiene la información en 145 sitios.

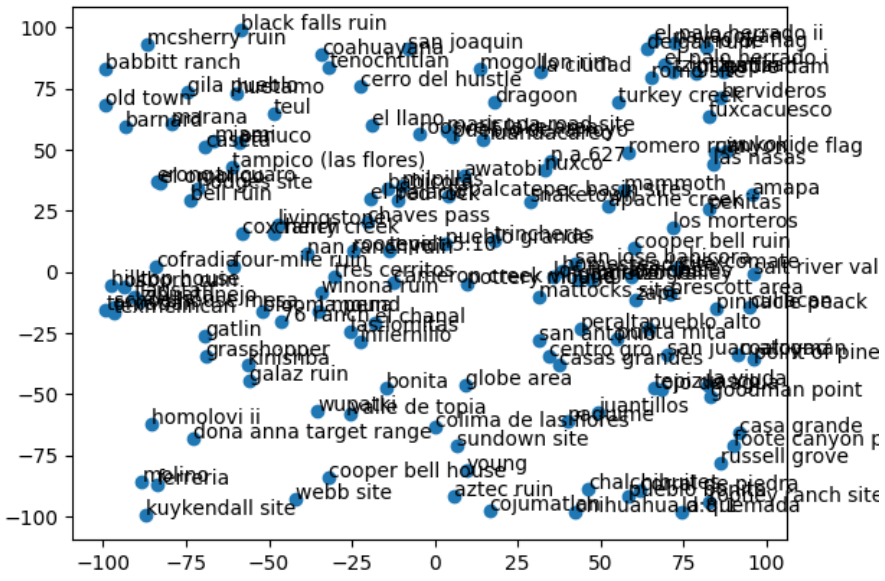

Figura 3.14: Estado inicial aleatorio de los sitios para la simulación en el cuadrado de 200 X 200 con los nombre de los sitios arqueológicos para la función λ y el método discreto.

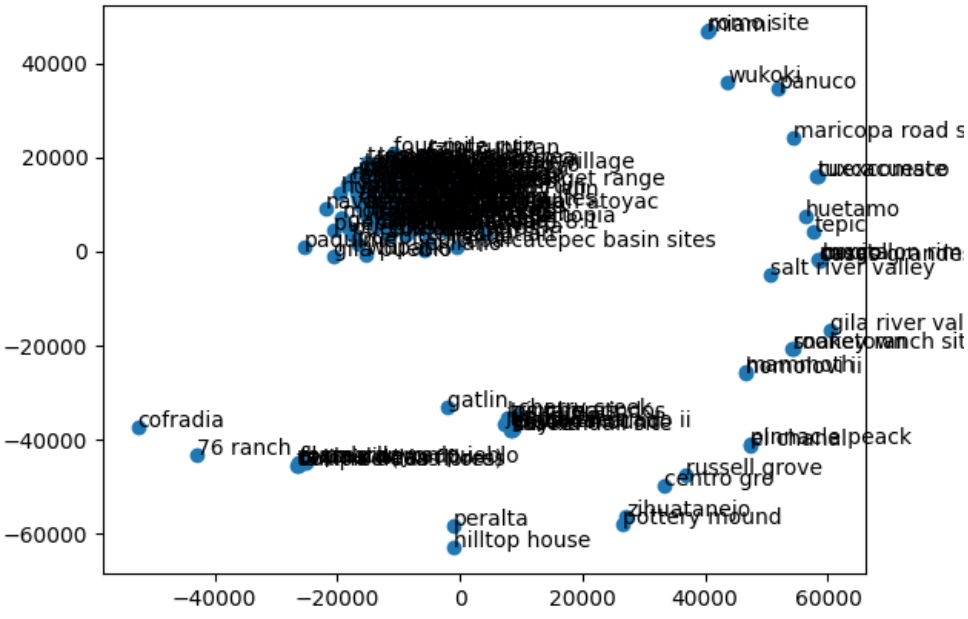

Figura 3.15: Estado final de la simulación igual que para el Runge-Kutta los parámetros son un tiempo $t = 100$ y un valor $dt = \frac{100}{1000} = 0.1$ es decir se realizaron 1000 pasos temporales.

Para la imagen 3.15 el tiempo de ejecución fue de 21170.79 segundos que en horas son 5.88 teniendo una mejora considerable al método Runge-Kutta como se observa en la tabla 3.5. Esto se debe al número de iteraciones que el mismo método Runge-Kutta aplica para ser de tercer orden y tener un error pequeño, otra característica importante para los objetivos de este proyecto es la formación clara de clústers se puede notar un clúster central y otros pequeños, mientras que unos sitios se alejan completamente de todos dando una estructura clara. Sobre el clúster grande se debería esperar que sean las relaciones comerciales ya conocidas y fuertes para confirmar que le método esta simulando lo que se conoce actualmente, lo interesante son los clústers pequeños donde el método busca relaciones comerciales únicas que no se tuvieran mapeadas aún.

A su vez los sitios arqueológicos que se alejan se esperaría que sean sitios con características de cascabeles ya sean únicas es decir que el sitio presente tipos únicos de cascabeles o muy poca diversidad de cascabeles. Un indicativo bueno del método es que en los sitios que se alejan individualmente no se encuentran sitios como Tzintzuntzan (la capital Tarasca de aquella época) o Paquimé dos ciudades que se saben que tuvieron gran importancia en el comercio y en el caso de Tzintzuntzan era la capital del imperio tarasco el principal productor de metal en aquella época. Estos resultados preliminares indican a primera mano un buen comportamiento del modelo y de la función λ; sin embargo, el análisis no es tan claro dada la figura **??** y además es importante la interpretación arqueológica posterior no solo para indicar si concuerda con la realidad si no que también brinde posibles nuevos resultados lo cual es el objetivo de este trabajo.

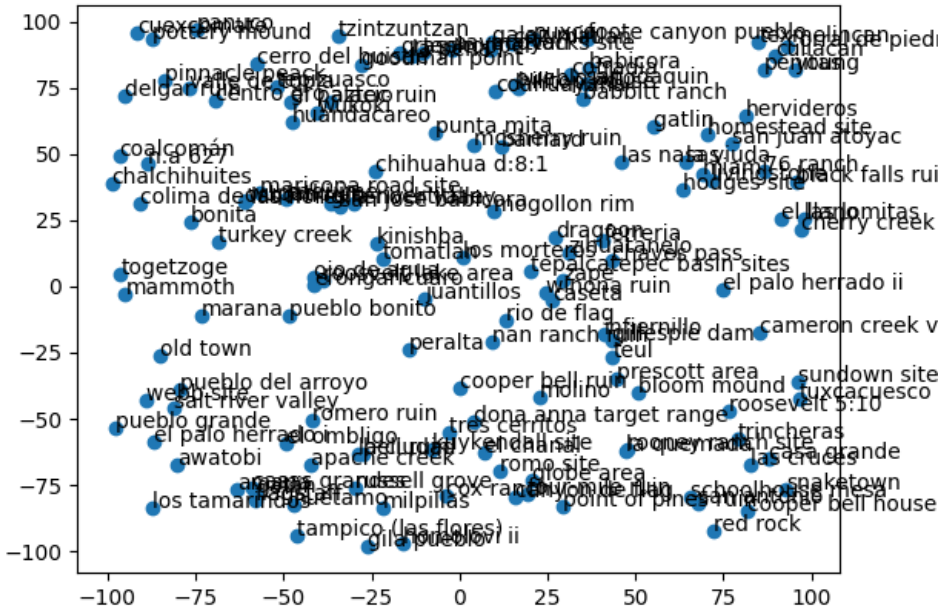

Figura 3.16: Estado inicial aleatorio de los sitios para la simulación en el cuadrado de 200 X 200 con los nombre de los sitios arqueológicos para la función λ y el método discreto.

Comparando la imagen 3.14 con la imagen 3.16 se observa claramente que son configuraciones distintas para el estado inicial de ambas simulaciones aunque ambas están limitadas a las mismas condiciones el cuadrado de $200x200$ y que ningún sitio pueda ocupar el mismo espacio que otro. Esto es con el objetivo de esclarecer si el método depende fuertemente del estado inicial de las configuraciones o no y en caso de que la respuesta sea no entonces el resultado de la simulación final deber ser similar al de la imagen 3.15. También sirve como punto de prueba para este método discreto.

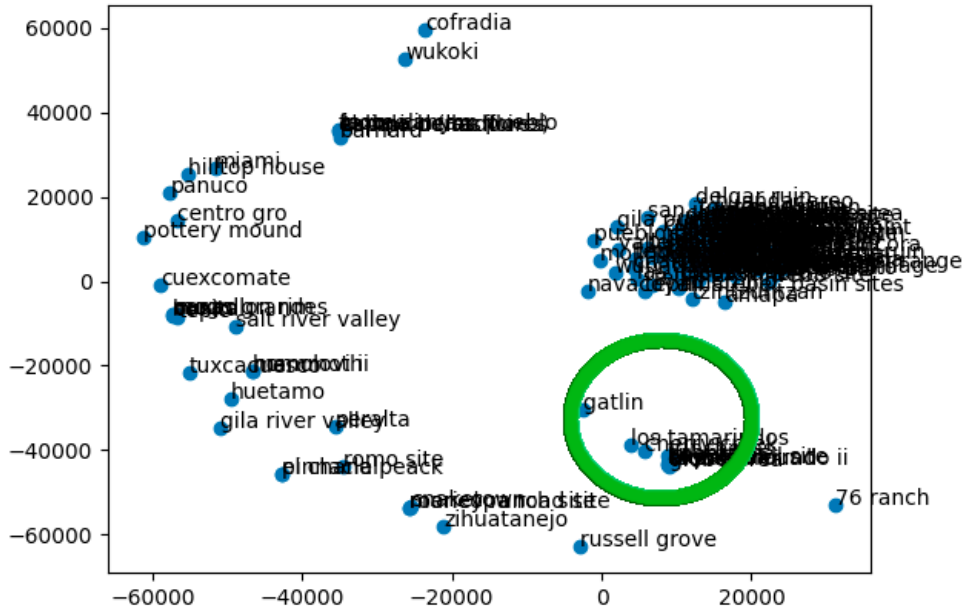

Figura 3.17: Estado final de la simulación de la figura 3.16 igual que para el Runge-Kutta los parámetros son un tiempo $t = 100$ y un valor $dt = \frac{100}{1000} = 0.1$ es decir se realizaron 1000 pasos temporales.

Los resultados para esta simulación se obtuvieron en un tiempo de 21420.79 segundos que son 5.95 horas. Una comparación del tiempo de ejecución entre los métodos se puede percibir mejor en la table 3.5, donde el método Runge-Kutta es muy lento para poder realizar varias simulaciones. En la figura 3.17 se nota la estructura similar inclusive las mismas escalas para las separaciones de los sitios como en la figura 3.15. También si se observa más a detalle el círculo verde que resalta ese clúster pequeño es muy similar al clúster de Gatlin en la primer simulación con sitios arqueológicos en común de los resultados de la primer simulación. Con esto se puede decir tres cosas, el método presenta una clara mejoría en cuanto al tiempo, esto es evidente ya que no usa tantas reiteraciones como el Runge-Kutta, también es mejor para crear estructuras de clústers; sin embargo, esto se puede deber a los parámetros de la fuerza de alejamiento de la función λ donde para el método Runge-Kutta es suficientemente grande tal que no permite la creación de clústers y finalmente que no depende fuertemente de las posiciones iniciales uno de los objetivos principales de este trabajo. Con

estos resultados se procede a experimentar con la función λ al cambiarla y observar las simulaciones finales que se obtienen.

Número de Sitios Arqueológicos	**Método Runge-Kutta**		**Método discreto**	
	Tiempo (s)	**Tiempo (hr)**	**Tiempo (s)**	**Tiempo (hr)**
3	231.47	0.06	66.88	0.01
145	77,904.67	21.64	21,420.70	5.9

Tabla 3.5: Comparación de ambos métodos en relación al tiempo de ejecución usando un procesador AMD Ryzen 9 5900X y los números de sitios arqueológicos usados en las simulaciones.

Capítulo 4

Visualización

La visualización de resultados de forma clara y concisa es un paso importante en todo trabajo de investigación. Una buena visualización permite no malinterpretar los datos, mostrar efectivamente el trabajo realizado y simplificar para facilitar su interpretación. En el capítulo 3 se presentan los resultados de distintos métodos;sin embargo, esta representación no es para nada clara y a pesar de que se hayan podido realizar ciertos análisis al observar los grupos que se forman no queda nada claro por lo que es evidente la falla en la visualización como lo que pasa en la figura 3.17.

Uno de los objetivos de este trabajo es presentar una nueva herramienta para el análisis de sitios arqueológicos por lo que el tema de la visualización de los resultados es fundamental al ser un trabajo multidisciplinario. Esta nueva visualización busca mezclar el sistema dinámico para la similitud de sitios junto a la facilidad de interpretación de las redes en un mapa geográfico, es natural pensar en mapas de países al tener las variables correctas de latitud y longitud de los sitios arqueológicos. Por lo que los sitios arqueológicos estarán en su lugar geográfico mientras que los resultados obtenidos por el método se verán reflejados en el número de sitios que aparecen en el mapa. Obteniendo así una representación más amigable para un posterior análisis. Para lograr esto una vez se termina la simulación se guarda las distancias del espacio de similitud entre cada sitio. A mayor distancia este un sitio arqueológico de otro se espera que sean menos similares indicando así que no estaban conectados directamente en una red comercial.

El algoritmo crea un mapa de similitud para un sitio arqueológico y el resultado de la simulación para alguna función λ. Esto debido a que las figuras del capitulo anterior buscan mostrar la similitud de todos los sitios con todos los otros sitios y esto complica la visualización correcta de resultados. Una vez dado el sitio arqueológico el algoritmo selecciona el 10 % de otros sitios arqueológicos más similares esto lo hace al medir la distancia del sitio seleccionado a todos los demás y eligiendo aquellas que entran en el 10 % más pequeñas. La fórmula de la distancia es la clásica euclidiana dada por:

$$d(S_1, S_2) = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}, \tag{4.1}$$

donde S_1 y S_2 son los sitios arqueológicos. Después para cada uno de estos sitios se calcula la diversidad del sitio arqueológico es decir se cuenta los números de distintos tipos de cascabeles esta información es usada para calcular el peso de los nodos. A continuación se realiza el mismo proceso pero para los sitios relacionados. Por lo que se obtienen dos conjuntos, que se incluyen en un mapa interactivo de México y Estados Unidos. En dicho mapa se puede acercar, alejar la imagen también seleccionar para obtener el nombre del sitio, donde el sitio a observar esta coloreado de rojo, sus conexiones en morado y el resto de las conexiones en color verde como se observa en la imagen 4. Además los números de los sitios cuando se seleccionan con el cursor indican su similitud en cuanto al sitio de estudio que en este caso es Tzintzuntzan, siendo el 1 reservado para el sitio en cuestión el 2 el sitio más similar y así sucesivamente. También se realizó otra propuesta para el orden de similitud entre los sitios en el cual se tomo en cuenta la distancia real entre los sitios como un peso además de los resultados de la simulación, se utiliza la fórmula de Harvesine que supone a la Tierra como una esfera (Mahmoud and Akkari (2016)) dada por la siguiente ecuación:

$$d = 2r \arcsin \sqrt{\sin^2\left(\frac{\theta_2 - \theta_1}{2}\right) + \cos(\theta_1)\cos(\theta_2)\sin^2\left(\frac{\lambda_2 - \lambda_1}{2}\right)}, \tag{4.2}$$

donde r es el radio de la Tierra, θ y λ son la latitud y longitud respectivamente. Después se va iterando sobre la lista ordenada de sitio comparando el sitio i con el

sitio $i + 1$ donde si la distancia de i al sitio principal es mayor a 100 km y la de $i + 1$ no se intercambian y se procede al índice $i + 2$. Se toma la decisión de este proceso por dos razones el principal objetivo de tomar en cuenta la distancia física la cual representaba un reto en aquellos tiempos donde no contaban con medios de transporte rápidos o eficaces;sin embargo, no darle tampoco tanto peso a la distancia física y evitar casos hipotéticos donde un sitio puede que fuera el más similar a otro y después de tomar en cuenta la distancia fuera el menos similar.

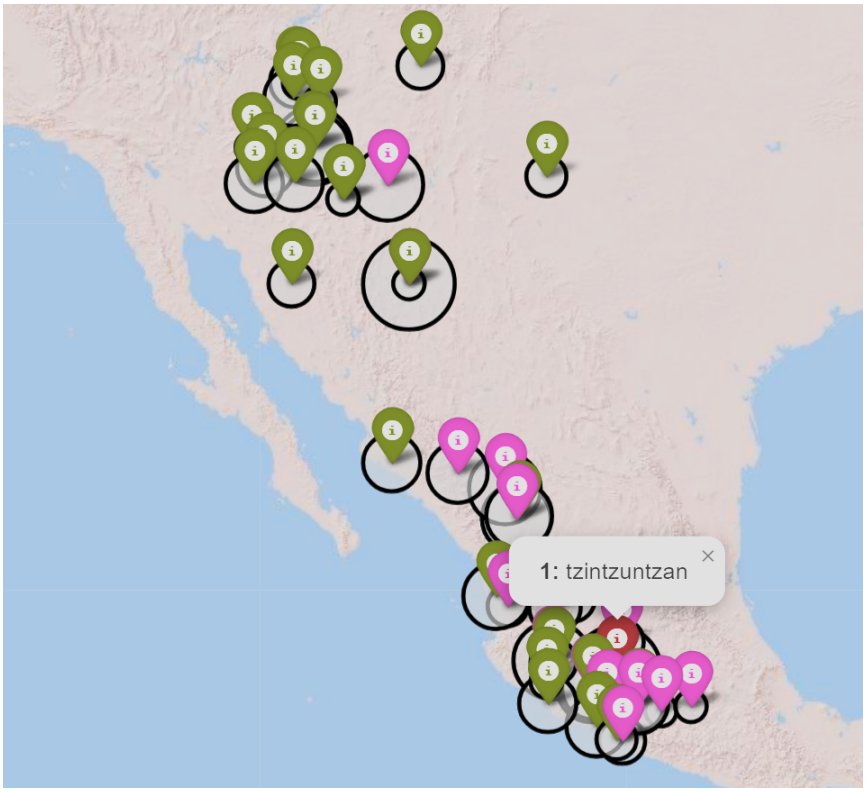

Figura 4.1: Ejemplo de la visualización de los resultados para el caso de Tzintzuntzan para la fuerza de masas.

También al ser un mapa interactivo se puede acercar o alejar la imagen para tener mejor visibilidad de lo que se está estudiando como es el caso de la figura 4.2 que es un acercamiento centrado en Tzintzuntzan para el mapa de la figura .

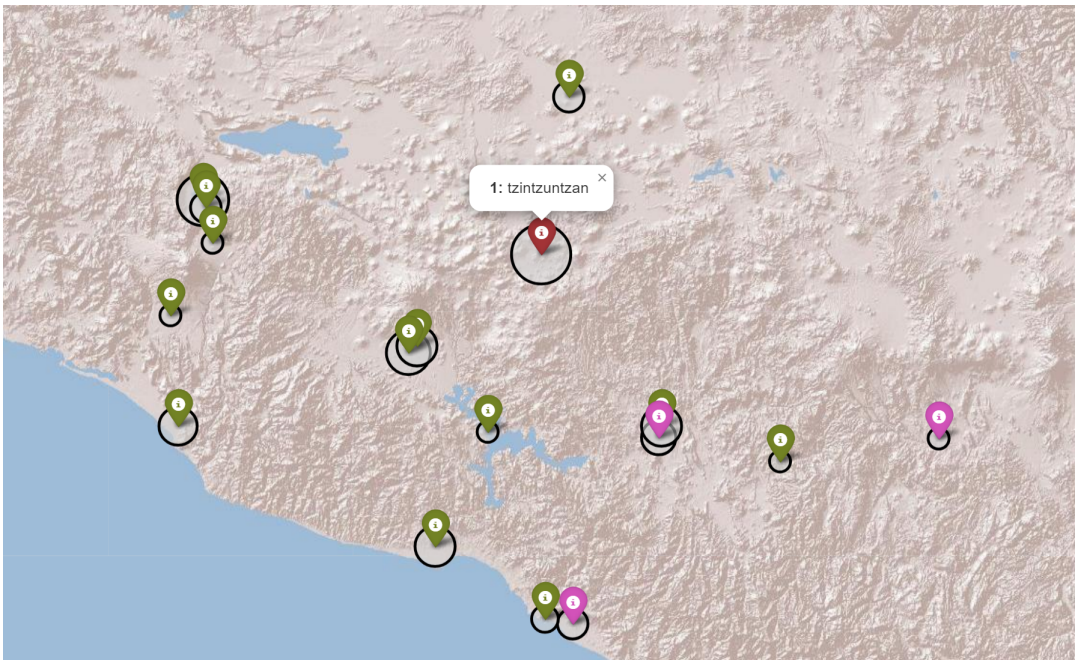

Figura 4.2: Acercamiento de la figura 4.2 donde se muestra el mapa interactivo y la facilidad de visualización.

Este proceso de visualización no fue el primero, al principio se tomó la idea de representarlo por medio de un grafo convencional donde el peso del nodo iba a representar la diversidad del sitio arqueológico y el grosor de la línea dado por el resultado para la similitud según las simulaciones; sin embargo, como se puede observar en la figura 4.3 el mapa es muy difícil de leer al estar varios sitios cercanos se amontonan los nombres y no es muy claro su interpretación. Eso aún dejando fuera todas las conexiones entre los sitios.

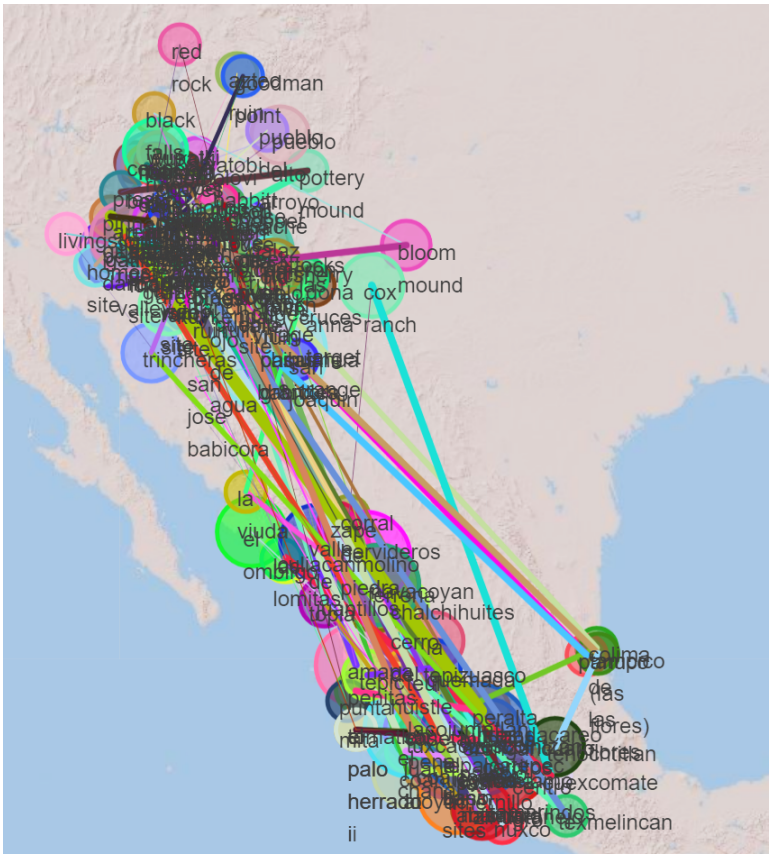

Figura 4.3: Primera propuesta de mapa para la visualización de los resultados.

Capítulo 5

Resultados

5.1. Análisis de la Base de Datos

Para está base de datos también se realiza un análisis de la misma por objeto metálico donde en total se encuentran 49 objetos distintos desde cuchillos hasta anillos metálicos. Donde se crearon documentos de Excel (por fácil acceso y universalidad de estudiarlos) para cada objeto teniendo los datos que se muestran en la tabla 2.1 y agregando una clasificación más general a la usada por Pendergast donde se usan solo las dos primeras letras de la clasificación. Por ejemplo si se tiene un objeto con la siguiente clasificación $IA1a - i$ se transforma solamente en IA obteniendo una clasificación menos específica y más general. Esto se puede apreciar de mejor manera en el ejemplo de la tabla 5.1.

SiteName	**Clasificación General**			**Clasificación de Pendergast**				
	E	**VI**	**XI**	**E**	**VIA**	**VIB**	**XIA1**	**XIA2**
amapa	0	0	4	0	0	0	2	2
cofradia	0	0	2	0	0	0	2	0
ferreria	1	0	0	1	0	0	0	0
huandacareo	1	0	0	1	0	0	0	0
molino	1	0	0	1	0	0	0	0
san juan atoyac	0	1	0	0	1	0	0	0

tzintzuntzan	2	5	0	2	2	3	0	0

Tabla 5.1: Ejemplo de las diferencias entre la clasificación de Pendergast y una clasificación más general para alfileres de cobre.

Sobre el análisis se hicieron tanto para la clasificación original de Pendergast como la nueva clasificación, los resultados separados por objetos incluyen cuales son los tipos con mayor porcentaje de sitios, mapas identificando los sitios arqueológicos, su densidad de artefactos encontrados, como se observa en la figura 5.1 y una gráfica que identifica cuales son los sitios arqueológicos que comparten mayor número de tipos como se observa en la imagen 5.2. Estos resultados además de apoyar a los arqueólogos con un análisis por objeto metálico sirven como un primer acercamiento al resultado visual que se obtiene al final de este trabajo donde se mejora la figura 5.2 utilizando un mapa como fondo de México ya que se tiene la Longitud y Latitud para todos los sitios.

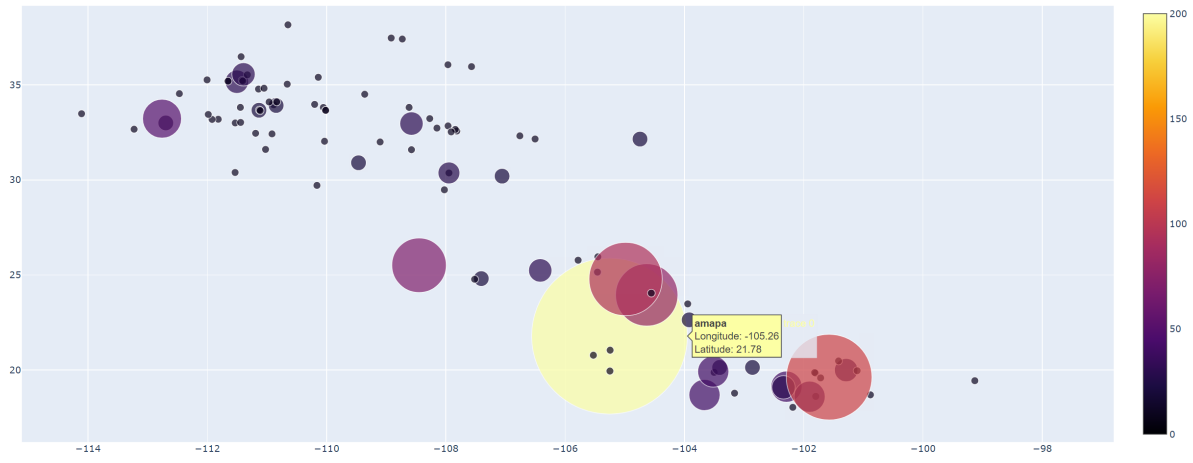

Figura 5.1: Mapa con la ubicación geográfica real de los sitios arqueológicos y sus densidad respectiva para el tipo de cascabel IB.

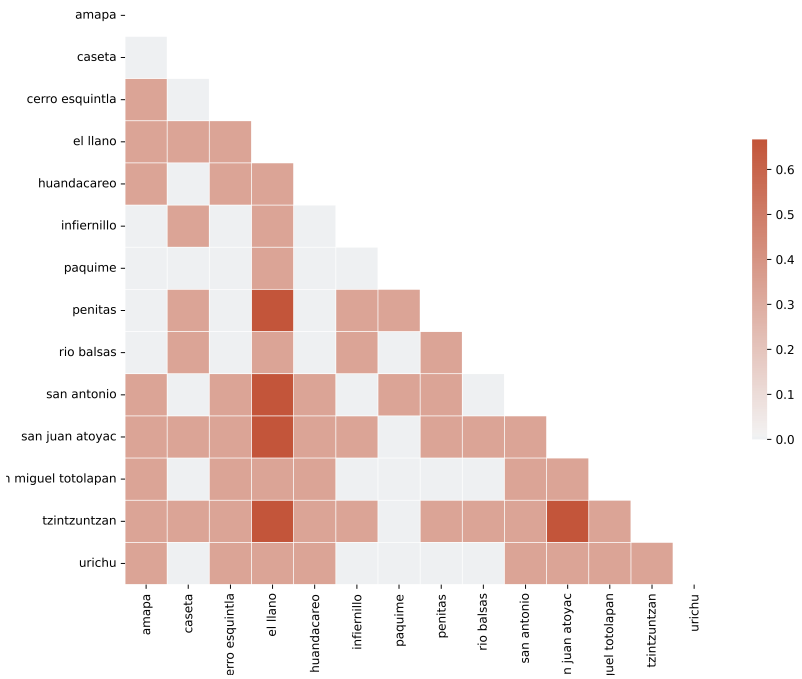

Figura 5.2: Imagen de cuales son los sitios arqueológicos que comparten mayor cantidad de tipos de agujas metálicas.

5.2. Análisis de Distintas Funciones λ

Una de las propiedades que dominan las simulaciones anteriores es la función λ. El valor del término de dispersión para la ecuación 3.9 se obtuvo de forma empírica pero se observó que si era muy grande la atracción no hacía ningún efecto y los sitios siempre se alejaban y no se formaban clústers. En esta sección se va a presentar las distintas funciones λ que se usaron. Los análisis arqueológicos posteriores van a ser los que sirvan de utilidad para identificar que función es mejor que otra, en este trabajo sólo se presenta la descripción de los resultados obtenidos por las distintas funciones λ.

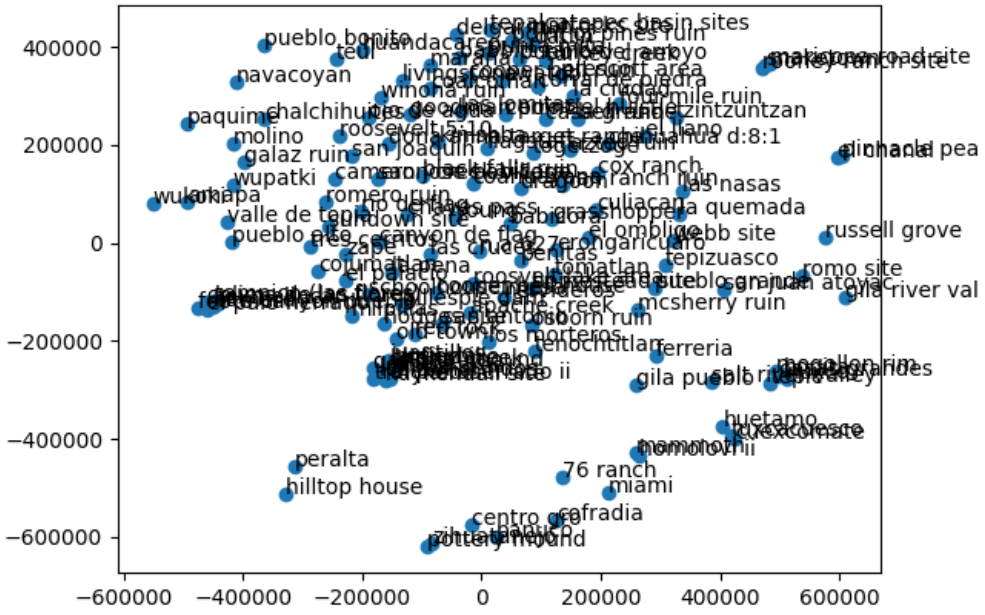

Figura 5.3: Estado final de la simulación los parámetros son un tiempo $t = 100$ y un valor $dt = \frac{100}{1000} = 0.1$ es decir se realizaron 1000 pasos temporales.

5.2.1. Fuerza atractora y repelente igual

Para esta fuerza para unos vectores $\vec{v_1}$ y $\vec{v_1}$, que representan los tipos de cascabeles encontrados en esos sitios, basta con que compartan un sólo tipo ya tienen una fuerza de atracción 1. En cambio si no comparten ningún tipo tienen una fuerza de repulsión de misma magnitud pero signo opuesto. Es decir la función λ es de la siguiente manera:

$$\lambda(\vec{v}_1, \vec{v}_2) = \begin{cases} -1 & \text{if } \forall i, j : v_1(i) \neq v_2(j) \\ 1 & \text{de otra manera.} \end{cases} \tag{5.1}$$

De la misma manera que las otras simulaciones el punto de partida fueron los sitios de manera aleatoria en un cuadrado de $200x200$, también usa el mismo valor para dt y Nt. Se puede observar en la imagen 5.3 como algunos sitios se están alejando más que otros, también una estructura de sitios relativamente constantes unos de los otros en forma de un círculo relleno y finalmente pequeños clústers a las afueras de estos sitios

constantes. Una posible explicación es que al analizar la base de datos la mayoría de los sitios no comparten más que uno o dos tipos de cascabeles al ser la fuerza igual se van a separar unos de otros; sin embargo, existen sitios únicos que tienen tipos muy específicos siendo los que se encuentran más en la periferia de la imagen porque su fuerza repelente es mayor. Para los clústers formados a las afueras de la primer región estos pueden ser los sitios únicos los cuales comparte tipos únicos y diferentes de los demás. Estas hipótesis se deben estudiar más a fondo con el contexto arqueológico que se tiene de los sitios para decidir si representan primero una realidad y después una posibilidad de rutas comerciales.

5.2.2. Fuerza repelente variable

Para este valor de la función λ el valor para la atracción es el mismo que la ecuación 3.9; sin embargo, la fuerza de repulsión es igual al número total de tipos de cascabeles que no comparten. Es decir si son sitios que ambos poseen una cantidad pequeña de variedad de tipos de cascabeles la fuerza de repulsión será pequeña, mientras que si sólo uno posee una gran cantidad de diversidad de tipos la fuerza de repulsión será grande. Viene dada por la siguiente ecuación:

$$\lambda(\vec{v}_1, \vec{v}_2) = \begin{cases} -\frac{\vec{v}_1 \cdot I + \vec{v}_2 \cdot I}{n} & \text{if } \forall i, j : v_1(i) \neq v_2(j) \\ \frac{\sum_{i=1}^{n} \delta_{v_1(i), v_2(i)}}{n} - 1 & \text{de otra manera.} \end{cases} \tag{5.2}$$

Donde I es el vector identidad de dimensión n. Para la simulación se utilizaron los mismos parámetros que se han estado usando. El resultado es bastante interesante, al hacer la fuerza de repulsión variable gana la interacción de atracción por lo que se concentran la mayoría de sitios y unos pocos salen expulsados. También el alejamiento de los sitios parece ser de forma no estructurada donde unos claramente se alejaron más rápido del centro de la concentración que otros a diferencia de la simulación anterior. Los dos sitios que más destacan por su distanciamiento de los demás son Huetamo y Cherry Creek uno localizado en Michoacán y el otro en Colorado. En principio ambos poseen pocos tipos de artefactos y puede que esta sea una de la

razones para que estén tan lejos.

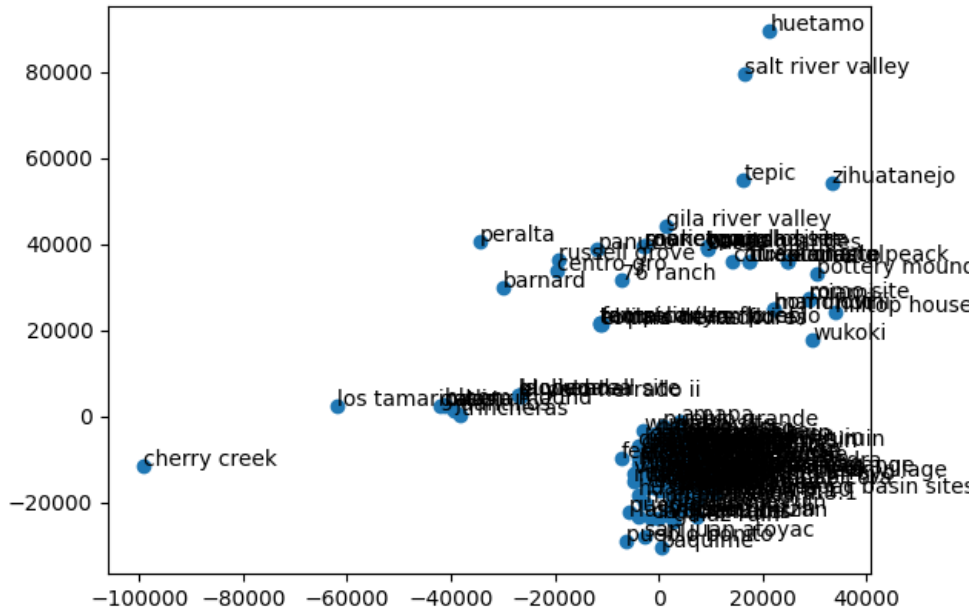

Figura 5.4: Estado final de la simulación los parámetros son un tiempo $t = 100$ y un valor $dt = \frac{100}{1000} = 0.1$ es decir se realizaron 1000 pasos temporales.

5.2.3. Fuerza de reducción de dimensión

Esta función λ es muy similar a la ecuación 3.9, la diferencia radica en la fuerza de atracción en lugar de tomar la dimensión n de los sitios que es el mismo valor para todos los sitios se toma m. La variable m se obtiene de quitar el valor en el índice i de $\vec{v}_1$ y $\vec{v}_2$ donde $\vec{v}_{1i}$ y $\vec{v}_{2i}$ sean ambos ceros reduciendo así la dimensión resultante.

$$\lambda(\vec{v}_1, \vec{v}_2) = \begin{cases} -0.1 & \text{if } \forall i, j : v_1(i) \neq v_2(j) \\ \frac{\sum_{i=1}^{n} \delta_{v_1(i), v_2(i)}}{m} & \text{de otra manera.} \end{cases} \tag{5.3}$$

Donde $m = \vec{v}_1 \cdot \vec{v}_2$, esto hace que la fuerza de atracción sea mayor que la ecuación 3.9 ya que $1 \leq m \leq n$. Al finalizar la simulación se crea una estructura circular ordenada. Al ser la fuerza de atracción mayor se nota como se crea un centro denso y al rededor de este centro se encuentran los sitios que no poseen tantos tipos como se muestra

en la figura 5.5. A pesar de la geometría ordenada que se obtiene, puede que no sirva para identificar nuevos nodos al no poseer clústers pequeños que puedan representar sitios arqueológicos aislados que comerciaban entre ellos, esta observación queda a la interpretación posterior de los resultados.

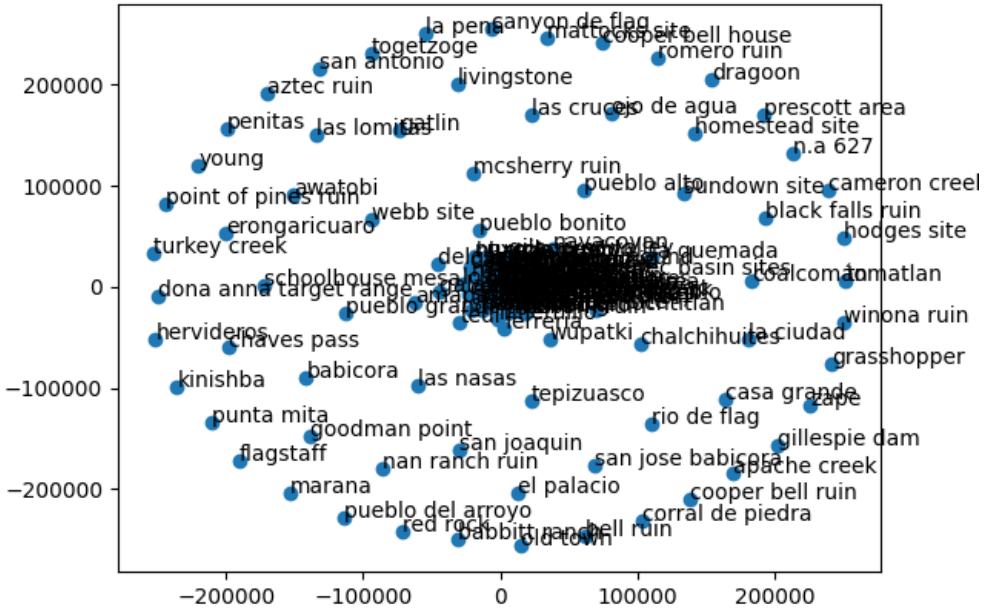

Figura 5.5: Estado final de la simulación los parámetros son un tiempo $t = 100$ y un valor $dt = \frac{100}{1000} = 0.1$ es decir se realizaron 1000 pasos temporales.

5.2.4. Fuerza de masas

En esta función cada sitio va a tener una masa relacionada la cual va a ser el número total de tipos de cascabeles que posean y para la función se van a multiplicar ambas masas. El signo como en las otras fuerzas dependerá si tienen al menos un tipo en común:

$$\lambda(\vec{v}_1, \vec{v}_2) = \begin{cases} -m_2 & \text{if } \forall i, j : v_1(i) \neq v_2(j) \\ m_1 & \text{de otra manera.} \end{cases} \tag{5.4}$$

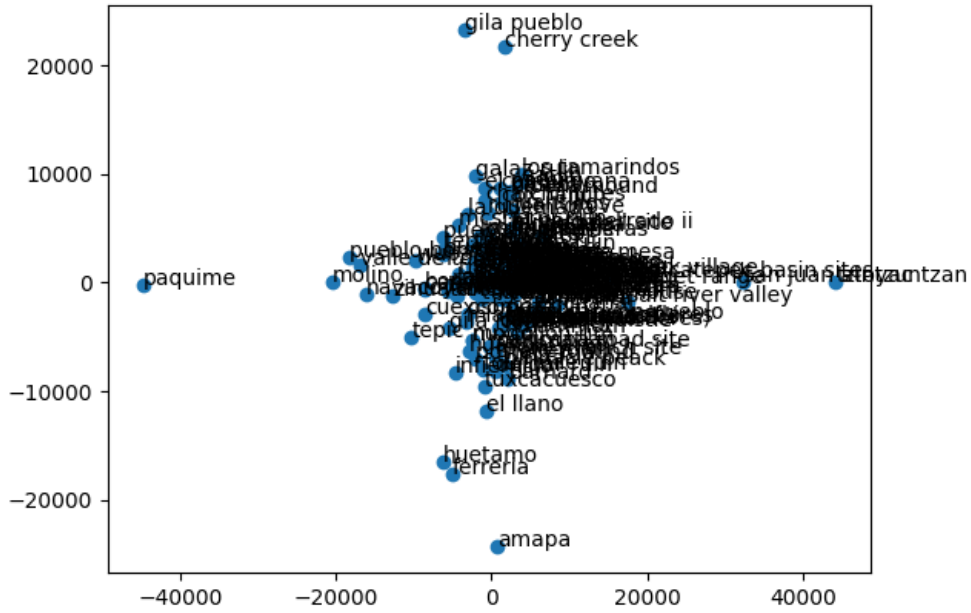

Figura 5.6: Estado final de la simulación los parámetros son un tiempo $t = 100$ y un valor $dt = \frac{100}{1000} = 0.1$ es decir se realizaron 1000 pasos temporales.

Donde $m_1 = \vec{v}_1 \cdot I$ y $m_2 = \vec{v}_2 \cdot I$ por lo que las masas representan el nivel de diversidad que tienen los sitios. Aquí sucede algo que no había pasado en ninguna simulación anterior, es que tanto Tzintzuntzan y Paquimé se encuentran en las periferias de la simulación siendo dos de los sitios con mayor diversidad de cascabeles y más importantes comercialmente. Esto se puede deber a que su masa es muy grande y con los sitios con los que comparten tipos de cascabeles tengan masa pequeña por lo que impera la masa de repulsión. Por lo que la imagen 5.6 podría estar representando cuales son los sitios más únicos.

5.3. Visualización

Una vez que se obtienen los resultados para las distintas λ, se proceden a su vez a crear mapas que representan estos resultados de una manera más entendible para el lector. En esta sección se van a presentar los mapas obtenidos para el sitio de Tzintzuntzan con distintas fuerzas. Es importante mencionar que estos mapas se pueden

obtener para cualquier sitio y son indicadores de similitud mientras que las simulaciones no sólo contenían esta información también los sitios que menos comparten similitudes de objetos. Es importante tomar como caso de estudio a Tzintzuntzan para comprobar la efectividad de las simulaciones debido a que es sabido que fue un nodo comercial muy importante en aquella época y se han realizado varios estudios en torno al sitio por lo que es muy fácil comparar la información actual y desechar modelos debido a su falta de exactitud.

5.3.1. Fuerza repelente constante

Estos resultados son obtenidos después de la simulación de la ecuación 3.9 para λ y siendo Tzintzuntzan el sitio principal del análisis.

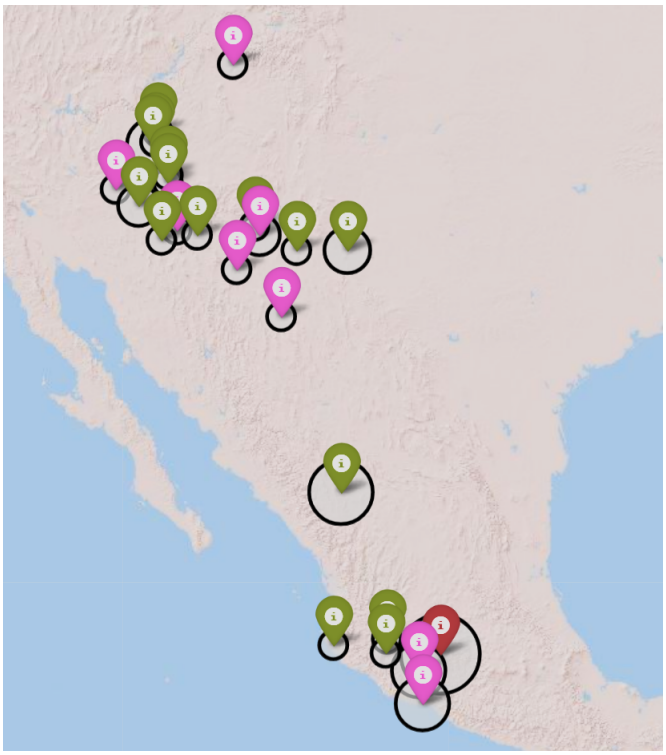

Figura 5.7: Mapa de sitios más similares a Tzintzuntzan (rojo) de acuerdo a la ecuación 3.9.

Es bastante interesante observar como los sitios con mayor similitud representados por el color morado son aquellos que están cerca de la frontera entre México y Estados Unidos.

5.3.2. Fuerza atractora y repelente igual

El resultado de la figura 5.8 es bastante interesante es la única simulación que presenta un sitio del este de México relacionado fuertemente a Tzintzuntzan. Este sitio es Pánuco ubicado en el estado de Veracruz, este sitio solo tiene dos distintos tipos de cascabeles por lo que es una gran información para descartar la función λ dada por la ecuación 5.1 ya que no corresponde con la realidad.

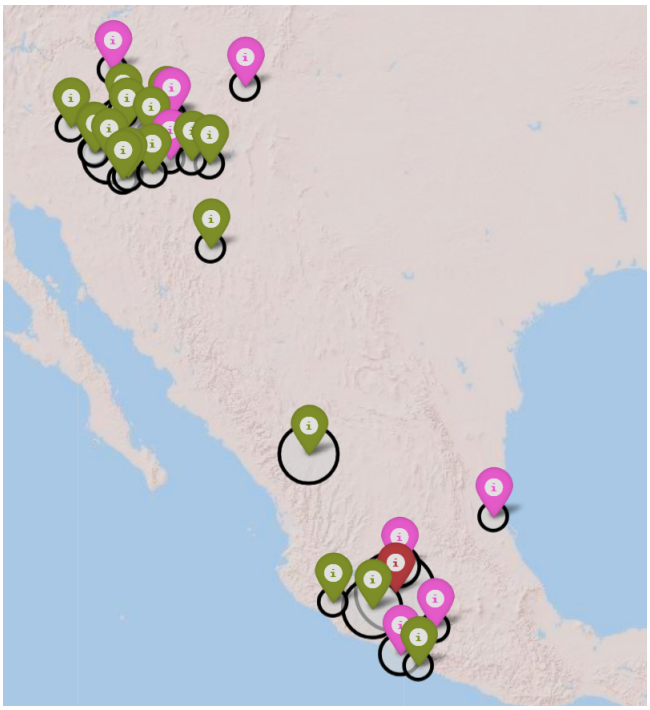

Figura 5.8: Mapa de sitios más similares a Tzintzuntzan (rojo) de acuerdo a la ecuación 5.1.

5.3.3. Fuerza repelente variable

En los resultados de la figura 5.9 se puede observar una mayor concentración de sitios cerca de Tzintzuntzan, a su vez también sitios importantes una vez más en la zona de la frontera de Estados Unidos y México. A diferencia de el caso anterior no presenta una clara evidencia para desestimar la función λ dada por la ecuación 5.2, por lo que se necesita un estudio más profundo como en el primer caso.

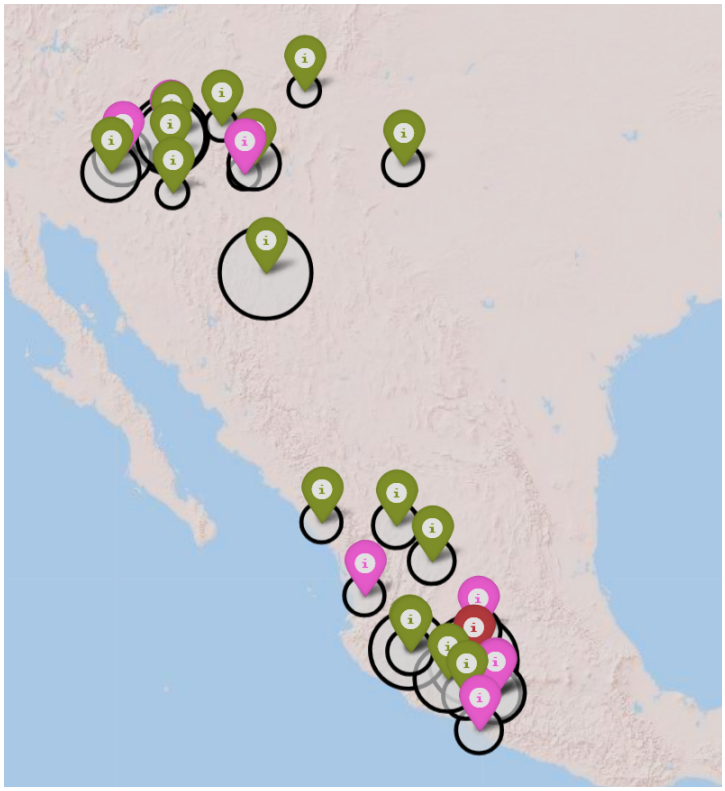

Figura 5.9: Mapa de sitios más similares a Tzintzuntzan (rojo) de acuerdo a la ecuación 5.2.

5.3.4. Fuerza de reducción de dimensión

Los resultados de la figura 5.10 son muy interesantes, ya que no se muestra en el conjunto de los sitios más similares a Tzintzuntzan a alguno que geográficamente se encuentre cerca. A pesar de esto pueda resultar una extrañes no representa todavía un caso tan evidente como el de Pánuco para descartar por completo el modelo. Pero observando otra de las características que presenta esta visualización se puede constatar que los sitios que presenta no contienen gran diversidad de cascabeles, esto al analizar los radios de los círculos en la localización, con estas dos características se puede desechar este modelo ya que al ser el sitio de Tzintzuntzan uno con gran cantidad de diversidad de cascabeles se espera esta característica al mencionar sitios similares.

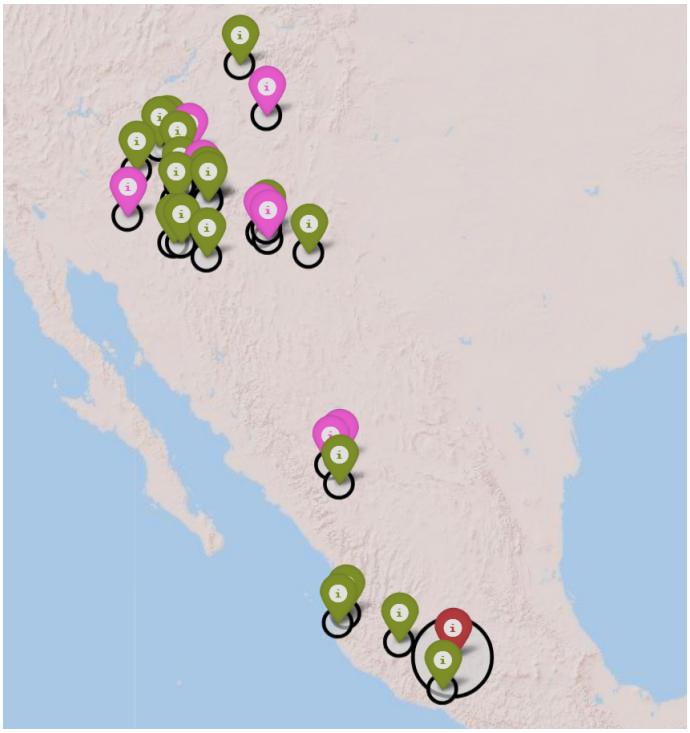

Figura 5.10: Mapa de sitios más similares a Tzintzuntzan (rojo) de acuerdo a la ecuación 5.3.

5.3.5. Fuerza de masas

El mapa de la figura 5.11 representa una gran cantidad de sitos similares cerca de Tzintzuntzan y como estos reducen conforme más se alejan de la ciudad. Además también muestra relación con Paquimé uno de los sitios más diversos en el tipo de cascabeles encontrados. Por estas razones propongo a esta simulación como la más apegada a la realidad; sin embargo, es importante hacer un análisis más profundos de los resultados con ayuda de la arqueología para determinar si es lo suficientemente bueno y poder realizar predicciones ahora utilizando sitios no tan estudiados. También muestra la importancia de una buena visualización ya que en principio

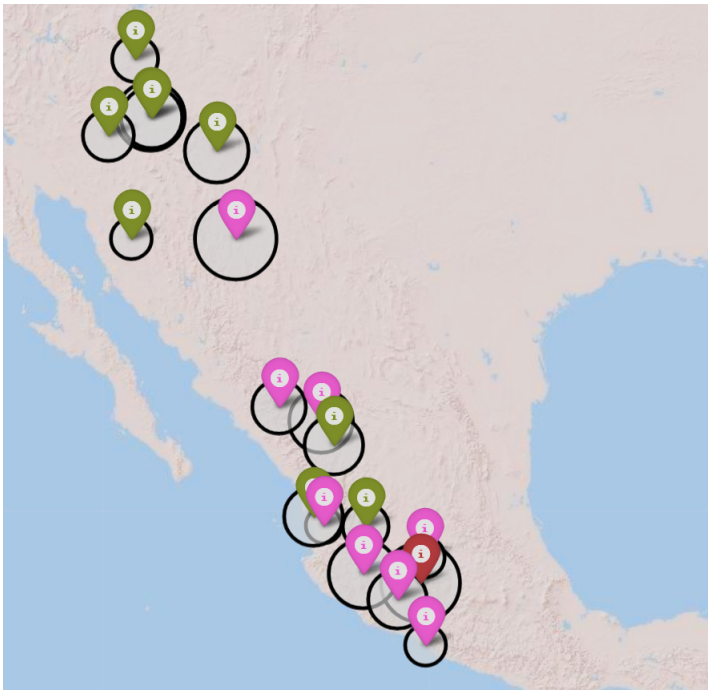

Figura 5.11: Mapa de sitios más similares a Tzintzuntzan (rojo) de acuerdo a la ecuación 5.4.

Capítulo 6

Conclusiones

En este trabajo se presenta una forma de crear redes de similitudes basadas en los campos descritos en la Física, se incluyeron distintas funciones de similitud que proporcionaban redes distintas y también una forma más visual de los resultados usando mapas de la zona de estudio. Además como un resultado alterno es el de una base de datos confiable sobre los objetos de metal en mesoamerica y el sur de los Estados Unidos.

Se compara en primera instancia el método discreto con la solución que se realiza en la física cuando es un problema de varias partículas en una fuerza atractora proporcional al inverso de la distancia. Se encontraron ciertas ventajas del método discreto para analizar datos obteniendo un comportamiento similar al de los campos físicos. Con esto de ninguna manera se pretende sustituir las ecuaciones físicas si no que es una alternativa muy buena para analizar datos con el comportamiento de atracción o alejamiento representando similitud o diferencia de los datos. También se verifico que las estructuras creadas mantienen su comportamiento aunque las condiciones iniciales cambien logrando así uno de los objetivos planteados. A su vez se verifican varias funciones lambda de las cuales una muestra un comportamiento anómalo con los objetivos al aparta tanto Paquimé como Tzintzuntzan, los cuales se tiene constancia de su importancia y como nodos importantes en las rutas del metal en aquel tiempo. Los otros resultados de las funciones se ven relativamente bien y concuerdan con lo esperado; sin embargo, es importante recalcar que no va a ser hasta

un estudio posterior más detallado por parte de los arqueólogos que no se sabrá si cumplen con el objetivo de brindar información importante sobre nodos comerciales más pequeños pudiendo así elaborar nuevas teorías.

Al ser un trabajo multidisciplinario es importante no olvidar que las herramientas serán utilizadas por personas ajenas al área por eso también es importante contar con sus comentarios con lo cual este trabajó se retrasó un poco pero los resultados hablan por si mismos. Gráficas que puedan entender y estudiar en este caso los arqueólogos es importante y recalcar que una retro alimentación es necesaria para mencionar si este método es de utilidad.

Anexo

Código de análisis de datos

```
{import plotly.figure_factory as ff
import plotly.graph_objects as go
import pandas as pd
import DataAnalysisFunctions
import numpy as np
from matplotlib.dates import DateFormatter
import datetime as dt
from pathlib import Path
from collections import Counter
from scipy.stats import chi2, chi2_contingency
from scipy.spatial.distance import pdist
from scipy.spatial.distance import squareform
import decimal
from statsmodels.stats.contingency_tables import (mcnemar,
↪  cochrans_q, SquareTable)
from sklearn.metrics.pairwise import cosine_similarity
import itertools
import statsmodels.formula.api as smf


# create a new context for the decimal format
```

```
ctx = decimal.Context()
ctx.prec = 4
prob = 0.95
alpha = 1.0 - prob
#Flag to not calculate figures every run False does not calculate
↪  over again
flag=False
cwd = Path.cwd()
file_I = Path("Data/In/copper_29_11mod.csv")
file_open = cwd / file_I
out_path_gen = Path(cwd, "Data/OutGeneralInfo/")
out_path_gen.mkdir(parents=True, exist_ok=True)
dataset_I=DataAnalysisFunctions.prepare_dataset(file_open)
all_nmaes= set(dataset_I['UltimateForm'])
DataAnalysisFunctions.create_directories(all_names, cwd)
for artifact_name in all_names:
    out_path_im = Path(cwd, "Data/OutIm/"+artifact_name)
    dataset = DataAnalysisFunctions.get_clean_dataset(dataset_I,
    artifact_name)
    DataAnalysisFunctions.savecsv(dataset,'dataset_'+
    artifact_name+'.csv',artifact_name)
    #scatter plots
    first_specific_type=
    ↪  DataAnalysisFunctions.get_first_specific_type(dataset,
    artifact_name)
    DataAnalysisFunctions.printing_images_general_and_specific(
    dataset, artifact_name)
    #Sending info to a csv
    DataAnalysisFunctions.percentage_of_types_located_in_sites(
```

```
    dataset, artifact_name)


    dataset_L = dataset.iloc[:,0:2]
    dataset_G = dataset.iloc[:,2:first_specific_type]
    dataset_S =
    ↪  dataset.iloc[:,first_specific_type:dataset.shape[1]]


    dataset_bin = dataset.copy()
    a =dataset_bin.iloc[:,2:dataset.shape[1]]
    a[a > 0] = 1
    dataset_bin.iloc[:,2:dataset.shape[1]] = a
    dataset_Lbin = dataset_bin.iloc[:,0:2]
    dataset_Gbin = dataset_bin.iloc[:,2:first_specific_type]
    dataset_Sbin =
    ↪  dataset_bin.iloc[:,first_specific_type:dataset.shape[1]]


    if flag:
        DataAnalysisFunctions.save_images_heatmaps(
        dataset_G_cs,dataset_S_cs,out_path_im)
    dataset_G_cs,dataset_S_cs=DataAnalysisFunctions.
    get_dataset_g_and_s_sharing_objects_between_sites(
    dataset_Gbin,dataset_Sbin,dataset,dataset_bin)
    dataset_G_csb = dataset_G_cs.copy()
    dataset_S_csb = dataset_S_cs.copy()
    dataset_G_csb[dataset_G_csb>0]=1
    dataset_S_csb[dataset_S_csb>0]=1
    dataset_G_csb = dataset_G_csb + dataset_S_csb
    dataset_G_csb[dataset_G_csb==2] = 10
    conections_totals = dataset_G_cs+dataset_S_cs
```

```
conections_totals =
↪  conections_totals/conections_totals.max().max()
nodes = pd.DataFrame(data=np.count_nonzero(dataset_Sbin, axis
↪  =1), index=dataset_Sbin.index)
nodes = nodes.reset_index().reset_index()
nodes.columns = ['ID','label', 'weight']
DataAnalysisFunctions.savecsv(nodes
,(artifact_name+'_nodesG.csv'), artifact_name)

dataset_G_csb=DataAnalysisFunctions.
save_and_calculate_general_edges(artifact_name,dataset_G_csb)

conections_totals = conections_totals.reset_index(drop=True)
conections_totals.columns =
↪  range(conections_totals.columns.size)
conections_totals = conections_totals.
mask(np.triu(np.ones_like(conections_totals
, dtype=bool),0))
conections_totals = conections_totals.unstack()
conections_totals =
↪  conections_totals[~conections_totals.isin([0, np.nan,
↪  np.inf, -np.inf])]
conections_totals = conections_totals.to_frame()
conections_totals = conections_totals.reset_index()
conections_totals.rename(columns={'level_0': 'Source',
↪  'level_1': 'Target', 0: 'weight'}, inplace=True)
DataAnalysisFunctions.savecsv(conections_totals
,(artifact_name+'_edges.csv'),artifact_name,True)
```

```
dataset_S_cs = pd.DataFrame(columns=dataset.index.tolist(),
↪  index=dataset.index.tolist(), dtype=np.float32)
dataset_Sbinn=dataset_Sbin
dataset_Sbinn = dataset_Sbinn.drop(dataset_Sbinn.columns[0],
↪  axis = 1)

if artifact_name=="ring":
    print(dataset_Sbinn)

M=dataset_Sbinn.shape[1]
if M==0:
    M=1
for pair in
↪  itertools.combinations(dataset_Sbinn.index.tolist(), 2):
    dataset_S_cs.loc[pair[0], pair[1]]=
    ↪  np.count_nonzero((dataset_Sbinn.loc[pair[0],:])
    &(dataset_Sbinn.loc[pair[1],:]))/M
    dataset_S_cs.loc[pair[1], pair[0]] =
    ↪  dataset_S_cs.loc[pair[0], pair[1]]

conections_totals = dataset_S_cs
conections_totals =
↪  conections_totals/conections_totals.max().max()
conections_totals = conections_totals.reset_index(drop=True)
conections_totals.columns =
↪  range(conections_totals.columns.size)
conections_totals =
↪  conections_totals.mask(np.triu(np.ones_like(
conections_totals, dtype=bool),0))
```

```
    conections_totals = conections_totals.unstack()
    conections_totals =
    ↪  conections_totals[~conections_totals.isin([0, np.nan,
    ↪  np.inf, -np.inf])]
    conections_totals = conections_totals.to_frame()
    conections_totals = conections_totals.reset_index()
    conections_totals.rename(columns={'level_0': 'Source',
    ↪  'level_1': 'Target', 0: 'weight'}, inplace=True)
    DataAnalysisFunctions.savecsv(conections_totals,(artifact_name
    +'_edgesSnA.csv'),artifact_name,True)}
```

Código de Runge-Kutta para tres partículas

```
import numpy as np
from pathlib import Path
import pandas as pd
import matplotlib.pyplot as plt
import itertools
import CreateGraphs
import sys
sys.path.append("C:\\Users\\Paris\\Documents\\Maestria\\Tesis\\Código")
import Tools
def createRandomPoints(siteNames):
    points ={}
    nums = np.random.choice(range(-1,1+1), size=(1, 2),
    ↪  replace=False)
    for item in siteNames:
        nums = np.random.uniform(low=-100, high=100, size=(1, 2))
```

```
        # Keep only the points that fall inside the circle of
        ↪ radius 1
        #distances = np.sqrt(np.sum(points**2, axis=1))
        #points = points[distances <= 1]
        points[item] = np.asarray((nums[0][0], nums[0][1]))
    return points


def createNonrandomPoints(siteNames):
    points={}
    i=0
    j=0
    for item in siteNames:
        points[item] = np.array([i,j])
        if i==1:
            j=+1
            i=0
        i=+1
    return points
def printingImagesWithNames(points):
    # Create empty lists for x and y coordinates
    x = []
    y = []
    for siteNames in points:
        # Append x and y coordinates to lists
        x.append(points[siteNames][0])
        y.append(points[siteNames][1])
    plt.scatter(x, y)
    for i, siteNames in enumerate(points):
        # Access individual x and y coordinates
```

```
        x_coord = points[siteNames][0]
        y_coord = points[siteNames][1]
        # Annotate point with site name
        plt.annotate(siteNames, (x_coord, y_coord))
    plt.show()
#Getting constants so initial velocity equals 0
def gettingConstants(points):
    X0=[]
    Y0=[]
    for siteNames in points:
        # Append x and y coordinates to lists
        X0.append(points[siteNames][0])
        Y0.append(points[siteNames][1])
    return X0,Y0

def calculateDistanceBetweenTwoPoints(firstPoint, secondPoint):
    return np.linalg.norm(firstPoint - secondPoint)
def calculateDirectionBetweenTwoPoints(firstPoint, secondPoint):
    return -(firstPoint-secondPoint)
def deleteZeroesFromBoth(firstArray,SecondArray):
    mask = np.logical_or(firstArray, SecondArray)
    indices = [i for i, x in enumerate(mask) if x]
    return np.array([firstArray[i] for i in indices])
    ,np.array([SecondArray[i] for i in indices])

def calculateForceMagnitud(pair, dataset_I):
    M=dataset_I.shape[1]-1
    firstRowNumber = dataset_I.loc[dataset_I["SiteName"]==pair[0]]
    .index[0]
```

```
    secondRowNumber =
    ↪  dataset_I.loc[dataset_I["SiteName"]==pair[1]]
    .index[0]
    dataset_I = dataset_I.drop(dataset_I.columns[0], axis = 1)
    firstSiteRow, secondSiteRow =deleteZeroesFromBoth(
     dataset_I.loc[firstRowNumber,:]
    .astype(int)
    ,dataset_I.loc[secondRowNumber,:].astype(int))
    M=len(firstSiteRow)
    number= np.count_nonzero((firstSiteRow)&(secondSiteRow))/M
    #First Try
    if number!=0:
        return number
    else:
        return -0.01
    #if number>=M/2:
    #    return number
    #else:
    #    return number-0.5
def calculateForceMagnitud1(pair, dataset_I):
    M=dataset_I.shape[1]-1
    firstRowNumber = dataset_I.loc[dataset_I["SiteName"]==pair[0]]
    .index[0]
    secondRowNumber =
    ↪  dataset_I.loc[dataset_I["SiteName"]==pair[1]]
    .index[0]
    dataset_I = dataset_I.drop(dataset_I.columns[0], axis = 1)
    number= np.count_nonzero(
    (dataset_I.loc[firstRowNumber,:].astype(int))
```

```
    &(dataset_I.loc[secondRowNumber,:].astype(int)))/M
    if number!=0:
        return number
    else:
        return -0.1

#Number of time steps I am going to use
Nt = 10000
#Parameter that sometimes helps
courant = 1
#maximum time
Ntmax=4
dt =courant*Ntmax/Nt

cwd = Path.cwd().resolve()
file_I = Path("TestsThreeParticles/Data/
ThreePointsData/first_three_point_try3.csv")  # Relative path

x1=[]
y1=[]
x2=[]
y2=[]
x3=[]
y3=[]
file_open = cwd / file_I
print(cwd)
dataset_I = pd.read_csv(file_open)
points = createNonrandomPoints(dataset_I["SiteName"])
constX, constY = gettingConstants(points)
```

```
printingImagesWithNames(points)
velocity = Tools.initializeVelocity(points)
rhsVelocity = Tools.initializeVelocity(points)
print("points", points)
print("velocity", velocity)
import time
start_time = time.time()

for t in np.arange(0,Ntmax, dt):
    pointsPast= points
    velocityPast = velocity
    for j in range(1,4):
        forces =np.zeros((len(dataset_I),2))
        for pair in
        ↪  itertools.combinations(dataset_I["SiteName"],2):
            distance=calculateDistanceBetweenTwoPoints(
            points[pair[0]], points[pair[1]])
            direction=calculateDirectionBetweenTwoPoints(
            points[pair[0]], points[pair[1]])
#This force is the one between the arrays of the sites
            force =calculateForceMagnitud1(pair,dataset_I)
#Calculate the i,j position of boths sites in the
            firstRowNumber = dataset_I.loc[dataset_I["SiteName"]
            ==pair[0]].index[0]
            secondRowNumber = dataset_I.loc[dataset_I["SiteName"]
            ==pair[1]].index[0]
#This piece of the code should be different, it deals
for divisions of 0
            if distance !=0:
```

```
                alfa1=(1/distance)*0.5*force*direction[0]
                beta1=(1/distance)*0.5*force*direction[1]
                #print("entrooo")
            else:
                alfa1=0.0
                beta1=0.0
#Adds boths terms in each site for later use the terms
are the ones of 0.5*t**2
            forces[firstRowNumber]+= np.asarray((alfa1,beta1))
            forces[secondRowNumber]+= np.asarray((-alfa1,-beta1))
#End part where each particle should have it forces of others
        for i, siteNames in enumerate(dataset_I["SiteName"]):
            x11=constX[i]+forces[i][0]
            y11=constY[i]+forces[i][1]
            rhsVelocity[siteNames]=np.asarray((x11,y11))
        for i, siteNames in enumerate(dataset_I["SiteName"]):
            if j == 1:
                points[siteNames] = pointsPast[siteNames]
                +dt*velocity[siteNames]
                velocity[siteNames] = velocityPast[siteNames]
                +dt*rhsVelocity[siteNames]
            if j == 2:
                points[siteNames] = 0.75*pointsPast[siteNames]
                +0.25*points[siteNames]
                +0.25*dt*velocity[siteNames]
                velocity[siteNames] = 0.75*velocityPast[siteNames]
                +0.25*velocity[siteNames]
                +0.25*dt*rhsVelocity[siteNames]
            if j==3:
```

```
                points[siteNames] = (pointsPast[siteNames]
                +2.0*points[siteNames]
                +2.0*dt*velocity[siteNames])/3
                velocity[siteNames] = (velocityPast[siteNames]
                +2.0*velocity[siteNames]
                +2.0*dt*rhsVelocity[siteNames])/3
    x1.append(points["76 ranch"][0])
    y1.append(points["76 ranch"][1])
    x2.append(points["amapa"][0])
    y2.append(points["amapa"][1])
    x3.append(points["apache creek"][0])
    y3.append(points["apache creek"][1])
end_time = time.time()
elapsed_time = end_time - start_time

print(f"Execution time: {elapsed_time} seconds")
CreateGraphs.createThreeTimeGraph(x1,y1,x2,y2,x3,y3)
```

Código del método discreto para tres partículas

```
import numpy as np
from pathlib import Path
import pandas as pd
import matplotlib.pyplot as plt
import itertools
import CreateGraphs
def createRandomPoints(siteNames):
    points ={}
```

```
    nums = np.random.choice(range(-1,1+1), size=(1, 2),
    ↪ replace=False)
    for item in siteNames:
        nums = np.random.uniform(low=-100, high=100, size=(1, 2))
        # Keep only the points that fall inside the circle of
        ↪ radius 1
        #distances = np.sqrt(np.sum(points**2, axis=1))
        #points = points[distances <= 1]
        points[item] = np.asarray((nums[0][0], nums[0][1]))
    return points


def createNonrandomPoints(siteNames):
    points={}
    i=0
    j=0
    for item in siteNames:
        points[item] = np.array([i,j])
        if i==1:
            j=+1
            i=0
        i=+1
    return points
def printingImagesWithNames(points):
    # Create empty lists for x and y coordinates
    x = []
    y = []
    for siteNames in points:
        # Append x and y coordinates to lists
        x.append(points[siteNames][0])
```

```
        y.append(points[siteNames][1])
    plt.scatter(x, y)
    for i, siteNames in enumerate(points):
        # Access individual x and y coordinates
        x_coord = points[siteNames][0]
        y_coord = points[siteNames][1]
        # Annotate point with site name
        plt.annotate(siteNames, (x_coord, y_coord))
    plt.show()
#Getting constants so initial velocity equals 0
def gettingConstants(points):
    X0=[]
    Y0=[]
    for siteNames in points:
        # Append x and y coordinates to lists
        X0.append(points[siteNames][0])
        Y0.append(points[siteNames][1])
    return X0,Y0



def calculateDistanceBetweenTwoPoints(firstPoint, secondPoint):
    return np.linalg.norm(firstPoint - secondPoint)
def calculateDirectionBetweenTwoPoints(firstPoint, secondPoint):
    return -(firstPoint-secondPoint)
def deleteZeroesFromBoth(firstArray,SecondArray):
    mask = np.logical_or(firstArray, SecondArray)
    indices = [i for i, x in enumerate(mask) if x]
    return np.array([firstArray[i] for i in indices]),
    ↪  np.array([SecondArray[i] for i in indices])
```

```
def calculateForceMagnitud(pair, dataset_I):
    M=dataset_I.shape[1]-1
    firstRowNumber =
    ↪  dataset_I.loc[dataset_I["SiteName"]==pair[0]].index[0]
    secondRowNumber =
    ↪  dataset_I.loc[dataset_I["SiteName"]==pair[1]].index[0]
    dataset_I = dataset_I.drop(dataset_I.columns[0], axis = 1)
    firstSiteRow, secondSiteRow =deleteZeroesFromBoth(
    dataset_I.loc[firstRowNumber,:].astype(int),
    dataset_I.lc[secondRowNumber,:].astype(int))

    M=len(firstSiteRow)
    number= np.count_nonzero((firstSiteRow)&(secondSiteRow))/M
    #First Try
    if number!=0:
        return number
    else:
        return -0.01
    #if number>=M/2:
    #    return number
    #else:
    #    return number-0.5
def calculateForceMagnitud1(pair, dataset_I):
    M=dataset_I.shape[1]-1
    firstRowNumber =
    ↪  dataset_I.loc[dataset_I["SiteName"]==pair[0]].index[0]
    secondRowNumber =
    ↪  dataset_I.loc[dataset_I["SiteName"]==pair[1]].index[0]
```

```
    dataset_I = dataset_I.drop(dataset_I.columns[0], axis = 1)
    number= np.count_nonzero(
    (dataset_I.loc[firstRowNumber,:].astype(int))
    &
    (dataset_I.loc[secondRowNumber,:].astype(int)))/M
    if number!=0:
        return number
    else:
        return -0.1
#Number of time steps I am going to use
Nt = 10000
#Parameter that sometimes helps
courant = 1
#maximum time
Ntmax=4
dt =courant*Ntmax/Nt

cwd = Path.cwd().resolve()
file_I = Path("TestsThreeParticles/Data/
ThreePointsData/first_three_point_try.csv")  # Relative path

x1=[]
y1=[]
x2=[]
y2=[]
x3=[]
y3=[]

file_open = cwd / file_I
```

```
print(cwd)
dataset_I = pd.read_csv(file_open)
points = createNonrandomPoints(dataset_I["SiteName"])
constX, constY = gettingConstants(points)
printingImagesWithNames(points)
import time
start_time = time.time()
for t in np.arange(0,Ntmax, dt):
    forces =np.zeros((len(dataset_I),2))
    for pair in itertools.combinations(dataset_I["SiteName"], 2):
        distance=calculateDistanceBetweenTwoPoints
        (points[pair[0]], points[pair[1]])
        direction=calculateDirectionBetweenTwoPoints
        (points[pair[0]], points[pair[1]])
        #This force is the one between the arrays of the sites
        force =calculateForceMagnitud1(pair,dataset_I)
        firstRowNumber =
        ↪  dataset_I.loc[dataset_I["SiteName"]==pair[0]].index[0]
        secondRowNumber =
        ↪  dataset_I.loc[dataset_I["SiteName"]==pair[1]].index[0]
        #This piece of the code should be different, it deals for
        ↪  divisions of 0
        if distance !=0:
            alfa1=(1/distance)*0.5*force*direction[0]*t**2
            beta1=(1/distance)*0.5*force*direction[1]*t**2
            print("entrooo")
        else:
            alfa1=0.0
            beta1=0.0
```

```
            #Adds boths terms in each site for later use the terms are
            ↪  the ones of 0.5*t**2
            forces[firstRowNumber]+= np.asarray((alfa1,beta1))
            forces[secondRowNumber]+= np.asarray((-alfa1,-beta1))
        #End part where each particle should have it forces of others
        for i, siteNames in enumerate(dataset_I["SiteName"]):
            x11=constX[i]+forces[i][0]
            y11=constY[i]+forces[i][1]
            points[siteNames]=np.asarray((x11,y11))
        x1.append(constX[0]+forces[0][0])
        y1.append(constY[0]+forces[0][1])
        x2.append(constX[1]+forces[1][0])
        y2.append(constY[1]+forces[1][1])
        x3.append(constX[2]+forces[2][0])
        y3.append(constY[2]+forces[2][1])
    end_time = time.time()
    elapsed_time = end_time - start_time

    print(f"Execution time: {elapsed_time} seconds")
    CreateGraphs.createThreeTimeGraph(x1,y1,x2,y2,x3,y3)
```

Código Runge-Kutta para n partículas

```
import numpy as np
from pathlib import Path
import pandas as pd
import matplotlib.pyplot as plt
import itertools
import Tools
```

```
import RungeKutta
#Number of time steps I am going to use
Nt = 15
#Parameter that sometimes helps
courant = 1
#maximum time
Ntmax=15
dt =courant*Ntmax/Nt


cwd = Path.cwd()
file_I = Path("Data/OutGeneralInfo/bell_Sbin.csv")
file_open = cwd / file_I


dataset_I = pd.read_csv(file_open)
points = Tools.createRandomPoints(dataset_I["SiteName"])
velocity =Tools.initializeVelocity(points)
rhsVelocity = Tools.initializeVelocity(points)
constX, constY = Tools.gettingConstants(points)
Tools.printingImagesWithNames(points)
import time
start_time = time.time()
#Decides which force I use for lambda
forceToUse = 9
lastPositionDataSet= Tools.creatingNewDataset(dataset_I)
for t in np.arange(0,Ntmax, dt):
    pointsPast= points
    velocityPast = velocity
    for j in range(1,4):
```

```
        forces= RungeKutta.calculateRhsOfVelocity(dataset_I,
        ↪  points,forceToUse)
        for i, siteNames in enumerate(dataset_I["SiteName"]):
            x11=constX[i]+forces[i][0]
            y11=constY[i]+forces[i][1]
            rhsVelocity[siteNames]=np.asarray((x11,y11))
        points,velocity = RungeKutta.calculatePosAndVelocity
        (points,velocity,pointsPast, velocityPast,
        rhsVelocity,dt,dataset_I,j)
end_time = time.time()
elapsed_time = end_time - start_time
print(f"Execution time: {elapsed_time} seconds")
Tools.printingImagesWithNames(points)
Tools.gettingLastDistances(points,lastPositionDataSet, forceToUse)
```

Código del método discreto para n partículas

```
import numpy as np
from pathlib import Path
import pandas as pd
import matplotlib.pyplot as plt
import itertools
import Tools
#Number of time steps I am going to use
Nt = 1000
#Parameter that sometimes helps
courant = 1
#maximum time
Ntmax=10000
```

```
dt =courant*Ntmax/Nt


cwd = Path.cwd()
file_I = Path("Data/OutGeneralInfo/bell_Sbin.csv")
file_open = cwd / file_I


dataset_I = pd.read_csv(file_open)
points = Tools.createRandomPoints(dataset_I["SiteName"])
constX, constY = Tools.gettingConstants(points)
Tools.printingImagesWithNames(points)
#Force to Use
forceToUse = 4
lastPositionDataSet= Tools.creatingNewDataset(dataset_I)
import time
start_time = time.time()
for t in np.arange(0,Ntmax, dt):
    forces =np.zeros((len(dataset_I),2))
    for pair in itertools.combinations(dataset_I["SiteName"], 2):
        distance=Tools.calculateDistanceBetweenTwoPoints
        (points[pair[0]], points[pair[1]])
        direction=Tools.calculateDirectionBetweenTwoPoints
        (points[pair[0]], points[pair[1]])
        force =Tools.calculateForceMagnitud(pair,dataset_I,
        ↪  forceToUse)
        if forceToUse == 4:
            massFirstSite, massSecondSite =Tools.getmasses(pair,
            ↪  dataset_I)
        else:
            massFirstSite =1
```

```
            massSecondSite =1
        firstRowNumber =
        ↪  dataset_I.loc[dataset_I["SiteName"]==pair[0]].index[0]
        secondRowNumber =
        ↪  dataset_I.loc[dataset_I["SiteName"]==pair[1]].index[0]
        if distance !=0:
            alfa1=((1/distance)*0.5*force*direction[0]*dt**2)
            /massFirstSite
            beta1=((1/distance)*0.5*force*direction[1]*dt**2)
            /massSecondSite
        else:
            alfa1=0.0
            beta1=0.0
        forces[firstRowNumber]+= np.asarray((alfa1,beta1))
        forces[secondRowNumber]+= np.asarray((-alfa1,-beta1))
    for i, siteNames in enumerate(dataset_I["SiteName"]):
        x1=constX[i]+forces[i][0]
        y1=constY[i]+forces[i][1]
        points[siteNames]=np.asarray((x1,y1))
    print(t/Ntmax*100)
end_time = time.time()
elapsed_time = end_time - start_time
print(elapsed_time)
Tools.printingImagesWithNames(points)
Tools.gettingLastDistances(points,lastPositionDataSet, forceToUse)
```

Código de visualización

```
""" Este código esta diseñado para analizar y visualizar datos
↪ geográficos y de conexión entre sitios arqueológicos
↪ específicos en México y el sur de Esatdos Unidos
utilizando varias bibliotecas de Python.
Este código genera un mapa interactivo con Folium que muestra
↪ conexiones entre distintos sitios según la distancia de fuerza
↪ y otros datos relacionados
con artefactos encontrados en esos sitios. Los círculos y
↪ marcadores en el mapa representan sitios específicos y están
↪ coloreados y dimensionados de acuerdo
con datos relevantes (como la cantidad de distintos tipos de
↪ cascabeles). La lógica detrás de circleadd incluye el filtrado
↪ de sitios basado en cuantiles de distancia
y el manejo de listas para asegurar que cada sitio sea procesado y
↪ visualizado adecuadamente.

Importar las bibliotecas necesarias para manejo de mapas, análisis
↪ de datos, y cálculos matemáticos. """
import folium
import numpy as np
import pandas as pd
from scipy.spatial.distance import pdist, squareform
from pathlib import Path
import math

def circleadd(force_dist, siteori, circle_data, SitePair, color1,
↪ color2, onlist):
    # Extraer las distancias de fuerza para el sitio de origen y
    ↪ calcular cuantiles.
```

```
force_dist_tzin = force_dist.loc[siteori,:]
quantiles = force_dist_tzin.quantile([0.25, 0.5, 0.75, 0.9,
↪ 0.95]).T
# Identificar los sitios con distancias en el cuantil más alto
↪ (90%).
#Colocar los sitios más cercanos primero
highest_quantiles = force_dist_tzin.apply(lambda row: row >=
↪ quantiles.loc[0.95])
result_df = force_dist_tzin[highest_quantiles]
SiteNames = result_df.index.tolist()
#SiteNames = rearrengeList(SiteNames,siteori)
# Calcular el total de tipos distintos de cascabeles (un tipo
↪ de artefacto) para cada sitio.
columns_to_count = ['IA', 'IB', 'IC', 'ID', 'IE', 'IA1a-i',
↪ 'IA1a-ii', 'IA1b', 'IA1c', 'IA2a', 'IA4a', 'IA5a', 'IA6a',
↪ 'IB1a', 'IB2a', 'IC10a', 'IC12a', 'IC13a', 'IC14a',
↪ 'IC15a', 'IC16a', 'IC18a', 'IC19a', 'IC1a', 'IC1a ',
↪ 'IC1c', 'IC2a', 'IC3a', 'IC4a', 'IC5a', 'IC6a', 'IC7a',
↪ 'IC8a', 'IC9a', 'ID10a', 'ID1a', 'ID2a', 'ID3a', 'ID4a',
↪ 'ID5a', 'ID6a', 'ID7a', 'ID9a', 'IE1a', 'IE2', 'IE3a']
location_info['NonZeroCount'] =
↪ location_info[columns_to_count].apply(lambda row: (row !=
↪ 0).sum(), axis=1)
# Normalizar los números totales de cascabeles para usarlos en
↪ la coloración.
bell_number_info['Number'] = bell_number_info['Number'] /
↪ bell_number_info.Number.values.max()
# Preparar los datos de círculos (localización y color) para
↪ el sitio de origen y sitios relacionados.
```

```
    info_Site_Coordinates=location_info[location_info["SiteName"]
    ==siteori]
    number_bell =
    ↪  bell_number_info[bell_number_info["SiteName"]==siteori]
    circ_data = (info_Site_Coordinates["Latitude"].values[0],
                info_Site_Coordinates["Longitude"].values[0],
                info_Site_Coordinates["NonZeroCount"]
                .values[0],
                siteori,
                color1
                )
    # Agregar el sitio de origen a la lista si no está ya
    ↪  incluido.
    if siteori not in onlist:
        onlist.append(siteori)
        circle_data.append(circ_data)
    # Repetir el proceso para cada uno de los sitios con
    ↪  distancias en el cuantil más alto.
    for site in SiteNames:
        SitePair.append([siteori, site])
        # Similar al bloque anterior pero para cada sitio
        ↪  relacionado.
        info_Site_Coordinates=location_info[
        location_info["SiteName"]==site]
        number_bell =
        ↪  bell_number_info[bell_number_info["SiteName"]==site]
        circ_data = (info_Site_Coordinates["Latitude"].values[0],
                    info_Site_Coordinates["Longitude"].values[0],
                    info_Site_Coordinates["NonZeroCount"]
```

```
                    .values[0],
                    site,
                    color2
                    )
        if site not in onlist:
            onlist.append(site)
            circle_data.append(circ_data)
    return circle_data, SiteNames, SitePair, onlist

def rearrengeList(SiteNames,siteori):
    latOr,lonOr = location_info.loc[location_info['SiteName'] ==
    ↪  siteori, ['Latitude', 'Longitude']].values[0]
    for i, sites in enumerate(SiteNames[:-1], start =1):
        latFir, lonFir =
        ↪  location_info.loc[location_info['SiteName'] == sites,
        ↪  ['Latitude', 'Longitude']].values[0]
        distFir =
        ↪  getDistanceBetweenPointsNew(latOr,lonOr,latFir,lonFir)
        latSec, lonSec =
        ↪  location_info.loc[location_info['SiteName'] ==
        ↪  SiteNames[i], ['Latitude', 'Longitude']].values[0]
        distSec =
        ↪  getDistanceBetweenPointsNew(latOr,lonOr,latSec,lonSec)
        if distFir > 100 and distSec < 100:
            SiteNames[i-1], SiteNames[i] = SiteNames[i],
            ↪  SiteNames[i-1]
            sites = SiteNames[i]
            i = i+1
    return SiteNames
```

```
from numpy import sin, cos, arccos, pi, round

def rad2deg(radians):
    degrees = radians * 180 / pi
    return degrees

def deg2rad(degrees):
    radians = degrees * pi / 180
    return radians

def getDistanceBetweenPointsNew(latitude1, longitude1, latitude2,
↪  longitude2):

    theta = longitude1 - longitude2

    distance = 60 * 1.1515 * rad2deg(
        arccos(
            (sin(deg2rad(latitude1)) * sin(deg2rad(latitude2))) +
            (cos(deg2rad(latitude1)) * cos(deg2rad(latitude2)) *
            ↪  cos(deg2rad(theta)))
        )
    )
    return round(distance * 1.609344, 2)

# Definir directorios de entrada y salida.
cwd = Path.cwd()
out_path = Path(cwd, "Out/")
```

```
in_path = Path(cwd, "In/")

# Cargar los datos de entrada.
iofile  = 'ForceMasses1.csv'
force_path = in_path / iofile
location_info = pd.read_csv(Path('In/dataset_bell.csv'))
bell_number_info = pd.read_csv(Path('In/dataset_bell_number.csv'))
force_table = pd.read_csv(force_path, index_col=0)

# Inicializar listas para almacenar datos de conexiones y
↪  círculos.
circle_data= []
SiteNames =  []
SitePair = []
onlist = []

# Analizar las conexiones para un sitio inicial y luego para los
↪  sitios relacionados.
sitename = 'tzintzuntzan' #amapa paquime tzintzuntzan
circle_data, SiteNames, SitePair, onlist = circleadd(force_table,
↪  sitename, circle_data, SitePair, "darkred", "purple", onlist)
for site in SiteNames:
    circle_data, SiteNames, SitePair, onlist =
    ↪  circleadd(force_table, site, circle_data, SitePair,
    ↪  "purple", "darkgreen", onlist)
# Eliminar duplicados de pares de sitios y preparar la
↪  visualización en el mapa.
unique_combinations = set()
for sublist in SitePair:
```

```
    unique_combinations.add(tuple(sorted(sublist)))
SitePairU = [list(combination) for combination in
↪  unique_combinations]
# Crear un mapa de Folium para visualizar los datos.
mexico_map = folium.Map(
    location=[23.6345, -102.5528],
    zoom_start=6,
    tiles='http://server.arcgisonline.com/ArcGIS/rest/services
    /World_Shaded_Relief/MapServer/tile/{z}/{y}/{x}',
    attr='Tiles &copy; Esri &mdash; Source: Esri, World Shaded
    ↪  Relief'
)
# Añadir marcadores al mapa basados en los datos de círculos.
sf = 8 # Factor de escala para el tamaño de los círculos.
name_number_list = []
for i, (lat, lon, weight, label, circle_color) in
↪  enumerate(circle_data, start=1):
    circle_radius = np.sqrt(weight) * sf  # Ajustar el factor de
    ↪  escala según sea necesario.
    # Añadir círculo al mapa.
    folium.CircleMarker(
        location=[lat, lon],
        radius=circle_radius,
        color='black',
        fill=True,
        fill_color='lightgray',
        fill_opacity=0.6,
    ).add_to(mexico_map)
```

```
    # Definir un mensaje con HTML para el pop-up, incluyendo el
    ↪ tamaño de la fuente.
    popup_message = f'<div style="font-size: 16px"><b>{i}:</b>
    ↪ {label}</div>'
    # Crear un marcador Folium en la ubicación del sitio.
    marker = folium.Marker(
        location=[lat, lon],
        popup=folium.Popup(popup_message, max_width=500),
        icon=folium.Icon(color=circle_color, icon="info-sign")
    )
    # Añadir el marcador al mapa.
    marker.add_to(mexico_map)
    name_number_list.append((i, label))
# Guardar el mapa en un archivo HTML.
iofile = "masses_map_try.html"
map_path = out_path / iofile
mexico_map.save(map_path)
```

Bibliografía

Albiez-Wieck, S. (2017). Albiez-Wieck, Sarah (2013): Contactos exteriores del Estado tarasco: Influencias desde dentro y fuera de Mesoamérica. Zamora: El Colegio de Michoacán. *kompetenznetz-lateinamerika*.

Barabási, A.-L. and Pósfai, M. (2016). *Network science*. Cambridge University Press, Cambridge.

Barnes, J. A. and Harary, F. (1983). Graph theory in network analysis. *Social Networks*, 5(2):235–244.

Borgatti, S. P., Mehra, A., Brass, D. J., and Labianca, G. (2009). Network Analysis in the Social Sciences. *Science*, 323(5916):892–895. Publisher: American Association for the Advancement of Science.

Choi, J. S., Lee, S., and Chun, S. J. (2021). A queueing network analysis of a hierarchical communication architecture for advanced metering infrastructure. *IEEE Transactions on Smart Grid*, 12(5):4318–4326.

De, J., Cheng, L., Zhang, X., Lin, F., Li, H., Ong, K. H., Yu, W., Yu, Y., and Ahmed, S. (2016). A graph-theoretical approach for tracing filamentary structures in neuronal and retinal images. *IEEE Transactions on Medical Imaging*, 35(1):257–272.

Eades, P. (1984). A heuristic for graph drawing. *Congressus Numerantium*, 42:149–160.

Erciyes, K. (2017). *Complex Networks: An Algorithmic Perspective*. CRC Press, Inc., USA, 1st edition.

Fruchterman, T. M. and Reingold, E. M. (1991). Graph drawing by force-directed placement. *Software: Practice and Experience*, 21(11):1129–1164.

Goldstein, H. and Twersky, V. (1952). Classical Mechanics. *Physics Today*, 5(9):19.

Hosler, D. (2005). *Los sonidos y colores del poder: la tecnología metalúrgica sagrada del occidente de México*. El Colegio Mexiquense. Google-Books-ID: HqPnAQAACAAJ.

Jackson, J. D. (1999). *Classical electrodynamics*. Wiley, New York, NY, 3rd ed. edition.

Jacomy, M., Venturini, T., Heymann, S., and Bastian, M. (2014). Forceatlas2, a continuous graph layout algorithm for handy network visualization designed for the gephi software. *PLOS ONE*, 9(6):1–12.

Křížek, M. (1995). Numerical experience with the three-body problem. *Journal of Computational and Applied Mathematics*, 63(1):403–409. Proceedings of the International Symposium on Mathematical Modelling and Computational Methods Modelling 94.

Mahmoud, H. and Akkari, N. (2016). Shortest path calculation: A comparative study for location-based recommender system. pages 1–5.

Mills, B. J. (2017). Social Network Analysis in Archaeology. *Annual Review of Anthropology*, 46(1):379–397. _eprint: https://doi.org/10.1146/annurev-anthro-102116-041423.

Moreno, J. (1953). Who shall survive?: A new approach to the problem of human interrelations.

Pendergast, D. M. (1962). Metal artifacts in prehispanic mesoamerica. *American Antiquity*, 27(4):520–545.

Perlsteinpollard, H. (2004). EL IMPERIO TARASCO EN ELMUNDO MESOAMERICANO. *Relaciones. Estudios de Historia y Sociedad*, (99):115–145.

Press, W. H., Flannery, B. P., Teukolsky, S. A., and Vetterling, W. T. (1992). *Numerical Recipes in FORTRAN 77: The Art of Scientific Computing*. Cambridge University Press, 2 edition.

Trenti, M. and Hut, P. (2008). Gravitational n-body simulations.

Yamamoto, S. and Makino, J. (2018). Hermite integrator for high-order mesh-free schemes. *Publications of the Astronomical Society of Japan*, 71(1).