UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE CIENCIAS

ANALOGÍA TEMPORAL ENTRE INTENSIDAD SÍSMICA Y NÚMERO DE HOMICIDIOS

T E S I S

QUE PARA OBTENER EL TÍTULO DE:

FÍSICO

P R E S E N T A:

ARTURO NÁJERA SANTOS

DIRECTOR DE TESIS:

Dr. MARCELO DEL CASTILLO MUSSOT

CIUDAD UNIVERSITARIA, CD. MX., 2019

1. Datos del alumno
Nájera
Santos
Arturo
55 6235 8131
Universidad Nacional Autónoma de México
Facultad de Ciencias
Física
309213798

2. Datos del tutor
Dr
Marcelo
Del Castillo
Mussot

3. Datos del sinodal 1
Dr
Adonis
Germinal
Cocho
Gil

4. Datos del sinodal 2
Dr
Ruben
Yvan
Maarten
Fossion

5. Datos del sinodal 3
Dr
Octavio
Reymundo
Miramontes
Vidal

6. Datos del sinodal 4
Dr
Jesús
Espinal
Enríquez

7. Datos del trabajo escrito.
Analogía temporal entre intensidad sísmica y número de homicidios
pag 75
2020

Dedicatoria

A mis padres y a mi hermana que me alegra los días.

Agradecimientos

Estoy agradecido con todas las personas que ayudaron a hacer esto posible.

Contenido

Introducción

En el gobierno de Felipe Calderón se tomó la decisión de usar a los militares como cambio en la estrategia contra el narcotráfico. El periodo de gobierno fue desde 1 de diciembre del 2006 hasta 30 de noviembre del 2011. Durante este periodo hubo un incremento en la violencia relacionadas con el narcotráfico. Este incremento no había tenido precedentes. En algunas partes del país redujo la expectativa de vida. Aunque no todo el aumento en homicidios es atribuible a la violencia relacionada con las drogas, en gran parte sí lo es como lo indica el artículo (Las Comisiones The Lancet, 2017)

Durante este periodo se hizo un registro de hecho violentos con los cuales se formó una base de datos. El Programa de Política de Drogas del CIDE, en colaboración con CentroGeo, sometieron a la base de datos[1] a un proceso de validación de los eventos registrados contrastando con información pública en internet. También se codificó algunos campos para garantizar la seguridad de las personas involucradas en los eventos registrados (*Documento_Descriptivo.pdf*, s/f, p.10). Esta base de datos procesada es la que se utiliza aquí.

Para cada evento registrado en la base de datos se tienen la fecha del día en que ocurrió, municipio, estado y número de muertos. Estas son las variables con las que se disponen.

Objetivo general: Averiguar si existe algún patrón temporal en el número de homicidios de la llamada guerra contra las drogas en México.

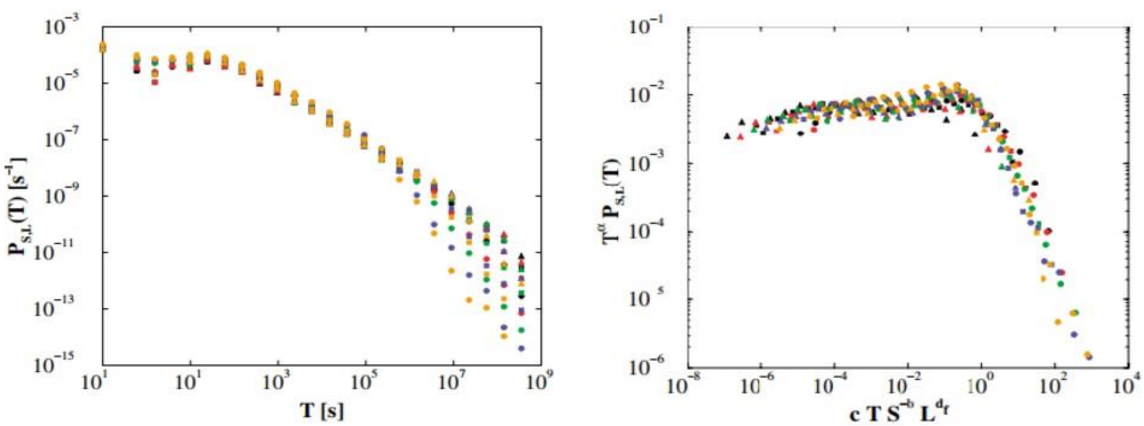
Se plantea la siguiente hipótesis.

El tiempo entre eventos sigue el mismo tipo de distribución que sismos lo cual permitirá describir su describir su distribución entre eventos con el mismo tipo de re-escalamiento lo cual permitirá obtener una expresión matemática con la cual representar los datos.

[1] En el documento Documento_descriptivo.pdf se hace referencia a esta base de datos como Base Madre.

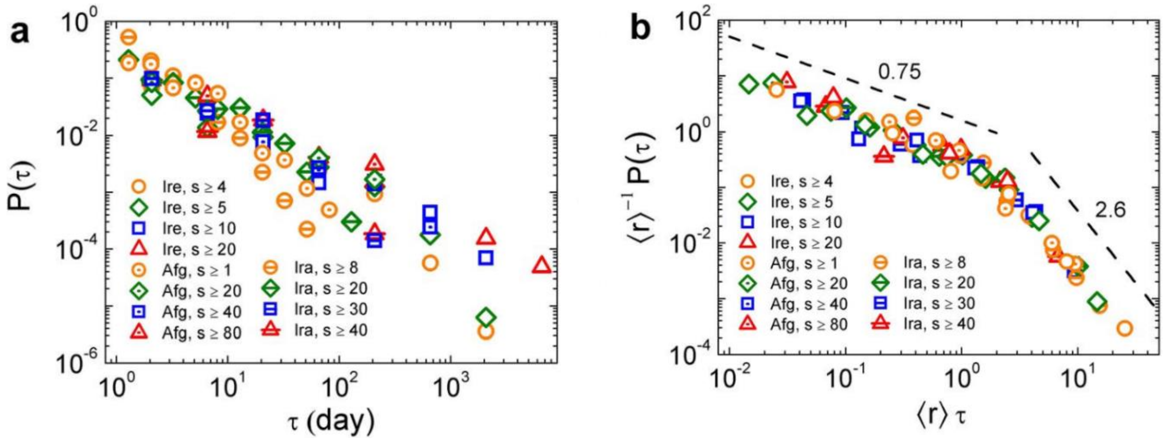
A través de este trabajo se encuentra que al contrario de lo planteado en la hipótesis la guerra contra no se comporta exactamente como sismos. En las conclusiones finales se tratará de interpretar lo obtenido.

En el artículo (Bak et al., 2002) se hace un análisis espaciotemporal para sismos. En este análisis se cubre una región con una cuadrícula con celdas de $L \times L$ y se define un tiempo T como el tiempo entre el inicio de dos sismos consecutivos. Entonces se mide la distribución $P_{S,L}(T)$, de tiempo de espera dentro de un rango L, cuyos valores de magnitud sean mayores o iguales a $m = \log(S)$. La restricción también se puede poner directamente en S. La distribución normalizada se muestra en la Figura 0.1a.



(a) Sin escalar *(b) Escalada*

Figura 0.1: Distribución de tiempo entre sismos. Los sub-indices S y L en $P_{S,L}(T)$ identifican la distribución correspondiente a una magnitud mínima S en sísmos y la cuadrícula de longitud L. Reimpreso de Bak et al., (2002). "Unified Scaling Law for Earthquakes".



(a) Distribución de tiempo entre eventos *(b) Distribución escalada de tiempo entre eventos*

Figura 0.2: Distribuciones de tiempo eventos encontrados en guerras de Irak, Irlanda del Norte y Afganistán. El tipo de figura corresponde a el país, los colores corresponden a la magnitud mínima s. Reimpreso de Picoli et al., (2014). "Universal bursty behaviour in human violent conflicts".

En la Figura 0.1a se muestra la distribución normalizada en el eje vertical como función del tiempo entre sismos, donde los símbolos del mismo color indican una sola longitud. Se hace para varios valores de intensidad mínima.

En la Figura 0.1: Distribución de tiempo entre sismosFigura 0.1b se muestran los mismos datos, pero re-escalados con nuevas variables horizontales y verticales. El factor $S^{-b}L^{d_f}$ es una medida del número promedio de sismos por unidad de tiempo. Se nota que se puede ajustar, más o menos, dos rectas a los mismos datos re-escalados. Una recta en una gráfica log-log representa una ley de potencia $e^{b}x^{-m}$ también conocida como distribución de Pareto (en el caso de que m sea positiva). En el Capítulo 5 se describe como utilizar un procedimiento similar en el caso de los fallecimientos por narcotráfico en México.

En los artículos (S. Picoli et al., 2014; Sergio Picoli et al., 2017) se realiza este tipo de análisis para las guerras en Afganistán, Irak e Irlanda del Norte en diferentes épocas. Obtienen la gráfica mostrada en Figura 0.2. En el caso presentado aquí los fallecidos están relacionados con el narcotráfico. Uno de los objetivos principales de esta tesis es determinar si en el caso de México se obtiene un comportamiento similar a los casos de Afganistán, Irak e Irlanda de Norte.

Del Capítulo 1 a Capítulo 4 se discuten los métodos utilizados para tratar de forma sistemática las distribuciones. Los aspectos a tratar de estos son: la ubicación del punto donde se separan las dos rectas, seleccionar la magnitud de s mínimo (número de muertos), el número de *bins* en las distribuciones y el manejo de eventos simultáneos que aparecen en los datos.

Una vez determinados los métodos a utilizar en el Capítulo 5 se procede a obtener los resultados. Se comparan los resultados con estudios previos para mostrar similitudes. El tipo de resultados mostrados son el número de muertos por día, la distribución en tiempo entre eventos, y su respectivo escalamiento. Se hace para determinar que el objeto de estudio cumple la hipótesis, es decir que existe una forma de describir el tiempo entre eventos.

En los capítulos 6 a 7 se exploran formas para ver si se puede mejorar los resultados. Esto se trata a través de simplificar el modelo planteado en el capítulo 5. Se utiliza una sola ley de potencias, después se toma en cuenta el número de habitantes para normalizar el número de muertos.

En el capítulo 8 están las conclusiones finales.

Descripción de las categorías

Las bases de datos utilizadas aquí son (Atuesta et al., 2016b, 2016c, 2016a). La descripción de las categorías utilizadas se hace referencia en (*Documento_Descriptivo.pdf*, s/f) y son las siguientes:

- **Ejecuciones**: homicidio doloso "cuya víctima y/o victimario es presumiblemente miembro de algún grupo criminal. No es resultado de un enfrentamiento ni de una agresión, tal como se define en el presente documento. Tampoco presupone la participación de autoridad alguna".
- **Enfrentamientos**: Por Enfrentamientos, el documento define los actos violentos perpetrados por presuntos criminales en contra de autoridades, víctimas o eventos que perturban el orden público y que se realizan "mediante el uso de armas de fuego y equipo militar", eventos en los que las propias fuerzas públicas hacen uso de las armas de fuego (en contra de

presuntos delincuentes, se entiende), o choques entre grupos delincuenciales o dentro de éstos.

- **Agresiones**: Ataques de organizaciones delictivas en contra de instalaciones gubernamentales o bien contra funcionarios públicos encargados de la seguridad, sin que la autoridad "tenga posibilidades de responder de forma armada". Estos eventos se refieren a acciones "dirigidas a personas u objetivos específicos".

Capítulo 1

Punto de separación entre dos curvas

La curva que se quiere ajustar a la distribución es una conformada por dos rectas porque las rectas en log-log forma leyes de potencia..

Se espera que se forme una recta en una gráfica log-log los puntos en la forma $(log(x),\ log(f))$. Si los puntos pertenecen a la recta se deduce que deben tener la siguiente forma.

$$\begin{aligned} \log(f) &= m\log(x)\ + b \\ f &= \exp(m\log(x) + b) \\ &= e^{b}\exp(\log(xm)) \\ &= e^{b}x^{m} \end{aligned}$$

m es la pendiente.

Se tiene que determinar en qué punto se separa una de otra, x_{sep}. La posición de este punto influye en que tan bien se ajustan las rectas a la distribución.

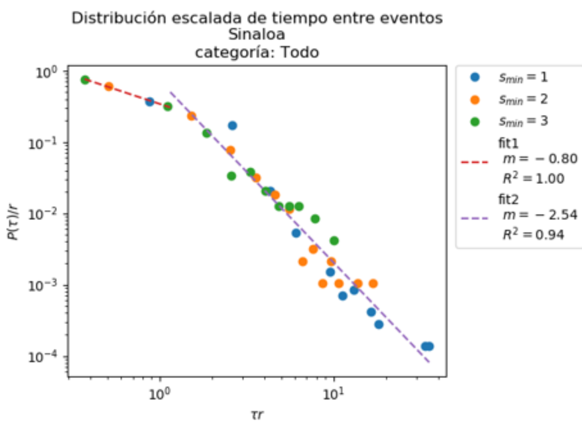
Método

Para seleccionar los resultados de forma apropiada se selecciona el punto que refleje un mayor coeficiente de determinación R^2. La curva se separa en dos rectas, se puede dar prioridad a la primera parte o la segunda. Al considerar cualquiera de las R^2 surgen inconvenientes. Si se selecciona x_{sep} maximizando R^2 de una sola recta, se podría obtener un mal valor para la otra recta. Se tiene que cuidar que le valor de R^2 asociado a alguna de las rectas no aumente por seleccionar dos puntos. Para considerar ambos valores de R^2 se suman. Se propone la suma porque es una función simple.

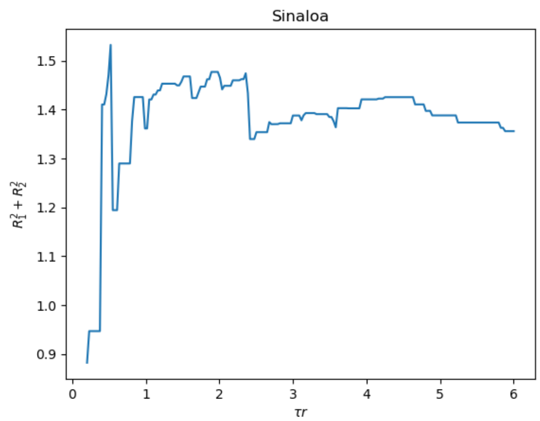
Se toma cualquiera de los criterios como una función que dependa del punto de separación de las rectas. Para el caso de la suma sería $f(x_{sep}) = R_1^2\,(x_{sep}) + R_2^2\,(x_{sep})$, para la cabeza de la curva $f(x_{sep}) = R_1^2\,(x_{sep})$ y para la cola de la curva $f(x_{sep}) = R_2^2\,(x_{sep})$ donde $R_1^2\,(x_{sep})$ y $R_2^2\,(x_{sep})$ son los coeficientes de determinación como funciones del punto que separa ambas rectas, x_{sep}.

No se puede maximizar la suma de los coeficientes de determinación por métodos númericos directamente debido a que esta curva es plana por partes. Esto se observa al graficar $f(x_{sep}) = R_1^2\,(x_{sep}) + R_2^2\,(x_{sep})$ como se muestra en la Figura 1.1 en la sección de resultados. Es inmediato que sus derivadas sean cero en todos los puntos. También para cuando se enfoca en la cabeza o cola de la curva.

La regresión para una recta solo depende de los puntos que se consideran hasta (o desde) x_{sep}. Evaluar múltiples veces $f(x_{sep})$ entre puntos lleva a formar las regiones planas. Así que evaluar una vez entre cada par de puntos consecutivos determina todos los posibles valores para $f(x_{sep})$.



(a) Ejemplo de distribución. Detalles de esta distribución se encuentran en capítulos posteriores.



(b) Grafica de la suma de los valores R^2 de ambas rectas. En el eje x es el punto donde se separan las rectas, en y es la suma.

Figura 1.1 Grafica de la suma de los valores R^2 de ambas rectas para la distribución de Sinaloa en la categoría Todo.

En la Figura 1.1 la curva se puede ver que está formada por partes planas. En las regiones donde los puntos están muy cerca se aprecia el cambio de $f(x_{sep})$. Esta curva puede variar dependiendo del número de bins y s.

Conclusiones

Se considera la suma $f(x_{sep}) = R_1^2\,(x_{sep}) + R_2^2\,(x_{sep})$ como la función a maximizar para obtener el valor del punto donde se separan las rectas, x_{sep}. Se maximiza $f(x_{sep})$ comparando valores de f obtenidos de evaluar x_{sep} entre cada punto de la distribución.

Capítulo 2 Cota mínima en la magnitud de los datos

Otro aspecto a considerar es que como se van haciendo cortes. A la sección de datos usada que cumple con una magnitud mínima s se le llama corte. La cantidad de puntos obtenidos varía al hacer cortes. También cambia la cantidad de datos respecto de lo que se esté tratando (municipio, estado, país).

Al hacer las distribuciones re-escaladas se generan los puntos sobre los cuales se hará regresión. Resulta conveniente aumentar la cantidad de puntos para hacer regresión. Así que se busca determinar cuántos cortes se puede hacer a los datos.

Método

Para visualizar el comportamiento se describe el perfil del valor $R_1^2 + R_2^2$ al variar el punto donde se separan las curvas. Este perfil es para un s y número de bins fijos. Se juntan perfiles para distintos s en un solo gráfico. De esta forma se puede comparar como varían los perfiles respecto al número de cortes, s.

También se hace una tabla de percentiles con el número de muertos para mostrar el límite de los cortes que se pueden hacer.

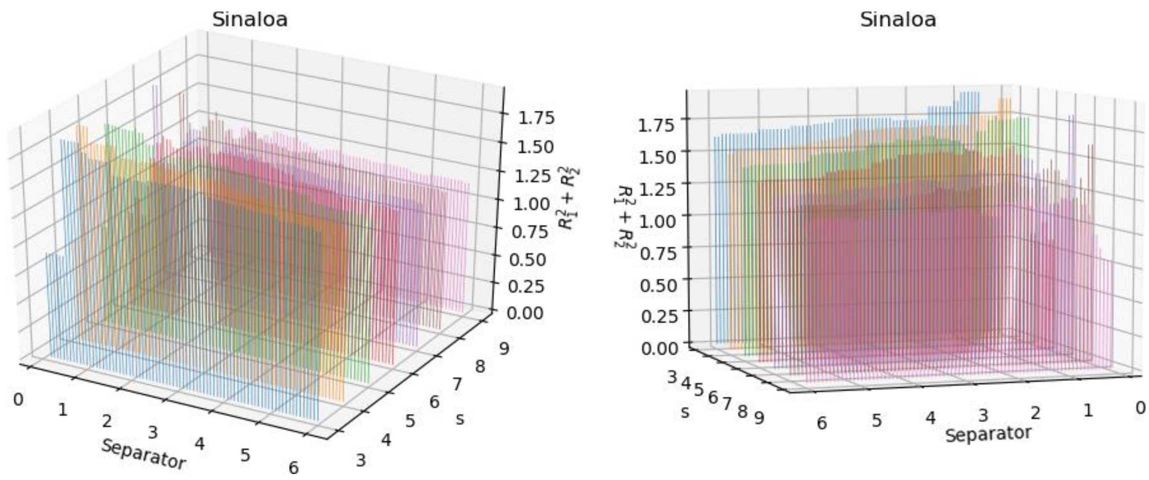
Resultados

La Tabla 2.1 contiene los percentiles del número de muertos. Esta tabla se hace con la base de datos completa.

Tabla 2.1: Percentiles de número de muertos por evento. El número de muertos limita por arriba al menos el porcentaje mostrado del total de datos.

Percentiles	Número de muertos por evento
90 %	2
95 %	3
99 %	6
100 %	72

En la Figura 2.1 se muestra el perfil de la suma de la regresión para diferentes cortes con s_{min}. El gráfico corresponde al estado de Sinaloa en la categoría Todos. Cada figura muestra la misma grafica en un distinto ángulo.



a) b)

Figura 2.1: En a) y b) se muestran gráficas agrupadas donde se puede comparar su magnitud. Se muestran diferentes vistas de la gráfica de la suma de R^2. Las variables en esta gráfica son el punto donde se separan y el número de cortes s.

Discusión

En la Figura 2.1a La amplitud de las curvas decrece con la cantidad de cortes que se hacen. En la Figura 2.1b se nota mejor que cerca de 0 hay picos. Significa que el punto en el cual se separan las rectas está cerca del cero por lo que se incluyen menos puntos en la primera recta. En la situación en la que solo hay 2 puntos en la primera recta se obtiene una regresión con $R^2 = 1$ dado que por dos puntos siempre se puede trazar una recta. Con lo cual es importante evitar esta situación cuando se trata de obtener el máximo para cualquiera de los criterios y encontrar el mejor ajuste posible.

En la Tabla 2.1 los percentiles muestran que el 95% de los datos tiene a lo más tres muertos por evento. Para una distribución con todos los datos si se pone una cota mínima superior a 3 menos del 5% formarían parte de esta. Agregar distribuciones con datos insuficientes podría disminuir el valor de R^2. De aquí se concluye que para generalizar el número de cortes, a municipios o estados, este no puede ser muy grande.

En las gráficas de la Figura 2.1 se muestran cortes desde 3 hasta 9. Es de esperar que al agregar distribuciones sin suficientes datos disminuyan los valores de R^2. Para formar las distribuciones es recomendable que el número de cortes sea alrededor de 3.

Conclusiones

El 95% de eventos registrados tiene a lo más 3 muertos. El aumentar a más de 3 puede disminuir los valores de R^2 por falta de datos. En lo que sigue se usa al menos 3 cortes para formar distribuciones.

Capítulo 3 Número de bins en distribuciones

El número de bins escogidos para la distribución puede variar los valores de R^2 obtenidos. Si se tienen demasiados bins en relación a los puntos cada punto podría quedar en un solo bin aplanando así la distribución. Si se tienen pocos bins se podría tener una distribución no muy bien definida. La cantidad de gráficas posibles, municipios y estados del país, hace difícil poder tratarlas de forma individual. Se busca tratar todos los conjuntos de datos de la misma manera. Se muestra cómo se abordó esto.

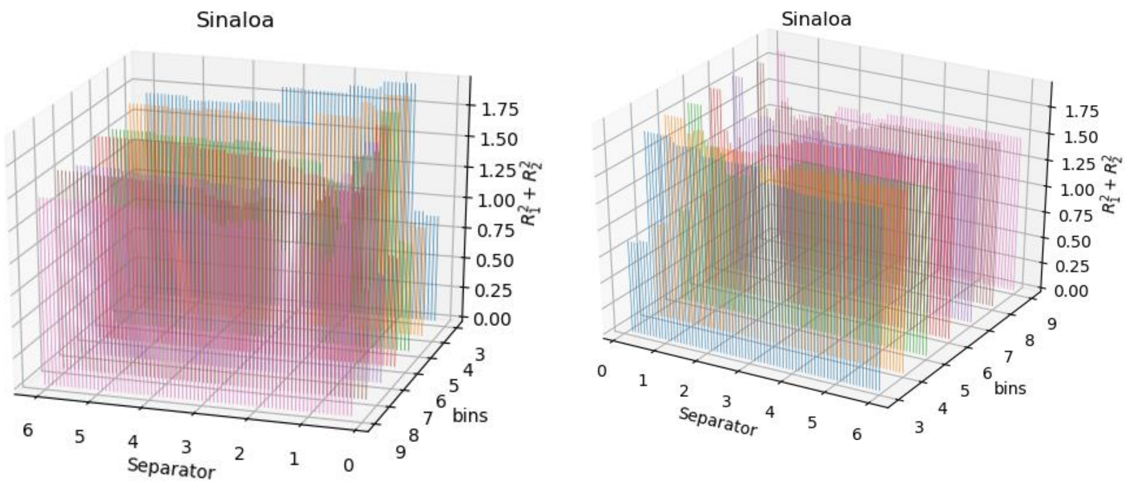
Métodos

Para visualizar el comportamiento se describe el perfil del valor $R_1^2 + R_2^2$ al variar el punto donde se separan las curvas. Este perfil es para un s y número de bins fijos. Se juntan perfiles para distintos bins en un solo gráfico. De esta forma se puede comparar como varían los perfiles respecto al número de bins y si la ubicación del máximo depende del número de bins.

Existen métodos para estimar el número de bins utilizados en una distribución. La Tabla A.1 se muestran métodos disponibles en numpy (*numpy.histogram_bin_edges—NumPy v1.15 Manual*, s/f) con la descripción breve que aparece en la documentación. Con esta tabla se hacen unas observaciones: Los valores del método “Scott” son muy similares a los de Freedman-Diaconis (“fd”) pero “fd” es resiliente a valores atípicos. Basta con probar “fd”. “sturges’” es óptimo para distribuciones normales, pero para los datos que se muestran aquí no es el caso pues forman una ley de potencias. ‘rice’ y ‘sqrt’ solo toman en cuenta la cantidad de datos. De esta forma los métodos que interesan comparar son “fd” y “doane”.

Resultados

En la Figura 3.1 se muestra el perfil de la suma de la regresión para diferentes números de bins. Cada figura muestra un ángulo distinto de la misma grafica.



(a) (b)

Figura 3.1: En a) y b) se muestran gráficas agrupadas donde se puede comparar su magnitud. Se muestran diferentes vistas de la gráfica de la suma de R^2. Las variables en esta gráfica son el punto donde se separan y el número de bins.

En las tablas Tabla 3.1, Tabla 3.2 se encuentran los resultados de comparar los métodos para la primera y segunda parte de la gráfica. Las tablas de comparación en otros capítulos con respecto al método “fd” tienen porcentajes muy similares, por esta razón no se agregan.

En la Tabla 3.1 se compara los valores obtenidos R^2 de diversos los estados. Las columnas muestran el conteo de resultados para los 12 estados con mayor número de muertos, luego para 26 estados con más muertos y finalmente todos los estados. Se selecciona un grupo de estados y para cada grupo se comparan los valores. Los resultados muestran el porcentaje del grupo que muestran el porcentaje del grupo donde el valor R^2 asociado a "fd" es mayor (igual o menor) al de "doane". El tipo de comparación es indicado al principio del renglón. La comparación es por estados porque se puede hacer para todas las gráficas aquí. La separación es por orden de magnitud. Los primeros doce es hasta Veracruz con 1208, los primeros 26 hasta Puebla con 178, y primeros 32 hasta Tlaxcala con 21. De esta forma al mostrar los primeros 32 se sabe lo que pasa con todos los datos y también se tiene una perspectiva de los municipios más violentos. En otras tablas se hace este tipo de comparación.

Al comparar los métodos

Tabla 3.1: Comparación de valores de R² para la primera recta. La notación es la siguiente: R_fd^2 corresponde a valores obtenidos método "fd", mientras que R_doane^2 corresponde a valores con el método "doane", finalmente "nulo" estados donde no haya sido posible comparar debido a que no se haya podido hacer una distribución. La comparación corresponde a Estados ordenados por cantidad de eventos.

Comparación	Primeros 12	Primeros 26	Primeros 32
$R^2_{fd} > R^2_{doane}$	8.33%	11.54%	21.875%
$R^2_{fd} < R^2_{doane}$	83.33%	84.62%	71.875%
$R^2_{fd} = R^2_{doane}$	0.0%	0.0%	0.0%
nulo:	8.33%	3.85%	6.25%

Tabla 3.2: Comparación de valores de R² para la segunda recta. La notación es la siguiente: R_fd^2 corresponde a valores obtenidos método "fd", mientras que R_doane^2 corresponde a valores con el método "doane", finalmente "nulo" estados donde no haya sido posible comparar debido a que no se haya podido hacer una distribución. La comparación corresponde a Estados ordenados por cantidad de eventos.

Comparación	Primeros 12	Primeros 26	Primeros 32
$R^2_{fd} > R^2_{doane}$	0.0%	11.54%	5.625%
$R^2_{fd} < R^2_{doane}$	100.0%	84.62%	78.125%
$R^2_{fd} = R^2_{doane}$	0.0%	0.0%	0.0%
null:	0.0%	3.85%	6.25%

Discusión

En la Figura 3.1a se ve que el perfil al cambiar en número de bins no cambia de forma de una cantidad a otra. Se muestra que aparentemente hay un máximo. En la Figura 3.1b la magnitud que alcanzan todos los perfiles es prácticamente la misma. Esto da una idea de los bins y su relación con R^2.

Aun cuando se hizo lo anterior para algunos estados para dar una idea para el comportamiento se requiere determinar el número de bins sistemáticamente. Usar el método de estimación "doane" funciona en la mayoría de los casos. En la primera parte de la recta 83.33%, esta suele tener menos puntos que la segunda, en la segunda parte es del 100.0% de los casos. Estas tablas se hacen a escala de estados porque hay muchos municipios se municipios con pocos puntos y en la comparación no serían muy relevantes. Al agruparlos por estados se tienen más puntos por distribución.

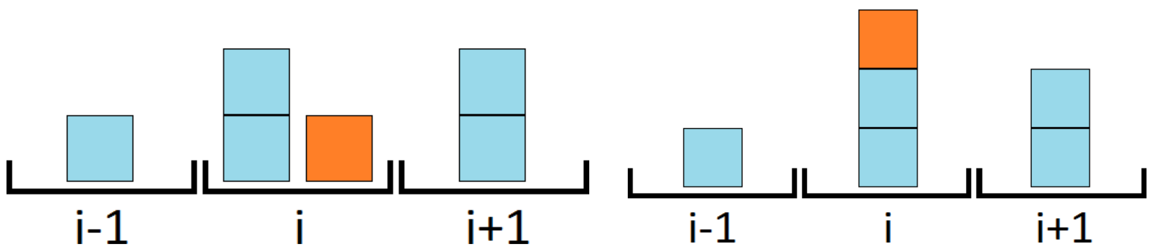
Conclusiones

Se puede determinar que el número de bins no cambia el perfil de la curva. Hay ligeras variaciones entre el perfil de las curvas, pero es casi el mismo. Para evitar el problema descrito al inicio se usa el método “doane” para el resto de las distribuciones en capítulos posteriores. Con este método el coeficiente de determinación es mejor en la mayoría de los casos.

Capítulo 4

Manejo de datos de eventos simultáneos

Para hacer una gráfica de muertos por día en una región se suma el número total de muertos en cada día. Valdría preguntarse ¿Se suma el número de muertes por día cuando se hacen las distribuciones o se dejan tal cual? Se discute esto aquí. En la Figura 4.1 se muestra un ejemplo de la diferencia producida al considerar la suma de magnitudes.



(a) hipótesis 0: Los eventos se dejan tal cual.

(b) hipótesis 1: La magnitud de eventos simultáneos se suma.

Figura 4.1: Se muestra una representación de cómo afecta sumar la magnitud de eventos simultáneos. Las cajas representan días consecutivos, marcados con i. Cada pila de cuadros representa un evento, con cada cuadro un muerto correspondiente a ese evento.

Estas hipótesis son consideradas porque no son muy complejas.

De forma explícita las hipótesis son:

hipótesis 0: Los eventos se toman tal cual, aun cuando haya eventos simultáneos.

hipótesis 1: Se suma la magnitud de los eventos simultáneos y son considerados como uno solo.

Lo que se cuenta en la distribución es el tiempo entre eventos consecutivos. Al ordenarlos por fecha los que son simultáneos quedan juntos. La diferencia entre estos eventos es cero (la resolución mínima es 1 día). Para hacer la gráfica log-log no son considerados los tiempos iguales a 0.

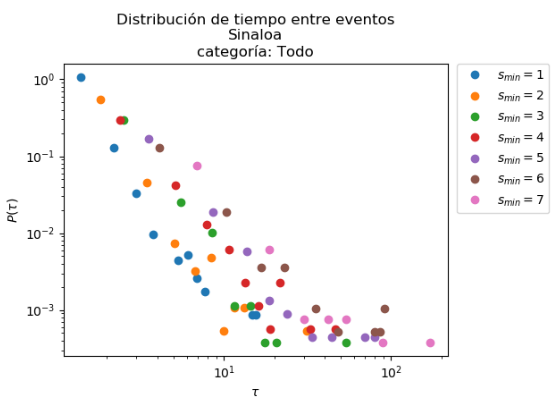
La diferencia recae en que la magnitud de un evento formado con la hipótesis 1 su magnitud es mayor a la los eventos individuales, porque es la suma de los eventos individuales. Al aumentar la magnitud de un evento este podría alcanzar una nueva s_{min} pasando a formar parte de una nueva distribución que antes no.

Método

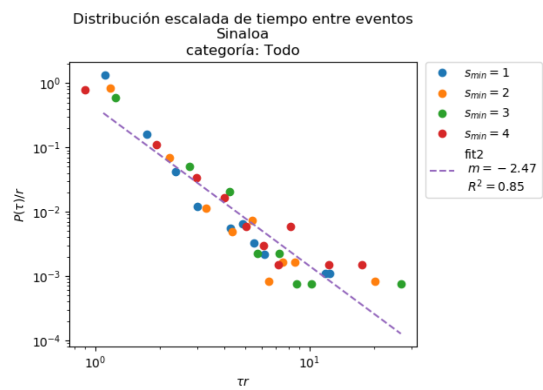
Se comparan los resultados de ambas hipótesis para determinar con cual se puede ajustar mejor la curva.

Resultados

Se hacen gráficas para la hipótesis 1 correspondiente a la suma de magnitud de eventos simultáneos. En la Figura 4.2a está las gráficas de la distribución sin escalar de los datos para Sinaloa en la categoría de ejecuciones. En la Figura 4.2b se hace el escalamiento con r.



a) Distribución de tiempo entre eventos sin escalar para hipótesis 1.



b) Distribución de tiempo entre eventos escalado para hipótesis 1.

Figura 4.2: Distribuciones de tiempo para hipótesis 1 para Sinaloa. τ es tiempo entre evento, $P(\tau)$ la distribución de tiempo entre eventos, los colores identifican a la distribución con una cota mínima s.

En la Tabla 4.1se compara los resultados para los valores de R_2^2 en ambas hipótesis. La R_2^2 debido a que es la parte que contiene más puntos. La gráfica contiene la comparación de estados.

Tabla 4.1: Comparación de valor de R^2 para la segunda recta. Se comparan las hipótesis 0 y 1. La notación es la siguiente: R_{hip0}^2 corresponde a valores con la hipótesis 0, mientras que R_{hip1}^2 corresponde a valores con la hipótesis 1, finalmente "nulo" estados donde no haya sido posible comparar debido a que no se haya podido hacer una distribución. La comparación corresponde a Estados ordenados por cantidad de eventos.

	Primeros 12	Primeros 26	Primeros 32

$R^2_{hip0} > R^2_{hip1}$	58.33%	57.69%	56.25%
$R^2_{hip0} < R^2_{hip1}$	33.33%	34.62%	34.375%
$R^2_{hip0} = R^2_{hip1}$	0.0%	0.0%	3.125%
Nulo	8.33%	7.69%	6.25%

Discusión

En la Figura 4.2 se muestra el número de muertos por día para poder mostrar la curva acumulada. El número de muertos totales es representativo de la hipótesis 1. Si se hiciera el gráfico con los valores de magnitud con la hipótesis 0, solo se vería el cambio en la altura de las líneas a las que corresponden eventos simultáneos. En relación al tiempo, si se tomara el tiempo entre eventos con $s_{min} = 1$ no habría diferencia entre el conjunto de τ obtenido entre las hipótesis, pero al aumentar $s_{min} \geq 2$ los conjuntos empezarían a ser distintos. Debido a la cantidad de cortes, que no es muy grande, puede no verse gran diferencia.

Conclusiones

Aunque las gráficas son parecidas se encuentra que la hipótesis 0 es mejor en el 56.25% mientras que solo es mejor para el 34.37% con la hipótesis 1. Por lo cual no se sumará la magnitud de los eventos para procesar el análisis.

Capítulo 5

Exploración de hipótesis de universalidad con dos leyes de potencia

Con las consideraciones hechas en los Capítulo 1 al 5 se obtuvo los métodos para tratar las distribuciones. Antes de proceder a obtener los resultados principales se toma una característica más de los datos, el número de muertos por día.

Se escogió Sinaloa como ejemplo de resultados porque está entre los municipios más violentos.

Preprocesamiento de datos

Cada categoría no contiene la misma cantidad de datos para hacer las distribuciones. En la Tabla 5.1 se muestra cómo se distribuyen la cantidad de datos en estas. En la Tabla A.2 se muestra información estadística de los datos. Con esto se sabe que tampoco el número de eventos es uniforme en los estados. De aquí que para la discusión se usa un estado y una categoría con suficientes datos. Resultados para otros estados y categorías se muestran en apéndices o se pueden obtener haciendo uso del código.

Tabla 5.1: Distribución de datos en las distintas categorías.

Ejecuciones	85.2 %
Enfrentamientos	10.5 %
Agresiones	4.3 %
Todo	100 %

La base de datos es extensa. Los datos están registrados por día. Eventos ocurridos en un mismo día serán considerados simultáneos debido a que no hay mayor precisión en tiempo.

Métodos

Procediendo con la analogía entre intensidad de sismos y número de fallecidos se puede definir una “intensidad” s y una cota inferior s_{min}. Se toman en cuenta los eventos (valores s) tales que $s \geq s_{min}$. Con estos valores se hizo una distribución $P(\tau)$ del tiempo entre eventos donde τ es el tiempo entre eventos consecutivos. Se hizo una gráfica en log-log, donde en el eje horizontal se tiene el tiempo τ y en el vertical $P(\tau)$.

También se presentan los mismos datos re-escalando con las nuevas variables horizontales y verticales, τr y $P\ (\tau)\ /\ r$, donde el factor r es el número promedio de eventos por unidad de tiempo. Los eventos tienen magnitud $s \geq s_{min}$. Se escala de la misma manera que en el artículo (S. Picoli et al., 2014). $\tau\, r$ es la variable re-escalda. En el caso de sismos la variable re-escalada no tiene un significado absoluto es de esperar que con muertos tampoco.

Resultados

En la Tabla A.3 se muestran algunos resultados de estados y municipios. Se pone como ejemplo representativo el estado de Sinaloa. La gráfica correspondiente a su municipio y a México (país) se encuentra en los apéndices. En las figuras se encuentran gráficas solo para Navolato, Sinaloa y México (país).

La categoría Todo se refiere a todos los datos juntos de Agresiones, Enfrentamientos y Ejecuciones. Son consideradas todas juntas porque son parte del mismo conflicto (la misma temática), el narcotráfico.

No se hacen graficas de algunos estados o municipios debido a que:

- No se hacen distribuciones cuando la distribución no contiene datos.
- En una gráfica no se hace regresión lineal cuando hay menos de dos datos correspondientes a la distribución.
- La cantidad de graficas en el anexo se vuelve muy extenso.

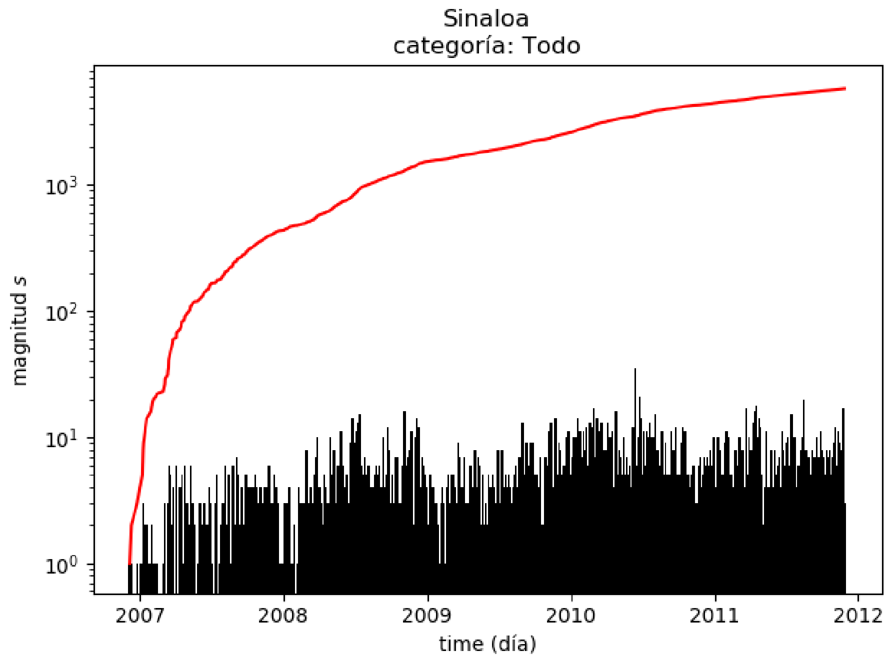
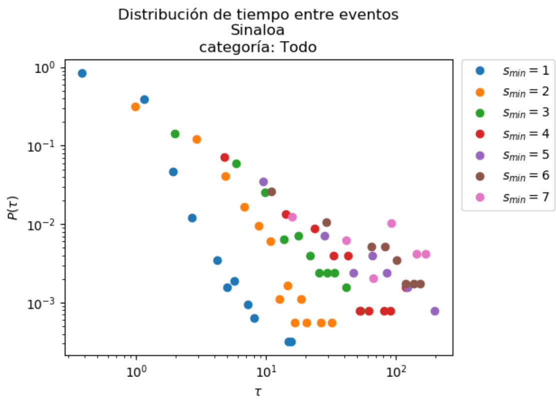


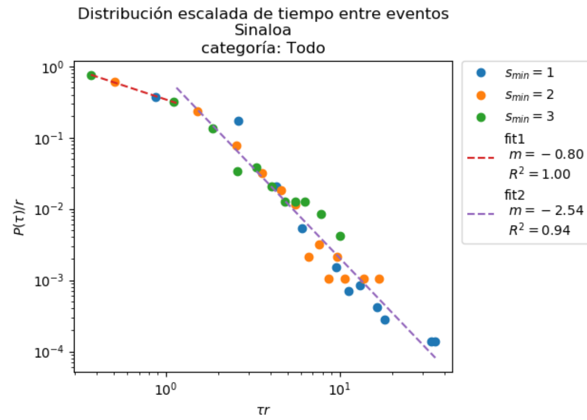
Figura 5.1: Número de muertos por día en el estado de Sinaloa. La curva roja es la curva acumulativa del número de muertos.

En la Figura 5.1 se muestra el número de muertos por día para el estado de Sinaloa y la curva acumulativa de muertos.

En la Figura 5.2 se muestran las distribuciones de tiempo entre eventos con y sin escalar. Cada conjunto de puntos de cada color representa una distribución a s_{min}. En las gráficas que son como Figura 5.2b la regresión lineal se hace con todos los puntos que se muestran. Para poder comparar municipios y estados, se hacen cortes bajos.



(a) Distribución de tiempo entre eventos sin escalar



(b) Distribución escalada de tiempo entre eventos

Figura 5.2: Distribuciones de tiempo entre eventos para Sinaloa de la base de Todo. τ es tiempo entre evento, $P(\tau)$ la distribución de tiempo entre eventos, los colores identifican a la distribución con una cota mínima s.

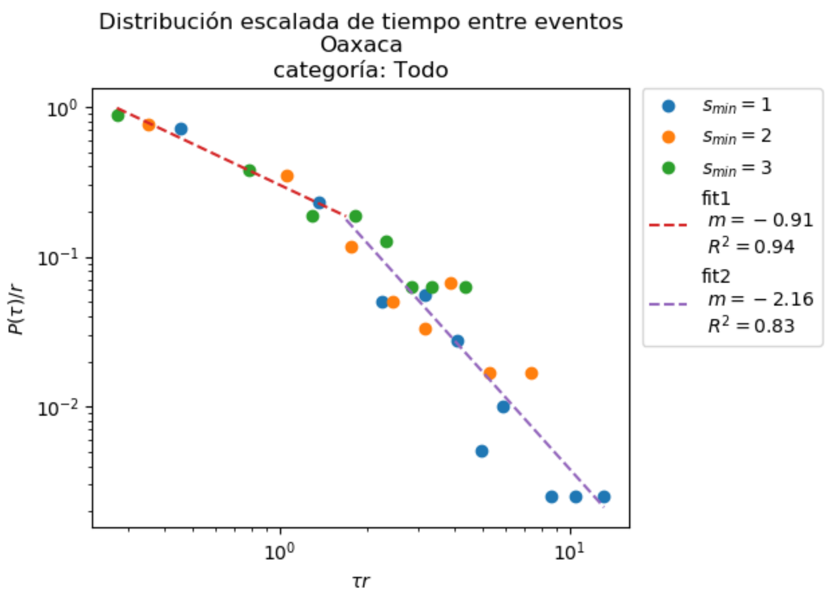


Figura 5.3: Distribución de tiempo entre eventos para el estado de Oaxaca en la categoría Todo. τ es tiempo entre evento, $P(\tau)/r$ la distribución de tiempo entre eventos, los colores identifican a la distribución con una cota mínima s, r es el número promedio de eventos por unidad de tiempo.

Discusión

En la Figura 5.1 se observa la secuencia de eventos para cada día y la magnitud de estos. En esta se muestra la curva acumulativa como dato de interés. Para mostrar esta curva se suma la magnitud de los eventos simultáneos. Teniendo en cuenta que la magnitud de los eventos no suele ser muy grande los eventos simultáneos deben ser comunes.

La grafica en la Figura 5.2a muestra cómo las distribuciones se van separando al aumentar el tiempo entre eventos. En el caso de sismos sucede de la misma manera, las distribuciones se separan Figura 0.1. La separación se puede notar al seguir cualquiera de los colores. Estos se ubican juntos, pero se van separando las curvas. Si se toma la distribución correspondiente a $s_{min} = 1$ se puede ver que los puntos están más a la izquierda que los demás. Para $s_{min} = 3$ se ve todavía una curva, pero ésta se empieza a mezclar con las curvas de $s_{min} = 4, 5, 6, 7$. De los cortes mayores se alcanza a distinguir la curva para el rojo pero no es muy distinto de $s_{min} = 5, 6, 7$, salvo justo al final de la curva.

La distribución de $s_{min} = 1$ se encuentra a la izquierda. Significa que el tiempo entre eventos es menor, comparando con las demás. Las distribuciones correspondientes a $s_{min} = 5, 6, 7$, que tienen una magnitud mayor, están a la derecha. Esto indica que el tiempo entre sus eventos es mayor. El tiempo entre eventos incrementa conforme s_{min} aumenta.

Es importante notar que se pueden ajustar rectas a las gráficas con los datos re-escalados. Las rectas en una gráfica log-log son leyes de potencia del tipo $e^b x^m$ donde b y m son constantes.

Las distribuciones obtenidas en la Figura 5.2 son comparables con las mostradas en las figuras Figura 0.1. Hay que tomar en cuenta que no se tiene el mismo número de cortes debido a que el número de muertos para la mayoría de eventos no es grande. El tiempo entre eventos alcanza varios ordenes de magnitud al igual que los valores de la distribución.

Al comparar con los resultados del artículo (S. Picoli et al., 2014) se ve que a pesar de que las causas de las muertes en los casos de Afganistán, Irak e Irlanda de Norte fueron guerras convencionales o de guerrillas mientras que en México se deben a agresiones, ejecuciones y enfrentamientos relacionados con el narcotráfico, los comportamientos son leyes de potencia. En las bases de datos estas tres causas de fallecimientos están separadas, lo cual permitirá hacer tres estudios independientes y otro más global juntando las tres categorías.

En las gráficas para las distribuciones escaladas, al aproximar dos rectas la primera se hace con pocos puntos. Esto se discute en un capítulo posterior.

Comportamiento general

En los resultados obtenidos hay ciertas generalidades. Se muestran algunos de los resultados obtenidos agrupados por características comunes. Por construcción la categoría Todo contiene más datos. En esta categoría el 85,2% corresponde a Ejecuciones, 10,5% a Enfrentamientos y 4,3% a Agresiones. La descripción que se hace aquí es con la categoría de Todo así comparte la mayoría de los datos con Ejecuciones.

No se hace una descripción de todas las categorías porque no cambia el comportamiento. Las gráficas se pueden clasificar distintamente para diferentes categorías.

Se tienen municipios en los cuales las gráficas contienen pocos puntos para trazar dos rectas o incluso una.

Las gráficas que se dividen en dos partes: la primera pendiente contiene pocos puntos y la segunda contiene la mayoría de los puntos. Aunque las pendientes no tienen los mismos valores, conservan la misma forma, la segunda pendiente es más inclinada que la primera.

La Figura 5.3 se muestra un caso en el cual la separación de las rectas está cerca de la mitad. Las pendientes m_1 y m_2 no son valores similares y ambos tienen un valor R^2 superior a 0.80. Este es el caso menos común, pero es la gráfica que más se asemeja al caso de guerras.

La discusión que se puede hacer con estados es similar.

Conclusiones

Se encontró que la curva se puede aproximar con leyes de potencias. En la primera parte con pocos puntos y en la segunda parte se tiene la mayoría. En comparación con las guerras de Irak, Afganistán e Irlanda del norte el número de muertos registrados por evento, en el cual son mayores, podría ser causa de la diferencia en el tipo de resultados obtenidos.

Capítulo 6

Una sola ley de potencias

En gráficas anteriores se observa que la primera parte de la curva tiene pocos puntos. Hay una gráfica en donde los valores de las pendientes son cercanos. Esto sugiere que se podría hacer la aproximación con una sola recta. Se aproxima con una recta y se compara para ver si esta aproximación mejora los resultados obtenidos.

Método

Se procesan los datos de igual manera como se hace en el Capítulo 5. El tipo de ajuste cambia. Se hace una sola regresión lineal a la distribución.

Resultados

Se obtienen distribuciones, pero aproximado con una sola recta. Los resultados se muestran en la Tabla A.4. El siguiente municipio se muestra para ver el cambio de valores debido a la anexión de los puntos correspondientes al hacer la regresión de ambas rectas en una sola. En la Tabla 6.1 se muestra la comparación de resultados obtenidos.

Tabla 6.1: Comparación de valores de R^2 para la segunda recta contra una sola. La notación es la siguiente: R_2^2 corresponde a valores de la segunda pendiente, mientras que R^2 corresponde a valores con una sola, finalmente "nulo" estados donde no haya sido posible comparar debido a que no se haya podido hacer una distribución. La comparación corresponde a Estados ordenados por cantidad de eventos.

	Primeros 12	**Primeros 26**	**Primeros 32**
$\boldsymbol{R_2^2 > R^2}$	8.3 %	15.38 %	21.875 %
$\boldsymbol{R_2^2 < R^2}$	91.6 %	80.77 %	75.0%
$\boldsymbol{R_2^2 = R^2}$	0.0 %	0.0 %	0.0 %
Nulo	0.0%	3.85%	3.125 %

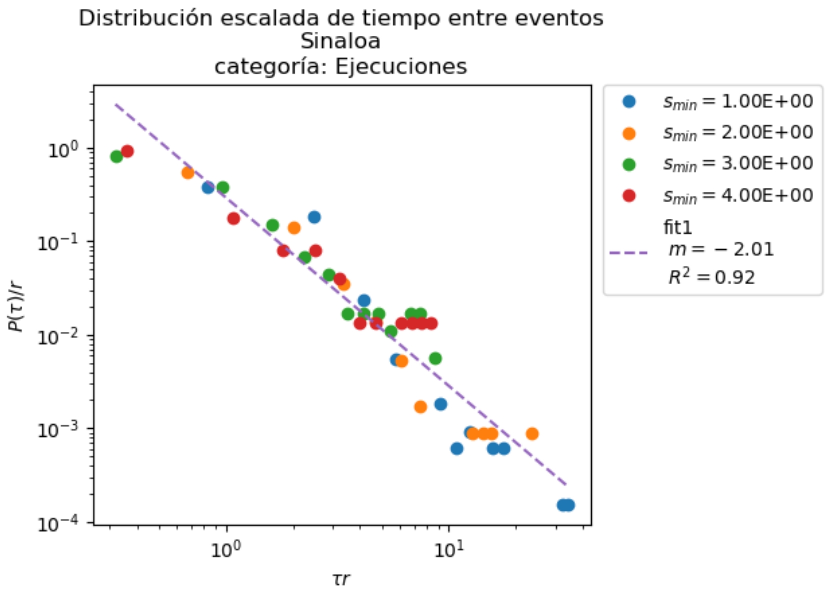


Figura 6.1: Distribución escalada de tiempo entre eventos para Sinaloa para la categoría de Ejecuciones. Aproximación con una sola recta. τ es tiempo entre evento, $P(\tau)$ la distribución de tiempo entre eventos, los colores identifican a la distribución con una cota mínima s.

Discusión

En casi todos los municipios y estados en los cuales se tienen suficientes puntos se pueden aproximar los puntos con una sola recta. Esto es porque se tiene más puntos para aproximar a una sola recta.

Los resultados de la tabla Tabla 6.1 se ve que para todos los estados el 75.0% es mejor usar una sola recta y en los doce estados más violentos es el 91.6%.

Conclusiones

Usar una sola ley de potencias en estos datos es una opción viable para describir las distribuciones re-escaladas.

Capítulo 7

Normalización de datos en municipios con población

El número de eventos ocurridos en un lugar se espera que esté relacionado con la población. Con un mayor grupo de personas es posible que se registren más eventos. Entre mayor número de eventos mayor variedad en magnitud.

Al juntar datos de distintos municipios se pierden los datos con magnitud pequeña. Los valores obtenidos son discretos en s. Para poder comparar distintas regiones se considera la magnitud en relación a su población. Esto podría dar más variedad al tipo de cortes que se pueden hacer. De esta forma para municipios con poca población la magnitud de eventos es más relevante. Es esperar que la correlación mejore.

Desarrollo

Se toma en cuenta a la población para hacer los cálculos de la siguiente manera: para un lugar un municipio sus valores s correspondientes son divididos entre la población. Esto da la perspectiva de la magnitud de s respecto a un lugar. Los valores para la población son del año 2010 tomados de la página del INEGI (Geografía (INEGI), 2016).

Para un solo municipio no se puede hacer esto debido a que la distribución está normalizada. El factor de población se pierde al normalizar y queda exactamente igual, así que no se hace. El factor de normalización es el mismo para todos los datos.

Para un estado se puede tomar un municipio y dividir sus valores de s para los eventos ocurridos entre su población. Se hace para cada uno de los municipios pertenecientes al estado. Juntando todos los datos ya no se pierde el factor de población al normalizar. Se puede hacer lo mismo a la escala de país usando los municipios.

Para seleccionar el corte de s_{min} se puede tomar múltiplos de una cantidad pequeña. Se utilizó el inverso del promedio de la población.

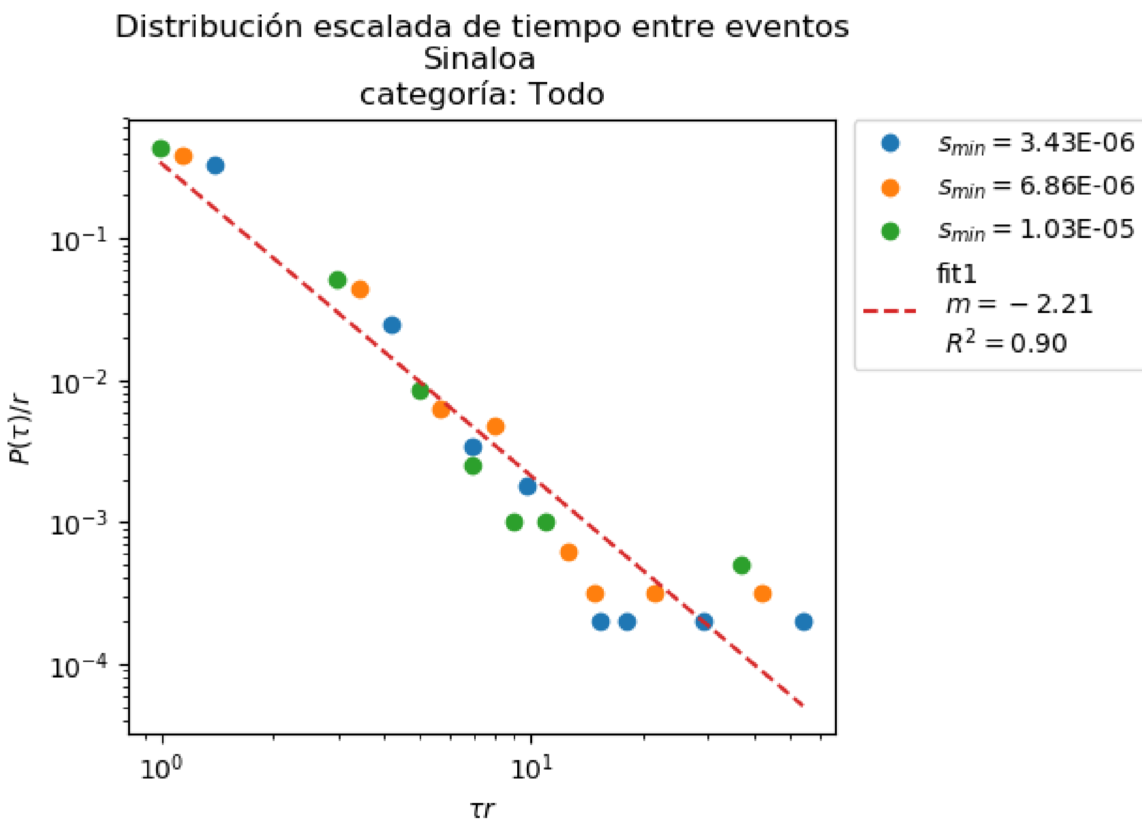


Figura 7.1: Distribución para el estado de Sinaloa, considerando la población. τ es tiempo entre evento, $P(\tau)$ la distribución de tiempo entre eventos, los colores identifican a la distribución con una cota mínima s.

En la Figura 7.1 se muestra la distribución para el estado de Sinaloa, considerando la población. En el Apéndice B se encuentran más gráficas.

Para comparar los resultados obtenidos con el capítulo anterior se comparan los valores de R^2 uno a uno. Los municipios están ordenados por número de eventos ocurridos en forma descendente. El resumen de los resultados se encuentra en la Tabla 7.1. En esta tabla se describen los resultados para estados en la categoría Todo.

Resultados para estados ordenados por número de eventos

Tabla 7.1: Se comparan los valores de R^2 correspondientes a los resultados ordenados por número de eventos para la categoría Todo por estado. La notación es la siguiente: R^2 corresponde a valores netos sin tomar en *cuenta la población, mientras que $R^2_{población}$ corresponde a valores normalizados con la población, finalmente "nulo" estados donde no haya sido posible comparar debido a que no se haya podido hacer una distribución.*

	primeros 12	primeros 26	primeros 32
$R^2 > R^2_{población}$	50.0 %	42.3 %	43.75 %
$R^2 < R^2_{población}$	50.0 %	57.7 %	56.25 %
$R^2 = R^2_{población}$	0.0 %	0.0 %	0.0 %
nulo	0.0 %	0.0 %	0.0 %

Discusión

Al comparar la gráfica en Figura 7.1 con su correspondiente en el Apéndice B hay una mejora. Considerando la Tabla 7.1 se puede hacer una afirmación más precisa. Los primeros estados cuentan con más datos. En particular, los primeros 12 están en el mismo orden de magnitud. De estos datos se puede ver que la mayoría mejora R^2 al considerar población. Con más datos sigue siendo mejor pero no en la misma proporción.

Conclusiones

Este tipo de análisis se puede hacer considerando población y el valor de R^2 mejora.

Capítulo 8

Conclusiones

Bajo la hipótesis de que los datos se pueden describir con dos leyes de potencias se describió un método para ubicar el punto en el cual se separan, las limitaciones que hay el número de cortes. El 99% de los eventos tienen 6 muertes o menos. Se encontró que método ‘doane’ es apropiado para los datos. Con esta información se determinó el tipo de distribución buscada, en el Capítulo 5. Estas distribuciones por los pocos puntos que contiene la primera recta se puede decir que la mayoría de los estados (también municipios) no cumple la hipótesis planteada. El estado de Oaxaca es el caso que más cercano a cumplir la hipótesis.

Aunque no se puedan aproximar dos curvas en los estados (también municipios) se puede aproximar al menos una ley de potencias, en los que se puede aproximar una curva. Como propuesta de mejora para esta base de datos se aproximó con una recta. Es una mejora en 75.0% de todos los estados, comparando con la segunda recta que contiene la mayoría de los puntos. En comparación al utilizar población en una sola recta mejora solo 43.75% pero es una opción para tratar los datos.

Como no se obtuvo el mismo comportamiento para guerras se puede hacer una distinción entre la guerra contra el narcotráfico y otras guerras. Una diferencia es el número de muertos que se pueden contar en un solo evento. En una guerra el número de muertos es mucho mayor. De la misma forma la guerra contra el narcotráfico y no se comporta como sismos. También difiere en las magnitudes que se pueden medir en sismos.

No es obvio porque en unas gráficas se puede ajustar una recta y en alguna otra dos. Es posible que haya un comportamiento determinante en alguno de los casos. El tema se podría seguir investigando a través de revisar intervalos de tiempo menores. El lapso disponible son seis años. Una vez determinado el tamaño del intervalo se puede recorrer para obtener una evolución temporal de las curvas. Con esto se podría observar si que se ajuste una o dos rectas es constante o no.

Referencias

Atuesta, L., Siordia, O. S., & Madrazo, A. (2016a). *La Guerra Contra las Drogas en México: Registros (oficiales) de eventos durante el periodo de diciembre 2006 a noviembre 2011, Agresiones (A-A)*. http://biiacs-dspace.cide.edu/handle/10089/17387

Atuesta, L., Siordia, O. S., & Madrazo, A. (2016b). *La Guerra Contra las Drogas en México: Registros (oficiales) de eventos durante el periodo de diciembre 2006 a noviembre 2011, Ejecuciones (A-X06 a A-X11).* http://biiacs-dspace.cide.edu/handle/10089/17389

Atuesta, L., Siordia, O. S., & Madrazo, A. (2016c). *La Guerra Contra las Drogas en México: Registros (oficiales) de eventos durante el periodo de diciembre 2006 a noviembre 2011, Enfrentamientos (A-E).* http://biiacs-dspace.cide.edu/handle/10089/17388

Bak, P., Christensen, K., Danon, L., & Scanlon, T. (2002). Unified Scaling Law for Earthquakes. *Physical Review Letters*, *88*(17). https://doi.org/10.1103/physrevlett.88.178501

Documento_Descriptivo.pdf. (s/f). Recuperado el 15 de noviembre de 2019, de http://datos.cide.edu/bitstream/handle/10089/17387/Documento_Descriptivo.pdf?sequence=11&isAllowed=y

Geografía (INEGI), I. N. de E. y. (2016). *Censo de Población y Vivienda 2010*. http://www.beta.inegi.org.mx/programas/ccpv/2010/

Las Comisiones The Lancet. (2017). La violencia en México por la guerra contra las drogas. *Ciencias*, *122–123*, 128–133.

numpy.histogram_bin_edges—NumPy v1.15 Manual. (s/f). Recuperado el 26 de enero de 2019, de https://docs.scipy.org/doc/numpy/reference/generated/numpy.histogram_bin_edges.html#numpy.histogram_bin_edges

Picoli, S., Castillo-Mussot, M. del, Ribeiro, H. V., Lenzi, E. K., & Mendes, R. S. (2014). Universal bursty behaviour in human violent conflicts. *Scientific Reports*, *4*(1). https://doi.org/10.1038/srep04773

Picoli, Sergio, Antonio, F. J., Itami, A. S., & Mendes, R. S. (2017). Power-law relaxation in human violent conflicts. *The European Physical Journal B*, *90*(8). https://doi.org/10.1140/epjb/e2017-80127-3

Apéndice A

Tablas

Tabla A.1: Métodos disponible en numpy para determinar el número de bins.

'auto'	Maximum of the 'sturges' and 'fd' estimators. Provides good all around performance.
'fd' (Freedman Diaconis Estimator)	Robust (resilient to outliers) estimator that takes into account data variability and data size.
'doane'	An improved version of Sturges' estimator that works better with non-normal datasets.
'scott'	Less robust estimator that that takes into account data variability and data size.
'stone'	Estimator based on leave-one-out cross-validation estimate of the integrated squared error. Can be regarded as a generalization of Scott's rule.
'rice'	Estimator does not take variability into account, only data size. Commonly overestimates number of bins required.
'sturges'	R's default method, only accounts for data size. Only optimal for gaussian data and underestimates number of bins for large non-gaussian datasets.
'sqrt'	Square root (of data size) estimator, used by Excel and other programs for its speed and simplicity.

Tabla A.2: Información estadística de estados de México.

lugar	no. eventos	promedio	std	min	25 %	50 %	75 %	max	total
Chihuahua	8884	1.45	1.01	1	1	1	2	20	12843
Sinaloa	4133	1.38	1.04	1	1	1	1	28	5711
Guerrero	3087	1.47	1.56	1	1	1	1	55	4545
Tamaulipas	1183	2.35	3.60	0	1	1	3	72	2782
Durango	1517	1.77	2.73	1	1	1	2	56	2684
Michoacán de Ocampo	1701	1.41	1.03	1	1	1	1	12	2403
Nuevo León	1342	1.76	2.40	1	1	1	2	52	2361
Baja California	1708	1.36	0.98	1	1	1	1	18	2331
México	1669	1.34	0.95	1	1	1	1	24	2233
Jalisco	1355	1.36	1.17	1	1	1	1	26	1839
Sonora	1056	1.47	1.47	1	1	1	2	23	1552
Coahuila de Zaragoza	819	1.66	1.44	1	1	1	2	19	1363
Veracruz de Ignacio de la Llave	692	1.75	2.04	1	1	1	2	35	1208
Nayarit	516	1.65	1.85	1	1	1	2	29	851
Ciudad de México	682	1.22	0.58	1	1	1	1	6	829
Morelos	593	1.33	0.78	1	1	1	1	7	786
Guanajuato	534	1.32	0.98	1	1	1	1	15	703
Oaxaca	439	1.34	0.97	1	1	1	1	13	587
Chiapas	294	1.31	0.84	1	1	1	1	8	386

lugar	no. eventos	promedio	std	min	25 %	50 %	75 %	max	total
San Luis Potosí	196	1.74	1.26	1	1	1	2	9	341
Tabasco	228	1.31	1.08	1	1	1	1	12	298
Zacatecas	113	2.53	2.36	1	1	2	3	15	286
Colima	197	1.30	0.87	1	1	1	1	8	257
Quintana Roo	153	1.42	1.27	1	1	1	1	12	218
Hidalgo	148	1.41	1.80	1	1	1	1	19	209
Aguascalientes	148	1.32	0.70	1	1	1	1	5	196
Puebla	145	1.23	0.55	1	1	1	1	4	178
Querétaro	36	1.53	1.30	1	1	1	2	8	55
Campeche	26	1.58	1.21	1	1	1	1.75	6	41
Baja California Sur	27	1.22	0.58	1	1	1	1	3	33
Yucatán	16	1.75	2.52	1	1	1	1	11	28
Tlaxcala	17	1.24	0.44	1	1	1	1	2	21

Tabla A.3: Resultados para todas las categorías a escala de municipios.

categoría	escala	edo	mpio	m_1	R_1^2	m_2	R_2^2
Agresiones	country	0	0	-0.541	0.851	-1.964	0.865
Todo	country	0	0	-0.624	0.882	-1.968	0.737
Ejecuciones	country	0	0	-0.327	0.718	-2.032	0.742
Agresiones	edo	8		-0.682	0.932	-1.077	0.581
Agresiones	edo	19		-1.108	0.704	-1.348	0.516
Agresiones	edo	12		-1.214	0.84	-1.884	0.991
Agresiones	edo	25		-0.544	0.414	-2.561	0.66
Agresiones	edo	16		-0.942	0.689	-1.799	1
Agresiones	edo	28		-1.542	0.78	-1.447	0.709
Agresiones	edo	14		-0.595	0.268	-3.946	0.778

categoría	escala	edo	mpio	m_1	R_1^2	m_2	R_2^2
Agresiones	edo	5		-1.149	0.319	-2.98	0.918
Agresiones	edo	2		-0.179	0.113	1.357	1
Agresiones	edo	10		-1.219	0.919	-1.728	0.548
Agresiones	edo	26		10.537	1	-0.686	0.257
Agresiones	edo	15		-1.268	0.53	-1.255	0.317
Todo	edo	8		-0.571	1	-2.294	0.904
Todo	edo	25		-0.831	0.997	-2.499	0.943
Todo	edo	12		-0.955	0.737	-2.258	0.853
Todo	edo	28		-0.758	0.968	-1.345	0.671
Todo	edo	10		-1.333	0.932	-2.066	0.817
Todo	edo	16		-0.827	0.763	-2.24	0.85
Todo	edo	19		-1.154	0.992	-1.685	0.769
Todo	edo	2		-1.333	0.983	-2.549	0.841
Todo	edo	15		-1.038	0.694	-2.233	0.841
Todo	edo	14		-0.543	0.828	-1.996	0.787
Todo	edo	26		-1.204	0.945	-2.032	0.774
Todo	edo	5		-0.68	0.986	-1.774	0.819
Enfrentamientos	edo	28		-0.816	0.996	-0.906	0.483
Enfrentamientos	edo	19		-0.615	0.983	-1.2	0.681
Enfrentamientos	edo	12		-0.693	1	-1.318	0.726
Enfrentamientos	edo	25		-1.449	0.995	-1.227	0.799
Enfrentamientos	edo	8		-0.642	0.871	-1.789	0.647
Enfrentamientos	edo	10		-0.553	0.982	-1.543	0.802
Enfrentamientos	edo	16		-0.977	0.989	-1.035	0.53
Enfrentamientos	edo	30		-0.695	0.954	-0.553	0.264
Enfrentamientos	edo	5		-0.879	0.909	-1.035	0.475
Enfrentamientos	edo	18		-1.499	0.959	-0.586	0.423
Enfrentamientos	edo	26		-1.095	0.775	-4.14	0.658
Enfrentamientos	edo	14		-1.448	0.852	-3.275	0.56
Ejecuciones	edo	8		-0.47	0.855	-2.229	0.888
Ejecuciones	edo	25		-0.705	0.947	-2.398	0.923
Ejecuciones	edo	12		-1.029	0.753	-2.157	0.811
Ejecuciones	edo	10		-0.884	0.974	-2.033	0.885
Ejecuciones	edo	2		-1.309	0.955	-2.374	0.812
Ejecuciones	edo	15		-0.683	0.999	-2.127	0.789
Ejecuciones	edo	16		-0.725	0.772	-2.263	0.82
Ejecuciones	edo	14		-1.312	0.995	-1.925	0.82
Ejecuciones	edo	19		-1.731	0.751	-1.591	0.815
Ejecuciones	edo	28		-1.805	0.934	-1.41	0.754
Ejecuciones	edo	26		-1.085	0.975	-1.963	0.821
Ejecuciones	edo	5		-0.825	1	-1.716	0.87
Agresiones	mpio	8	37	-0.935	0.901	-1.147	0.455

categoría	escala	edo	mpio	m_1	R_1^2	m_2	R_2^2
Agresiones	mpio	19	39	21.919	1	-2.086	0.181
Agresiones	mpio	12	1	-1.394	0.867	-1.894	0.987
Agresiones	mpio	8	19	-1.35	0.604	-4.411	0.328
Agresiones	mpio	5	35	-0.819	1	-1.575	0.196
Agresiones	mpio	2	4	-0.263	0.491	-7.385	0.599
Agresiones	mpio	25	6	-0.953	0.93	-6.409	0.252
Agresiones	mpio	16	112	12.457	1		0
Agresiones	mpio	25	18	2.275	0.426	-5.755	0.215
Agresiones	mpio	19	46	-1.436	0.78	-1.698	1
Agresiones	mpio	25	12	-1.836	1	-5.865	1
Agresiones	mpio	10	12	-1	1	2.111	1
Todo	mpio	8	37	-0.124	0.091	-2.127	0.887
Todo	mpio	25	6	-1.048	0.876	-2.318	0.873
Todo	mpio	8	19	-0.446	0.925	-2.09	0.89
Todo	mpio	2	4	-1.241	0.955	-2.19	0.826
Todo	mpio	12	1	-0.554	0.649	-1.794	0.751
Todo	mpio	5	35	-0.665	0.987	-1.722	0.78
Todo	mpio	19	39	-1.019	0.618	-1.122	0.485
Todo	mpio	10	5	-1.551	0.872	-1.523	0.73
Todo	mpio	25	12	-2.296	1	-1.642	0.784
Todo	mpio	10	7	-0.733	0.884	-1.196	0.569
Todo	mpio	25	18	-0.724	0.927	-1.945	0.808
Todo	mpio	26	43	-0.702	0.989	-1.751	0.789
Enfrentamientos	mpio	28	27	-1.308	0.836	-0.661	0.272
Enfrentamientos	mpio	28	32	-1.7	0.905	0.358	0.062
Enfrentamientos	mpio	28	22	-1.33	0.655	-0.489	0.29
Enfrentamientos	mpio	5	35	-1.247	0.831	-1.784	0.865
Enfrentamientos	mpio	18	17	-0.896	0.829	-0.214	0.145
Enfrentamientos	mpio	8	37	-0.718	0.64	3.119	0.886
Enfrentamientos	mpio	19	39	0.284	0.173	-1.19	0.565
Enfrentamientos	mpio	28	24	-1.08	0.675	-1.126	0.932
Enfrentamientos	mpio	28	40	-0.776	0.563	-1.062	0.527
Enfrentamientos	mpio	2	4	-0.601	0.331	-0.128	0.008
Enfrentamientos	mpio	25	12	-0.405	0.821	-2.171	0.636
Enfrentamientos	mpio	10	5	-0.444	0.181	-1.56	0.649
Ejecuciones	mpio	8	37	-0.583	0.86	-2.142	0.893
Ejecuciones	mpio	25	6	-0.929	0.846	-2.481	0.868
Ejecuciones	mpio	8	19	-0.623	1	-2.039	0.895
Ejecuciones	mpio	2	4	-1.195	0.843	-1.863	0.819
Ejecuciones	mpio	12	1	-0.62	0.998	-1.764	0.814
Ejecuciones	mpio	5	35	-1.474	0.784	-1.577	0.805
Ejecuciones	mpio	10	5	-1.979	0.88	-1.503	0.658

categoría	escala	edo	mpio	m_1	R_1^2	m_2	R_2^2
Ejecuciones	mpio	25	12	-0.86	0.999	-1.367	0.631
Ejecuciones	mpio	10	7	-0.505	0.991	-1.323	0.656
Ejecuciones	mpio	19	39	-0.71	0.981	-1.247	0.576
Ejecuciones	mpio	25	18	-0.811	0.881	-1.965	0.787
Ejecuciones	mpio	26	43	-1.325	0.851	-1.78	0.747

Tabla A.4: Distribuciones escaladas con aproximación de una sola recta.

categoría	escala	edo	mpio	m_1	R_1^2
Agresiones	country	0	0	-1.703	0.89
Todo	country	0	0	-2.146	0.882
Enfrentamientos	country	0	0	-2.002	0.851
Ejecuciones	country	0	0	-2.117	0.875
Agresiones	edo	8		-1.265	0.781
Agresiones	edo	19		-1.333	0.745
Agresiones	edo	12		-0.994	0.838
Agresiones	edo	25		-1.051	0.503
Agresiones	edo	16		-0.916	0.734
Agresiones	edo	28		-1.066	0.723
Agresiones	edo	14		-0.435	0.092
Agresiones	edo	5		-1.066	0.593
Agresiones	edo	2		-0.421	0.331
Agresiones	edo	10		-0.945	0.723
Agresiones	edo	26		-0.833	0.648
Agresiones	edo	15		-0.974	0.544
Todo	edo	8		-2.202	0.91
Todo	edo	25		-2.19	0.935
Todo	edo	12		-2.117	0.905
Todo	edo	28		-1.702	0.826
Todo	edo	10		-1.972	0.929
Todo	edo	16		-1.78	0.894
Todo	edo	19		-1.909	0.889
Todo	edo	2		-1.938	0.844
Todo	edo	15		-1.978	0.867
Todo	edo	14		-1.799	0.863
Todo	edo	26		-1.857	0.891
Todo	edo	5		-1.805	0.898
Enfrentamientos	edo	28		-1.517	0.753
Enfrentamientos	edo	19		-1.427	0.827

categoría	escala	edo	mpio	m_1	R_1^2
Enfrentamientos	edo	12		-1.509	0.877
Enfrentamientos	edo	25		-1.274	0.924
Enfrentamientos	edo	8		-1.277	0.756
Enfrentamientos	edo	10		-1.231	0.861
Enfrentamientos	edo	16		-1.3	0.831
Enfrentamientos	edo	30		-1.178	0.704
Enfrentamientos	edo	5		-1.33	0.788
Enfrentamientos	edo	18		-1.207	0.9
Enfrentamientos	edo	26		-1.094	0.781
Enfrentamientos	edo	14		-1.217	0.785
Ejecuciones	edo	8		-2.179	0.919
Ejecuciones	edo	25		-2.111	0.931
Ejecuciones	edo	12		-2.002	0.877
Ejecuciones	edo	10		-1.936	0.932
Ejecuciones	edo	2		-1.955	0.858
Ejecuciones	edo	15		-2.056	0.867
Ejecuciones	edo	16		-1.839	0.851
Ejecuciones	edo	14		-1.855	0.905
Ejecuciones	edo	19		-1.769	0.911
Ejecuciones	edo	28		-1.571	0.875
Ejecuciones	edo	26		-1.801	0.9
Ejecuciones	edo	5		-1.721	0.927

Tabla A.5: Valores de pendiente con una sola para población..

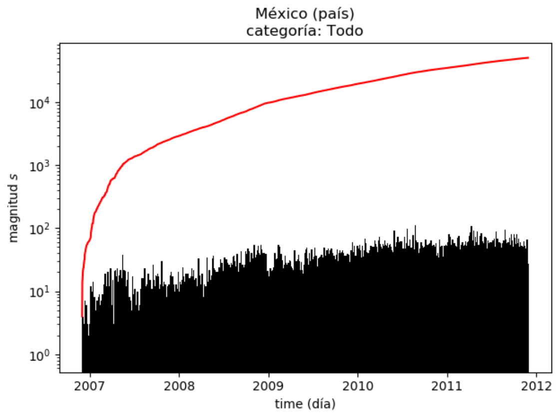
categoría	escala	edo	mpio	m_1	R_1^2
Agresiones	country	0	0	-1.601	0.887
Todo	country	0	0	-2.317	0.923
Enfrentamientos	country	0	0	-2.08	0.874
Ejecuciones	country	0	0	-2.194	0.922
Agresiones	edo	8		-0.837	0.974
Agresiones	edo	19		-0.574	0.54
Agresiones	edo	12		-0.696	0.58
Agresiones	edo	25		-0.749	0.735
Agresiones	edo	16		-0.795	0.772
Agresiones	edo	28		-0.854	0.72
Agresiones	edo	14		-0.339	0.288
Agresiones	edo	2		-0.291	0.37
Agresiones	edo	10		-0.093	0.001
Agresiones	edo	26		-0.64	0.739
Agresiones	edo	15		0.244	0.008

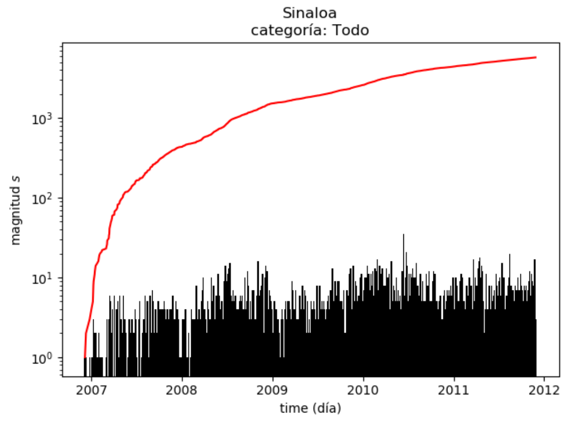
categoría	escala	edo	mpio	m_1	R_1^2
Agresiones	edo	32		1.292	0.127
Todo	edo	8		-2.011	0.903
Todo	edo	25		-2.206	0.904
Todo	edo	12		-2.142	0.936
Todo	edo	28		-1.647	0.793
Todo	edo	10		-1.76	0.912
Todo	edo	16		-1.949	0.895
Todo	edo	19		-1.753	0.912
Todo	edo	2		-1.771	0.914
Todo	edo	15		-1.901	0.891
Todo	edo	14		-1.716	0.913
Todo	edo	26		-1.545	0.884
Todo	edo	5		-1.156	0.778
Enfrentamientos	edo	28		-1.374	0.71
Enfrentamientos	edo	19		-1.363	0.847
Enfrentamientos	edo	12		-1.434	0.86
Enfrentamientos	edo	25		-1.305	0.89
Enfrentamientos	edo	8		-1.125	0.863
Enfrentamientos	edo	10		-1.043	0.865
Enfrentamientos	edo	16		-1.34	0.843
Enfrentamientos	edo	30		-1.057	0.79
Enfrentamientos	edo	5		-0.774	0.732
Enfrentamientos	edo	18		-1.06	0.803
Enfrentamientos	edo	26		-0.931	0.806
Enfrentamientos	edo	14		-0.887	0.82
Ejecuciones	edo	8		-2.016	0.89
Ejecuciones	edo	25		-2.07	0.919
Ejecuciones	edo	12		-2.043	0.934
Ejecuciones	edo	10		-1.743	0.934
Ejecuciones	edo	2		-1.784	0.903
Ejecuciones	edo	15		-1.87	0.895
Ejecuciones	edo	16		-1.825	0.892
Ejecuciones	edo	14		-1.809	0.922
Ejecuciones	edo	19		-1.604	0.909
Ejecuciones	edo	28		-1.513	0.785
Ejecuciones	edo	26		-1.563	0.889
Ejecuciones	edo	5		-1.07	0.804

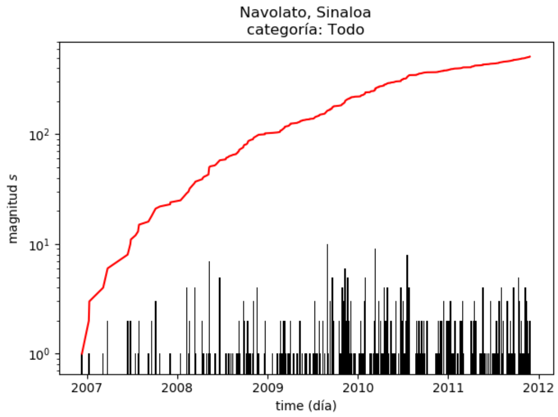
Apéndice B

Gráficas

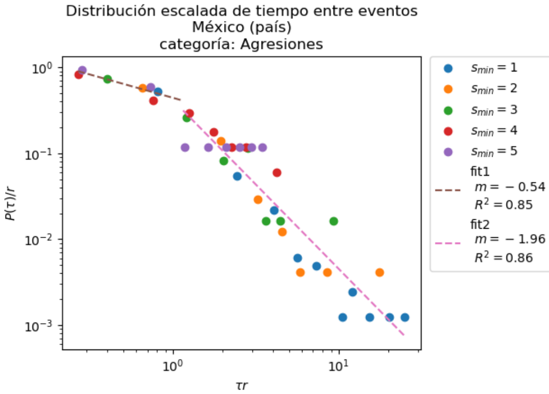
Muertos por día

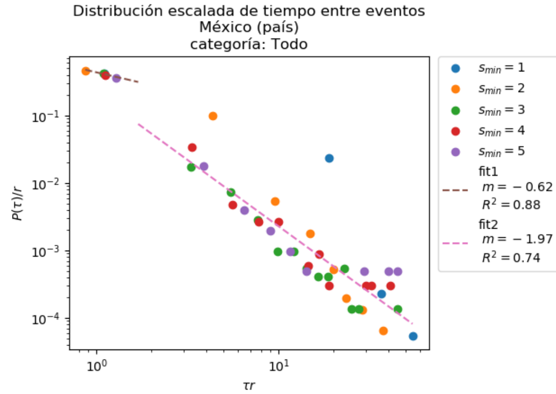


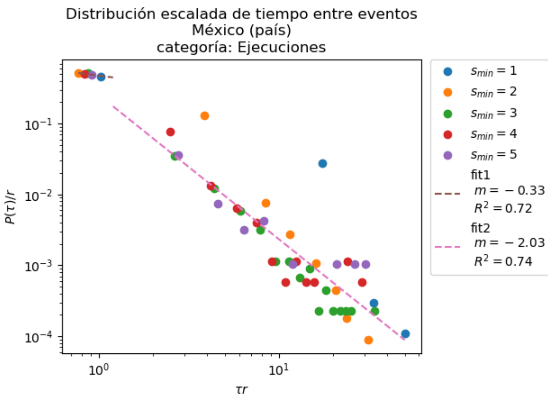


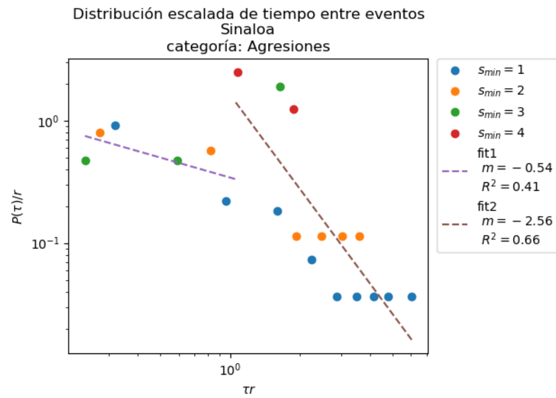


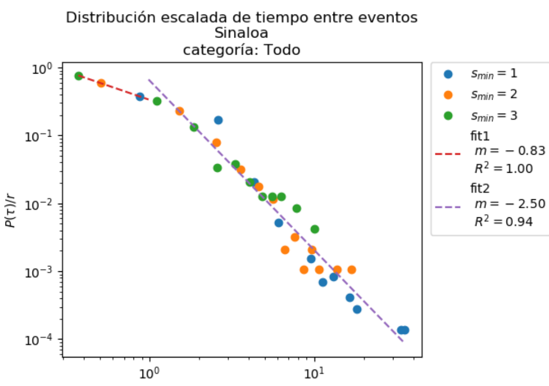
Distribución de tiempo entre eventos

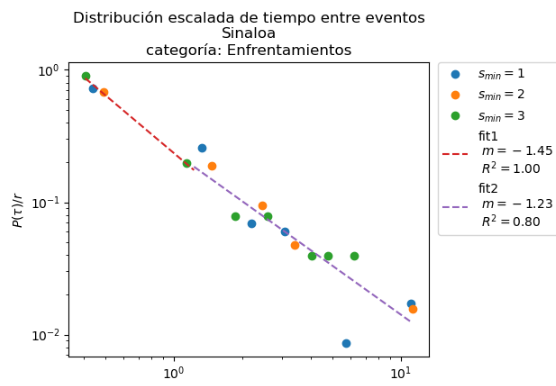


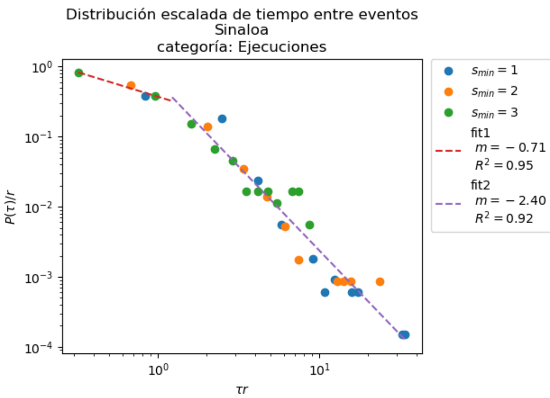


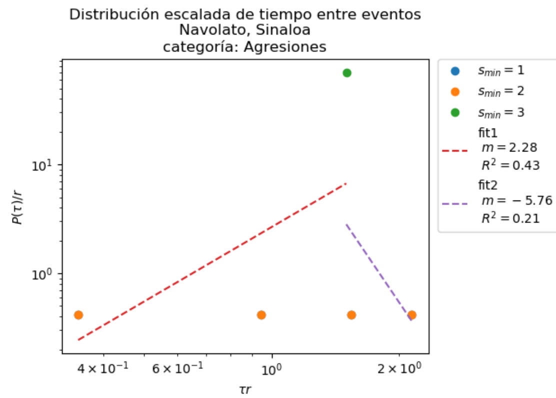


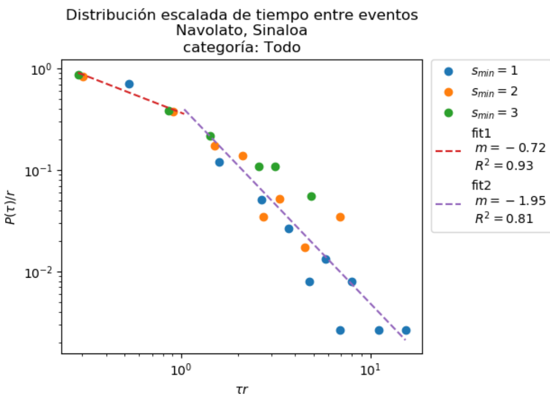


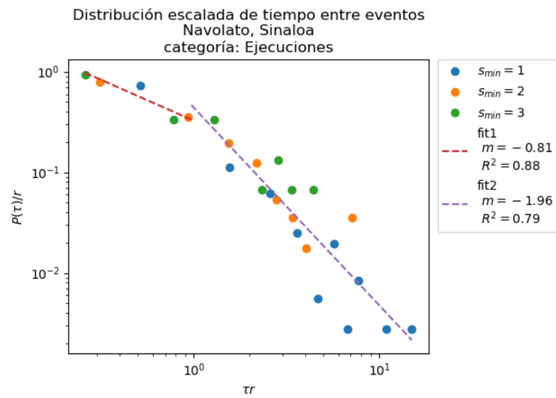




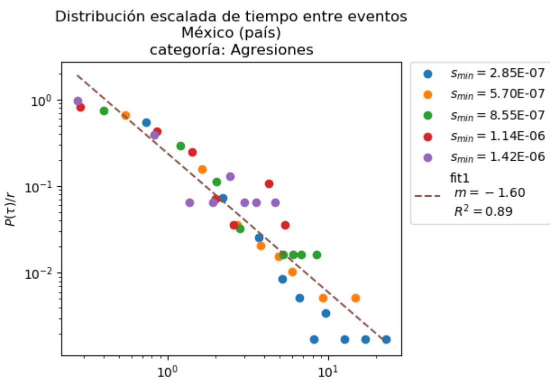

Distribución escalada de tiempo entre eventos
Sinaloa
categoría: Ejecuciones
$s_{min} = 1$
$s_{min} = 2$
$s_{min} = 3$
fit1
$m = -0.71$
$R^2 = 0.95$
fit2
$m = -2.40$
$R^2 = 0.92$
$P(\tau)/r$
τr

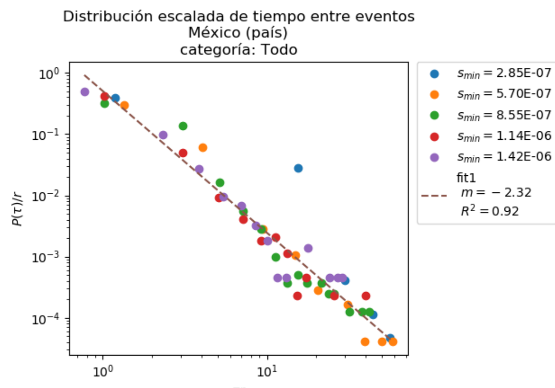

Distribución escalada de tiempo entre eventos
Navolato, Sinaloa
categoría: Agresiones
$s_{min} = 1$
$s_{min} = 2$
$s_{min} = 3$
fit1
$m = 2.28$
$R^2 = 0.43$
fit2
$m = -5.76$
$R^2 = 0.21$
$P(\tau)/r$
τr

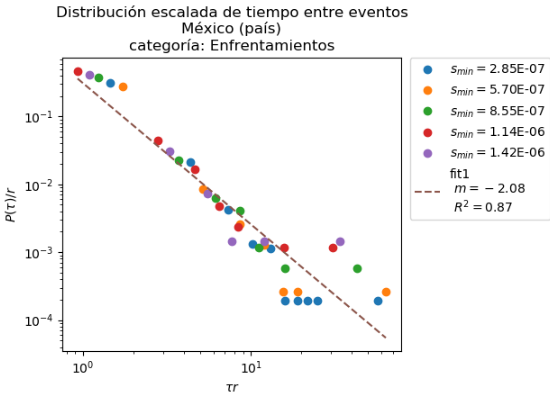

Distribución escalada de tiempo entre eventos
Navolato, Sinaloa
categoría: Todo
$s_{min} = 1$
$s_{min} = 2$
$s_{min} = 3$
fit1
$m = -0.72$
$R^2 = 0.93$
fit2
$m = -1.95$
$R^2 = 0.81$
$P(\tau)/r$
τr

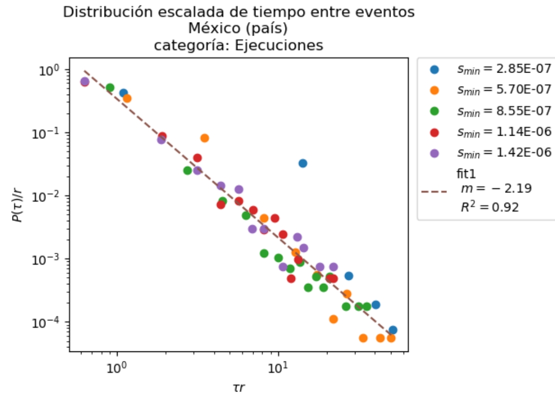

Distribución escalada de tiempo entre eventos
Navolato, Sinaloa
categoría: Ejecuciones
$s_{min} = 1$
$s_{min} = 2$
$s_{min} = 3$
fit1
$m = -0.81$
$R^2 = 0.88$
fit2
$m = -1.96$
$R^2 = 0.79$
$P(\tau)/r$
τr

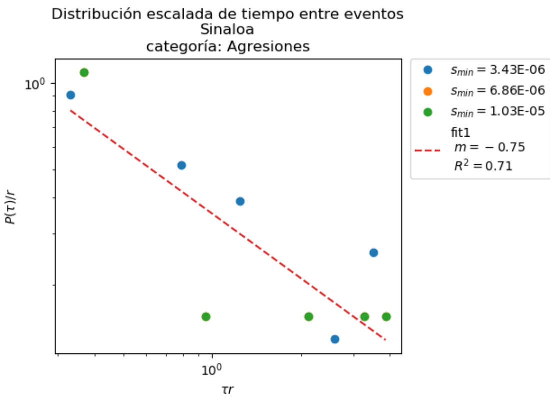
Población

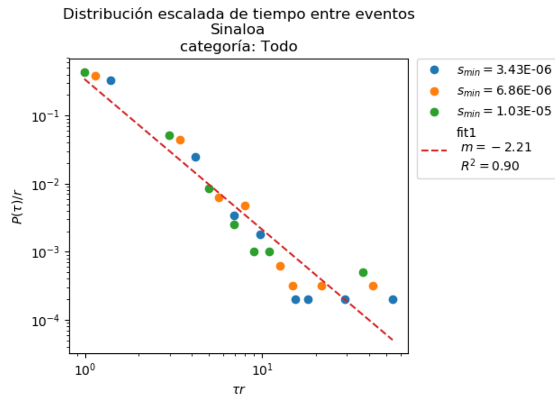

Distribución escalada de tiempo entre eventos
México (país)
categoría: Agresiones
$s_{min} = 2.85E-07$
$s_{min} = 5.70E-07$
$s_{min} = 8.55E-07$
$s_{min} = 1.14E-06$
$s_{min} = 1.42E-06$
fit1
$m = -1.60$
$R^2 = 0.89$
$P(\tau)/r$
τr

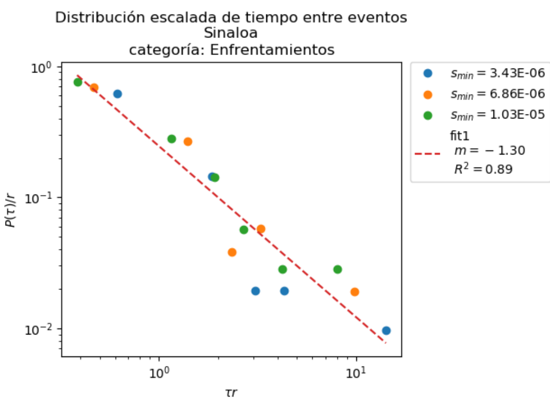

Distribución escalada de tiempo entre eventos
México (país)
categoría: Todo
$s_{min} = 2.85E-07$
$s_{min} = 5.70E-07$
$s_{min} = 8.55E-07$
$s_{min} = 1.14E-06$
$s_{min} = 1.42E-06$
fit1
$m = -2.32$
$R^2 = 0.92$
$P(\tau)/r$
τr

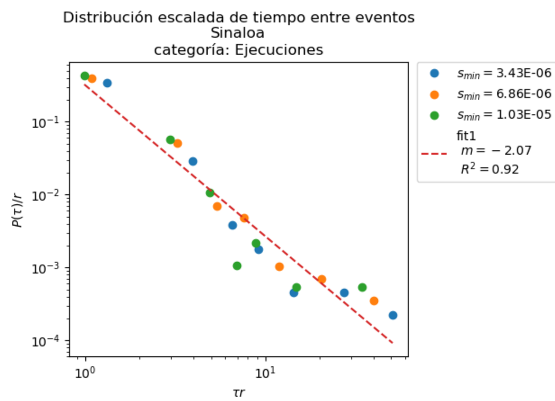

Distribución escalada de tiempo entre eventos
México (país)
categoría: Enfrentamientos
$P(\tau)/r$
τr
$s_{min} = 2.85E-07$
$s_{min} = 5.70E-07$
$s_{min} = 8.55E-07$
$s_{min} = 1.14E-06$
$s_{min} = 1.42E-06$
fit1
$m = -2.08$
$R^2 = 0.87$


Distribución escalada de tiempo entre eventos
México (país)
categoría: Ejecuciones
$P(\tau)/r$
τr
$s_{min} = 2.85E-07$
$s_{min} = 5.70E-07$
$s_{min} = 8.55E-07$
$s_{min} = 1.14E-06$
$s_{min} = 1.42E-06$
fit1
$m = -2.19$
$R^2 = 0.92$


Distribución escalada de tiempo entre eventos
Sinaloa
categoría: Agresiones
$P(\tau)/r$
τr
$s_{min} = 3.43E-06$
$s_{min} = 6.86E-06$
$s_{min} = 1.03E-05$
fit1
$m = -0.75$
$R^2 = 0.71$


Distribución escalada de tiempo entre eventos
Sinaloa
categoría: Todo
$P(\tau)/r$
τr
$s_{min} = 3.43E-06$
$s_{min} = 6.86E-06$
$s_{min} = 1.03E-05$
fit1
$m = -2.21$
$R^2 = 0.90$


Distribución escalada de tiempo entre eventos
Sinaloa
categoría: Enfrentamientos
$P(\tau)/r$
τr
$s_{min} = 3.43E-06$
$s_{min} = 6.86E-06$
$s_{min} = 1.03E-05$
fit1
$m = -1.30$
$R^2 = 0.89$


Distribución escalada de tiempo entre eventos
Sinaloa
categoría: Ejecuciones
$P(\tau)/r$
τr
$s_{min} = 3.43E-06$
$s_{min} = 6.86E-06$
$s_{min} = 1.03E-05$
fit1
$m = -2.07$
$R^2 = 0.92$

Apéndice C

Código

Referencia

Este archivo indica la ubicación donde se aloja el código. Antes de ejecutar cualquier archivo este se debe ajustar la carpeta donde se quiera ejecutar.

reference.py

```
"""This file conatins the reference_folder"""
import os

reference_folder = r"C:\Users\Arturo\Desktop\thesis\Network"  # reference folder
```

Archivos de ejecución previa a gráficas

Estos archivos necesitan ser ejecutados antes de las gráficas porque preparan archivos que serán utilizados en el programa.

sf01_SortEd.py

```
""" This make a list sorted of all states by pf.

pf is the total number of deaths.
This returns the keys in order without names."""

import pandas as pd
import names
import routine as rt
import numpy as np
from os import chdir

chdir(rt.reference_folder)
chdir("Data")

#  statistic information of each category
scale = ["edo", "mpio", "country"]
category = ["Ejecuciones", "Agresiones", "Enfrentamientos", "Todo"]

for scl in scale:
    if scl is "country":
        for cat in category:
            df = []
            # list of parameters
            kedo, kmpio, vnt, pf = [], [], [], []
```

```
            kedo.append(0)
            kmpio.append(0)
            tm, dt = rt.import_data(cat, 0, 0)
            vnt.append(len(tm))     # number of events
            pf.append(np.sum(dt))   # number of deaths
            df = pd.DataFrame()

            df["edo"] = kedo
            df["mpio"] = kmpio
            df["count"] = vnt
            df["pf"] = pf

            df = df.sort_values(by="pf", ascending=False)
            df.to_csv("TableSorted-{}-{}.csv".format(scl, cat), index=False)
        continue

    for cat in category:
        df = []
        # list parameters
        kedo, kmpio, vnt, pf = [], [], [], []
        for edo in range(1, 33):
            OneLoop = False
            for mpio in names.rangeEdo(edo):
                if OneLoop:
                    continue
                elif scl is "edo":
                    mpio = None
                    OneLoop = True
                # parameters
                kedo.append(edo)
                kmpio.append(mpio)
                tm, dt = rt.import_data(cat, edo, mpio)
                vnt.append(len(tm))     # evnts
                pf.append(np.sum(dt))
        df = pd.DataFrame()
        df["edo"] = kedo
        df["mpio"] = kmpio
        df["count"] = vnt
        df["pf"] = pf
        df = df.sort_values(by="pf", ascending=False)
        # saving table
        df.to_csv("TableSorted-{}-{}.csv".format(scl, cat), index=False)
```

sf02_parameters_calculation.py

```
"""This is a file optimize the parameters and returns table of R^2's a file.

Generate a table with the results."""
import routine as rt
import pandas as pd
import names
import os
from Hmodule import *
```

```
category = ["Agresiones", "Enfrentamientos", "Ejecuciones", "Todo"]
scale = ["country", "edo", "mpio"]
hypothesis = ["0", "1"]
criteria = ["head", "tail", "sum"]

# scale = ["mpio"]
# hypothesis = ["0"]
# criteria = ["sum"]

# table
df = pd.DataFrame(columns=["category", "uniform", "criteria", "scale",
"hypothesis", "edo", "mpio", "sep",
                           r"m_1", r"R_1^2",
                           r"m_2", r"R_2^2", "R_1^2+R_2^2", "bins", "s"])


for hyp in hypothesis:
    for cri in criteria:
        for scl in scale:
            for cat in category:
                # directory
                cwd = os.getcwd()
                os.chdir(rt.reference_folder)
                os.chdir("Data")

                # list of keys
                keys = rt.keys(cat, scl)
                if scl == "mpio": keys = keys[:500]
                if scl == "country": pass

                os.chdir(cwd)

                for edo, mpio in zip(keys.edo, keys.mpio):
                    # hypothesis
                    tm, pf = rt.import_data(cat, edo, mpio)
                    if hyp == "0":
                        tm, pf = rt.same_event_0(tm, pf)
                    elif hyp == "1":
                        tm, pf = rt.same_event_1(tm, pf)
                    elif hyp == "2":
                        tm, pf = rt.same_event_2(tm, pf)

                    # finding maximum
                    sep, b, s = selecting_parameters(tm, pf, option=cri)

                    # adding results
                    if scl is "country": s = range(1, 6)
                    xdata, ydata = InterEvent(tm, pf, bins="doane", s=s,
scaled=True, uniform=False)

                    try:
                        R2_1, R2_2 = Regression(xdata, ydata, sep, option="")
                        m_1, m_2 = regression_m(xdata, ydata, sep, option="")
                    except ValueError:
```

```
                        R2_1, R2_2 = 0, 0
                        continue

                    df = df.append({"category": cat, "uniform": False,
"criteria": cri, "scale": scl, "hypothesis": hyp,
                                    "edo": edo, "mpio": mpio, "sep": round(sep,
3),
                                    r"m_1": round(m_1, 3), r"R_1^2": round(R2_1,
3),
                                    r"m_2": round(m_2, 3), r"R_2^2": round(R2_2,
3),
                                    "R_1^2+R_2^2": round(R2_1 + R2_2, 3),
"bins": b, "s":s},
                                   ignore_index=True)


os.chdir(rt.reference_folder)
os.chdir("Data")
df.to_csv("results.csv", encoding='utf-8', index=False)
```

sf_taza.py

```
"""This file makes a reduced file from iter_00_cpv2010.csv

Is not needed the population data form a locality.
This file separtes muniicipalities by state.
This make file for names.
"""
import os
import routine as rt
import pandas as pd


os.chdir(rt.reference_folder)
os.chdir("Data")

df = pd.read_csv("iter_00_cpv2010.csv")      # population data source
df = df.loc[:, ["entidad", "nom_ent", "mun", "nom_mun", "loc", "pobtot"]]
df = df[df.loc[:, "loc"] == 0]

# make file for states names
for i in range(33):
    df_i = df[df.loc[:, "entidad"] == i]
    df_i.to_csv("entidad_{}.csv".format(i), index=False)


# # file for municipalities names
# for i in range(1, 33):
#     df_i = df[df.loc[:, "entidad"] == i]
#     df_i.to_csv("entidad_{}.csv".format(i))
#
# # file for state population
# for i in range(1, 33):
```

```
#     df_i = df[df.loc[:, "entidad"] == i]
#     df_i.to_csv("entidad_{}.csv".format(i))
#
# # file for municipalities population
# for i in range(1, 33):
#     df_i = df[df.loc[:, "entidad"] == i]
#     df_i.to_csv("entidad_{}.csv".format(i))
```

Módulos

Esto son los módulos utilizados para el código.

cumulative.py

```
"""Plot for the cumulative distribution and Regression

Integral of distribution using the sample. using the simpsom method
from scipy.integrate import simps
"""

import matplotlib.pyplot as plt
import numpy as np
import Hmodule as hm
from scipy import stats
from scipy import integrate
from scipy.optimize import curve_fit


def F(x, m, b):
    return x ** (m + 1) * np.exp(b) / (m + 1)


def cumulative_InterEvent(tm, pf, bins, sequence):
    """This function computes the cumulative curve for a set of data tm, pf."""
    # distribution
    x, y = hm.InterEvent(tm, pf, bins=bins, s=sequence, scaled=True,
uniform=False)

    m = [(x[i], y[i]) for i in range(len(x))]
    m.sort()

    x, y = [row[0] for row in m], [row[1] for row in m]
    x, y = np.array(x), np.array(y)

    A = sum(y)
    y = 1 - np.cumsum(y) / A
    y = y/A

    data = [(x[i], y[i]) for i in range(len(x))]
    data = [row for row in data if row[0] > 0 and row[1] > 0]
    return [row[0] for row in data], [row[1] for row in data]
```

```
def integral_InterEvent(tm, pf, bins, sequence):
    """This function computes the cumulative curve for a set of data tm, pf."""
    # distribution
    x, y = hm.InterEvent(tm, pf, bins=bins, s=sequence, scaled=True,
uniform=False)

    m = [(x[i], y[i]) for i in range(len(x))]
    m.sort()

    x, y = [row[0] for row in m], [row[1] for row in m]
    x, y = np.array(x), np.array(y)

    A = integrate.simps(y, x)
    y = [integrate.simps(y[:i], x[:i]) for i in range(2, x.size)]
    x = np.array(x[2:])

    y = 1 - np.array(y) / A

    data = [(x[i], y[i]) for i in range(len(x))]
    data = [row for row in data if row[0] > 0 and row[1] > 0]
    return [row[0] for row in data], [row[1] for row in data]


# def plot_linear_regression(xdata, ydata, interval, scaled=True):
#     data = [(xdata[i], ydata[i]) for i in range(len(xdata))]
#     data = [row for row in data if row[0] > 0 and row[1] > 0]
#     # linear regression
#     for i in range(len(interval) - 1):
#         if not scaled: break
#         # selecting data for the interval
#         M = [row for row in data if interval[i] <= row[0] <= interval[i + 1]]
# (n, 2)
#         # checking there is data
#         if len(M) < 2: continue
#         # linear regression
#         xdata, ydata = [[row[i] for row in M] for i in range(len(M[0]))]  #
(2, n)
#         slope, intercept, r_value, p_value, std_err =
stats.linregress(np.log(xdata), np.log(ydata))
#         # plot fit
#         x = np.linspace(interval[i], interval[i + 1])
#         plt.loglog(x, np.exp((lambda x, m, b: m * x + b)(np.log(x), slope,
intercept)), '--',
#                    label=r"fit{}".format(i + 1) + "\n" + \
#                          r" $m ={:.2f}$".format(slope) + "\n" + \
#                          r" $R^2={:.2f}$".format(r_value ** 2))
#
#     plt.legend(bbox_to_anchor=(1, 0.4, 0.25, 0.18),
#                loc=7,  # 'center right'
#                ncol=1,
#                borderaxespad=None)


def plot_regression(xdata, ydata, interval, scaled=True):
```

```
    data = [(xdata[i], ydata[i]) for i in range(len(xdata))]
    data = [row for row in data if row[0] > 0 and row[1] > 0]
    # linear regression
    for i in range(len(interval) - 1):
        if not scaled: break
        # selecting data for the interval
        M = [row for row in data if interval[i] <= row[0] <= interval[i + 1]]  #
(n, 2)
        # checking there is data
        if len(M) < 2: continue
        # fit
        xdata, ydata = [[row[j] for row in M] for j in range(len(M[0]))]  # (#
2, n)
        popt, pcov = selection(xdata, ydata)
        # print(np.sqrt(np.diag(pcov)))
        x = np.linspace(interval[i], interval[i + 1])
        plt.plot(x, (lambda x, B, m, C: B * x ** m + C)(x, *popt), '--',
                   label=r"fit{}".format(i + 1) + "\n" + \
                         r" $m ={:.2f}$".format(popt[-2]) + "\n")# + \
                         # r" $R^2={:.2f}$".format(r_value ** 2))

    plt.legend(bbox_to_anchor=(1, 0.4, 0.25, 0.18),
               loc=7,   # 'center right'
               ncol=1,
               borderaxespad=None)


def coef_determination(xdata, ydata, *popt):
    f = lambda x, B, m: B*x**m
    sigmax = sum((f(xdata, *popt) - xdata)**2)
    sigmay = sum((f(ydata, *popt) - ydata)**2)
    return


def sumcumrand(x, y):
    """This function compute the area from points randomly distributed on x-
axis."""
    return x, y


def selection(xdata, ydata):
    """This select the correct parameters for p0"""
    try:
        popt1, pcov1 = curve_fit((lambda x, B, m, C: B * x ** m + C), xdata,
ydata, p0=(1, 6, 1))
        # print("part 1")
    except:
        popt1, pcov1 = None, None
        # print("fail 1")
    if popt1 is None:
        try:
            popt2, pcov2 = curve_fit((lambda x, B, m, C: B * x ** m + C), xdata,
ydata, p0=(+1, -1, 1))
            # print("part 2")
        except:
```

```
            popt2, pcov2 = None, None
            # print("fail 2")
    # if popt2 is  None:
    try:
        popt3, pcov3 = curve_fit((lambda x, B, m, C: B * x ** m + C), xdata,
ydata, p0=(-1, -1, 1))
        # print("part 3", np.sqrt(np.diag(pcov3)))
    except:
        popt3, pcov3 = None, None
        # print("fail 3")
    try:
        popt4, pcov4 = curve_fit((lambda x, B, m, C: B * x ** m + C), xdata,
ydata, p0=(1, 1, -1))
        # print("part 4", np.sqrt(np.diag(pcov4)))
    except:
        popt4, pcov4 = None, None
        # print("fail 4")

    if popt1 is not None:
        popt = popt1
        pcov = pcov1
        # print("selected 1", np.sqrt(np.diag(pcov1)))
    elif popt2 is not None:
        popt = popt2
        pcov = pcov2
        # print("selected 2", np.sqrt(np.diag(pcov2)))
    elif popt3 is not None:
        popt = popt3
        pcov = pcov3
        # print("selected 3", np.sqrt(np.diag(pcov3)))
    elif popt4 is not None:
        popt = popt4
        pcov = pcov4
        # print("selected 4", np.sqrt(np.diag(pcov4)))
    elif [popt1, popt2, popt3] == [None, None, None]:
        print("None of the worked.")
        return (None, None, None), (None, None, None)

    # print("-----------------")
    return  popt, pcov
```

Hmodule.py

```
# This is a file that optimize the parameters.
import numpy as np
import routine as rt
import scipy


def Regression(xdata, ydata, separator, option=""):
    """This returns R^2, a measure of how accurate is approximation."""
    data = [(x, y) for x, y in zip(xdata, ydata)]
    if data == []:
```

```
        interval = []
    elif isinstance(separator, tuple):
        interval = [xdata[0], *separator, xdata[-1]]
    else:
        interval = [xdata[0], separator, xdata[-1]]

    R2 = []
    for i in range(len(interval)-1):
        if option == "sum":
            pass
        elif option == "tail":
            if i == 0:
                continue
        elif option == "head":
            if i == 1:
                continue

        # selecting data for the invertal
        M = [row for row in data if interval[i] <= row[0] <= interval[i+1]]
# (n, 2)

        # checking there is data
        if len(M) < 2: continue

        # linear regression
        xdata, ydata = [[row[i] for row in M] for i in range(len(M[0]))]  # (2,
n)

        # R^2
        slope, intercept, r_value, p_value, std_err =
scipy.stats.linregress(np.log(xdata), np.log(ydata))
        R2.append(r_value**2)

    if option == "": return R2
    return sum(R2)


def regression_m(xdata, ydata, separator, option=""):
    """This function returns the value of m_1,2.
    options: head
    """

    data = [(x, y) for x, y in zip(xdata, ydata)]

    if data == []:
        interval = []
    elif isinstance(separator, tuple):
        interval = [xdata[0], *separator, xdata[-1]]
    else:
        interval = [xdata[0], separator, xdata[-1]]

    m_i = []
    for i in range(len(interval)-1):
        if option == "tail":
            if i == 0:
```

```
                continue
        elif option == "head":
            if i == 1:
                continue

        # selecting data for the invertal
        M = [row for row in data if interval[i] <= row[0] <= interval[i+1]]
# (n, 2)

        # checking there is data
        if len(M) < 1: continue

        # linear regression
        xdata, ydata = [[row[i] for row in M] for i in range(len(M[0]))]  # (2,
n)

        # R^2
        slope, intercept, r_value, p_value, std_err =
scipy.stats.linregress(np.log(xdata), np.log(ydata))
        m_i.append(slope)
        # sstandard deviation
        # popt, pcov = cm.selection(xdata, ydata)
        # sigma_m = np.sqrt(np.diag(pcov))[1]
        # R2.append(sigma_m)
    return m_i


def InterEvent(tm, pf, bins='auto', s=4, scaled=False, uniform=False):
    """ If bins and s are sequences, they must have the same length."""

    # sequence of cuts (s)
    if isinstance(s, str) and "(" in s and ")" in s:
        s = eval(s)
    if isinstance(s, int):
        sequence = range(1, s + 1)
    elif isinstance(s, float):
        sequence = range(1, int(s) + 1)
    else:
        sequence = s

    xdata, ydata = [], []
    for s_min in sequence:
        x, y = rt.InterEventDistribution(tm, pf, bins=bins, s_min=s_min,
uniform=uniform, scaled=scaled)
        # issue warning
        if not x: continue
        # gathering data
        xdata += x
        ydata += y

    # setting data
    data = [[xdata[i], ydata[i]] for i in range(len(xdata))]    # (n, 2)

    # checking data for loglog regression,
    data = [row for row in data if row[1] > 0]
```

```
    data.sort()  # sorted by x-data
    return [d[0] for d in data], [d[1] for d in data]


def opti0(xdata, ydata, option="sum"):
    """
    :param xdata:
    :param ydata:
    :param option: 'sum', 'tail', 'head'
    :return:
    """
    xa = 1.
    xb = 4.
    intvl = np.linspace(xa, xb, 1000)
    L = [Regression(xdata, ydata, x, option=option) for x in intvl]
    x0 = L.index(max(L))
    x0 = intvl[x0]
    return x0


def opti1(xdata, ydata, option="sum", bounds=(1.3, 5)):
    """Selecciona el maximo
    xdata, ydata son los puntos sobre los ccuaels se hace regresion.
    El intervalo xdata se hace otoman mpuntos en los el absica a de los cuales
se puede ahcer regression. Esto determina los intervalo
     en el cual se ahce la regression. Debido a que no es una fucinon continua
pero est plana a trozos.
    xdata = np.log(np.array(xdata))
    ydata = np.log(np.array(ydata))
    """
    xdata = np.array(xdata)
    ydata = np.array(ydata)
    xpath = (xdata[1:] + xdata[:-1]) / 2

    # this was another option
    # xpath = xpath[2:-2] # The regression with two curves needs at least 6
points.

    xpath = [x for x in xpath if bounds[0] <= x <= bounds[1]]

    if xpath == []:
        return 0

    L = [Regression(xdata, ydata, sep, option=option) for sep in xpath]
    x0 = L.index(max(L))
    x0 = xpath[x0]
    return x0


def selecting_parameters(tm, pf,  bins="default", s=range(3,7), uniform=False,
option='sum', bounds=(.95, 5)):
    """Return parameters sep, b, s"""
    # cuts
    if isinstance(s, int):
        sequence = range(1, s+1)
```

```python
    else:
        sequence = s

    # bins
    if bins == "default":
        bin_method = ["fd", "doane", "rice", "sturges", "sqrt"]
    elif isinstance(bins, str):
        bin_method = (bins,)
    elif isinstance(bins, int):
        bin_method = (bins,)
    else:
        bin_method = bins

    parameters = []
    for b in bin_method:
        for s in sequence:
            xdata, ydata = InterEvent(tm, pf, bins=b, s=s, scaled=True,
uniform=uniform)
            sep = opti1(xdata, ydata, option=option, bounds=bounds)
            mP = Regression(xdata, ydata, sep, option=option)
            parameters.append((mP, sep, b, s))

    parameters.sort()

    return parameters[-1][1:]


def main(*arg):
    """Organize the options for availables and returns the categery and scale
    H1 is restricted to a single option"""

    category = ["Agresiones", "Enfrentamientos", "Ejecuciones", "Todo"]
    scale = ["edo", "mpio"]
    hypothesis = [1, 0, "0", "1", "h0", "h1", "H0", "H1"]
    cat, scl, hypo = [], [], "0"
    if arg == tuple():
        return category, scale, "0"
    else:
        for a in arg:
            if a in category:
                cat.append(a)
            elif a in scale:
                scl.append(a)
            elif a in hypothesis:
                if "1" in a:
                    hypo = "1"
                else:
                    hypo = "0"

        if cat == []:
            cat = category
        if scl == []:
            scl = scale

        return cat, scl, hypo
```

```python
def average_separation(cat, scl, number):
    """number must be an integer or a list with two numbers

    number: int
    number: (from, up_to)
    """
    if isinstance(number, int):
        interval = (0, number)
    else:
        interval = number

    keys = rt.keysSorted(cat, scl)
    keys = keys[interval[0]:interval[-1]]

    separator = []

    for edo, mpio, sep in zip(keys.edo, keys.mpio, keys.sep):
        separator.append(sep)

    sep = np.average(separator)
    return sep


def single_separator(cat, scl, counter):
    # individual separator
    keys = rt.keysSorted(cat, scl)
    keys = keys[counter-1:counter]

    for edo, mpio, sep in zip(keys.edo, keys.mpio, keys.sep):
        return sep


def unique_separation(edo=25, mpio=18, cat=None, scl=None):
    """This function return specific separation for given characteristics.

    If every parametar is None ir returns None"""

    # No arguments
    if cat==None and scl==None:
        return None

    # list of keys
    keys = rt.keyResults(scale=scl, category=cat, edo=None, mpio=None,
criteria="Todo",
                         uniform=False, hypothesis=0)

    reference_edo = edo
    reference_mpio = mpio

    if scl == "mpio":
        keys = keys[:200]

    for edo, mpio, sep in zip(keys.edo, keys.mpio, keys.sep):
```

```
        if edo == reference_edo and mpio == reference_mpio:
            separator = sep
            break
    else:
        return None

    return separator
```

names.py

```
"""This modle gives the name given edo and mpio."""
import pandas as pd
from reference import reference_folder
from numpy import nan
import os


def language(func):
    """This removes accents."""

    def inner(*arg):
        s = func(*arg)
        s = s.replace("á", "a")
        s = s.replace("é", "e")
        s = s.replace("í", "i")
        s = s.replace("ó", "o")
        s = s.replace("ú", "u")
        s = s.replace("(país)", "(country)")
        return s

    return inner


def translate(category):
    """Translate into spanish common words"""
    D = {"Todo":"Todo", "Agresiones":"Agresiones",
"Enfrentamientos":"Enfrentamientos", "Ejecuciones":"Ejecuciones"}
    return D[category]


def code2name(edo, mpio=0):
    """This loads the file for """
    # code for the states
    cwd = os.getcwd()
    os.chdir(reference_folder)
    os.chdir("Data")
    df = pd.read_csv("entidad_{}.csv".format(edo), index_col="mun")
    os.chdir(cwd)
```

```python
    if mpio == 0:
        return df.loc[mpio, "nom_ent"]
    else:
        return df.loc[mpio, "nom_mun"]


def name2code(edo_name, mpio_name=None):
    """"""
    list_of_states = [
        "México (país)",
        "Aguascalientes",
        "Baja California",
        "Baja California Sur",
        "Campeche",
        "Coahuila de Zaragoza",
        "Colima",
        "Chiapas",
        "Chihuahua",
        "Distrito Federal",
        "Durango",
        "Guanajuato",
        "Guerrero",
        "Hidalgo",
        "Jalisco",
        "México",
        "Michoacán de Ocampo",
        "Morelos",
        "Nayarit",
        "Nuevo León",
        "Oaxaca",
        "Puebla",
        "Querétaro",
        "Quintana Roo",
        "San Luis Potosí",
        "Sinaloa",
        "Sonora",
        "Tabasco",
        "Tamaulipas",
        "Tlaxcala",
        "Veracruz de Ignacio de la Llave",
        "Yucatán",
        "Zacatecas"]

    edo = list_of_states.index(edo_name)

    if mpio_name is None:
        return edo, 0
    else:
        cwd = os.getcwd()
        os.chdir(reference_folder)
        os.chdir("Data")
        df = pd.read_csv("entidad_{}.csv".format(edo), index_col="nom_mun")
        os.chdir(cwd)
        return edo, df.loc[mpio_name, "mun"]
```

```python
# @language
def name_place(edo=None, mpio=None):
    """This return the name indicated.

    For mexico the values is edo=0, mpio=0"""
    if pd.isna(mpio): mpio = None
    if pd.isna(edo): edo = None
    if edo == 0 and mpio == 0:
        return "México (país)"
    elif mpio == 0:
        mpio = None

    if edo is not None and mpio is not None:
        return "{1}, {0}".format(code2name(edo), code2name(edo, mpio))
    elif edo is not None and mpio is None:
        return code2name(edo)


# @language_reverse
def code_place(place_name):
    """Return the keys for the place named.

    :returns
    --------
    out: list
        [edo, mpio]
    """
    if place_name == "México (país)":
        return 0, 0

    place_name = place_name.split(",")

    if len(place_name) == 2:    # municipality
        place_name[1] = place_name[1][1:]
        edo_name = place_name[1]
        mpio_name = place_name[0]
        return name2code(edo_name, mpio_name)
    else:   # state
        edo_name = place_name[0]
        return name2code(edo_name)


def rangeEdo(edo):
    """"Returns the number of mipos.
    """
    cwd = os.getcwd()
    os.chdir(reference_folder)
    os.chdir("Data")
    df = pd.read_csv("entidad_{}.csv".format(edo), index_col="mun")
    os.chdir(cwd)

    for mpio in df.index.values:
        yield mpio
```

routine.py

```python
"""Common Functions"""

import datetime
import scipy
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from scipy import stats
from os import getcwd, chdir
from names import name_place, code_place, translate
from reference import reference_folder


def Deaths(flnm, key, edo=None, mpio=None):
    # new notation
    if mpio is 0:
        mpio = None
    if edo is 0:
        edo = None

    data = np.loadtxt(flnm, delimiter='\t', skiprows=1, dtype=np.int32)
    if edo is None and mpio is None:  # It is the country.
        pass  # Borth keys are needed.
    elif edo is not None and mpio is not None:  # It is a municipality.
        data = data[data[:, 5] == edo]  # state column 5
        data = data[data[:, 6] == mpio]  # municipality column 6
    elif edo is not None and mpio is None:  # It is a state.
        data = data[data[:, 5] == edo]  # state column 5
    year = data[:, 4]  # year column 4
    month = data[:, 3]  # month column 3
    day = data[:, 2]  # day column 2
    tm = ["{}-{}-{}".format(year[i], month[i], day[i]) for i in range(len(day))]
    tm = [t.split("-") for t in tm]
    tm = [[int(t[i]) for i in range(3)] for t in tm]
    tm = [datetime.datetime(*t) for t in tm]
    pf = data[:, key]  # deaths column key
    pf = pf.tolist()
    return tm, pf



def import_data(category, edo=None, mpio=None):
    # It is assumed that the folder data is where the rutine.py is.
    cwd = getcwd()
    chdir(reference_folder)
    chdir("Data")

    # files and column with deaths
    D = {"Agresiones": [("BD_A-A.txt", 10)],
         "Enfrentamientos": [("BD_A-E.txt", 8)],
```

```python
          "Ejecuciones": [("BD_A-X06.txt", 11),
                          ("BD_A-X07.txt", 11),
                          ("BD_A-X08.txt", 13),
                          ("BD_A-X09.txt", 13),
                          ("BD_A-X10.txt", 13),
                          ("BD_A-X11.txt", 13)]
          }
    D["Todo"] = D["Agresiones"] + D["Enfrentamientos"] + D["Ejecuciones"]

    tm, pf = [], []
    # gathering data
    for fnm, ky in D[category]:
        temp1, temp2 = Deaths(fnm, ky, edo, mpio)
        tm += temp1
        pf += temp2
    # tm = [(t - tm[0]).days for t in tm]     # Express in days
    M = [[tm[i], pf[i]] for i in range(len(tm))]
    M = [m for m in M if m[1] < 9999]  # 9999: undefined
    M.sort()
    # M = [[m[0]-M[0][0]+1, m[1]] for m in M]     # Sets the first day as 1.
    chdir(cwd)
    return [m[0] for m in M], [m[1] for m in M]


def same_event_0(tm, pf):
    """Events are kept the same."""
    return tm, pf


def same_event_1(tm, pf):
    """ Events are summed.

    It is considered that (tm, pf) are sorted."""
    try:
        ed, td = [pf[0]], [tm[0]]
    except IndexError:
        return [], []
    j = 0
    for i in range(len(pf) - 1):
        if tm[i] == tm[i + 1]:
            ed[j] += pf[i + 1]
        else:
            td.append(tm[i + 1])
            ed.append(pf[i + 1])
            j += 1
    return td, ed


def same_event_2(tm, pf):
    """ The highest event is kept. """
    # matrix of events
    M = [[tm[i], pf[i]] for i in range(len(tm))]
    # Selecting the highest.
    try:
        newList, idx = [], [M[0][1]]
```

```
    except IndexError:
        return [], []
    for i in range(len(M) - 1):  # The last event is added separately.
        if M[i][0] == M[i + 1][0]:  # If the last two events are simultaneous
are omitted. They trigger **continue**.
            idx.append(M[i][1])
            continue
        newList.append([M[i][0], max(idx)])  # The event i-th is added.
        idx = [M[i + 1][1]]  # The events i, i+1 are different.
    # comparison of last elements
    if len(M) == 1:
        pass
    elif M[-2][0] == M[-1][0]:
        newList.append([M[-1][0], max(M[-2][1], M[-1][1])])
    else:
        newList.append(M[-1])
    return [nl[0] for nl in newList], [nl[1] for nl in newList]


def different_event_1(tm, pf):
    """Events are distinct. Time inter event is added to the distribution."""
    # matrix of events
    M = [[tm[i], pf[i]] for i in range(len(tm))]

    for i in range(len(M) - 1):
        if M[i][0] == M[i][0]:
            print(M[i][0])
    M = [[row[i] for row in M] for i in range(len(M[0]))]
    return M[0], M[1]


def DeathPlot(tm, pf, NamePlot, Category):
    plt.title(NamePlot +
              "\ncategoría: {}".format(translate(Category)))
    plt.xlabel("time (día)")
    plt.ylabel(r"magnitud $s$")
    # par.set_ylabel(r"events")
    plt.semilogy(tm, np.cumsum(pf), color='r')
    plt.bar(tm, pf, width=5, color='k', log=True)
    plt.tight_layout()
    plt.savefig("{}_{}_Day.png".format(NamePlot, Category))
    plt.clf()


def PlaceStat(category, edo=None, mpio=None):
    """Gives an np.array with the statiscial information.
    count, mean, std, min, 25%, 50%, 75%, max"""
    cols = ["place", "count", "mean", "std", "min", "25%", "50%", "75%", "max",
"Total deaths"]
    # "count" is the number of events.
    tm, pf = import_data(category, edo, mpio)
    place = name_place(edo, mpio)
    # Determinar si hay datos
    if not tm:
        df = pd.DataFrame(np.zeros((1, len(cols) - 1)), columns=cols[1:])
```

```
        df["place"] = place
        return df[cols]
    else:
        df = pd.DataFrame(pf, index=tm)

    s = df.describe().T
    s["place"] = place
    s["Total deaths"] = df.sum()[0]
    s = s[cols]
    return s


def InterEventDistribution(tm, pf, bins='auto', s_min=4, uniform=False,
scaled=False):
    """tm, pf are ready for the distribution"""
    # matrix of data
    M = [[tm[i], pf[i]] for i in range(len(tm))]  # (n, 2)
    # selecting data
    M = [row for row in M if row[1] >= s_min]
    # size warning
    if len(M) < 1:
        return [], []
    # inter-event time
    ts = np.array([row[0] for row in M])
    tau = [t.days for t in (ts[1:] - ts[:-1])]
    # size warning
    if (ts[-1] - ts[0]).days == 0:
        return [], []
    # <r> and possible interpretation.
    r = len(M) / (ts[-1] - ts[0]).days
    # distribuition
    hist, bins = np.histogram(tau, bins=bins, normed=True)

    x = .5 * (bins[1:] + bins[:-1])  # tau
    y = np.array(hist)  # P(tau)
    # <r> scaling
    if scaled is True:
        x = x * r  # tau * <r>
        y = y / r  # P(tau) / <r>
    return x.tolist(), y.tolist()


def PlotInterEventDistribution(tm, pf, category, namePlot, bins=10, s=4,
separator=10,
                               save=False, scaled=False, uniform=False):
    """s int  ->  range(1, s+1)"""
    try:
        edo, mpio = code_place(namePlot)
    except:
        edo, mpio = -1, -1

    # plot description
    plt.figure()

    if scaled:
```

```
        plt.title("\nDistribución escalada de tiempo entre eventos"+
                  "\n" + namePlot +
                  "\ncategoría: {}".format(translate(category)) )
        plt.xlabel(r"$\tau r$")
        plt.ylabel(r"$P(\tau)/r$")
        namePlot += "-scaled"
        namePlot = namePlot.replace(" ", "_")
    else:
        plt.title("\nDistribución de tiempo entre eventos" +
                  "\n" + namePlot +
                  "\ncategoría: {}".format(translate(category)))
        plt.xlabel(r"$\tau$")
        plt.ylabel(r"$P(\tau)$")
        namePlot = namePlot.replace(" ", "_")
    # sequence of cuts (s)
    if isinstance(s, str) and "(" in s and ")" in s: s = eval(s)
    if isinstance(s, int):
        sequence = range(1, s + 1)
    elif isinstance(s, float):
        sequence = range(1, int(s) + 1)
    else:
        sequence = s

    xdata, ydata = [], []
    for s_min in sequence:
        x, y = InterEventDistribution(tm, pf, bins=bins, s_min=s_min,
uniform=uniform, scaled=scaled)
        # issue warning
        if not x:
            continue
        # log-log plot
        #plt.loglog(x, y, 'o', label=r"$s_{min}=$" + str(s_min))
        plt.loglog(x, y, 'o', label=r"$s_{min}=$" + "{:.2E}".format(s_min))
        # gathering data
        xdata += x
        ydata += y

    # setting data
    data = [[xdata[i], ydata[i]] for i in range(len(xdata))]  # (n, 2)
    # checking data for loglog regression,
    data = [row for row in data if row[1] > 0]
    data.sort()  # sorted by x-data

    try:
        if isinstance(separator, tuple):
            interval = [data[0][0], *separator, data[-1][0]]
        else:
            interval = [data[0][0], separator, data[-1][0]]
    except IndexError:
        interval = [None]  # var data is empty

    for i in range(len(interval) - 1):
        if not scaled: break
        # selecting data for the invertal
        M = [row for row in data if interval[i] <= row[0] <= interval[i + 1]]  #
```

```
(n, 2)
        # checking there is data
        if len(M) < 2:
            # print("few points at interval", [interval[i], interval[i+1]],
len(M))
            continue
        # linear regression
        xdata, ydata = [[row[i] for row in M] for i in range(len(M[0]))]  # (2,
n)
        slope, intercept, r_value, p_value, std_err =
stats.linregress(np.log(xdata), np.log(ydata))
        # plot fit
        x = np.linspace(interval[i], interval[i + 1])
        plt.loglog(x, np.exp((lambda x, m, b: m * x + b)(np.log(x), slope,
intercept)), '--',
                   label=r"fit{}".format(i + 1) + "\n" + \
                         r" $m ={:.2f}$".format(slope) + "\n" + \
                         r" $R^2={:.2f}$".format(r_value ** 2))
    plt.legend(bbox_to_anchor=(1.025, 1.),
               loc='upper left',
               borderaxespad=0.)
    # output format
    plt.tight_layout()
    if save:
        plt.savefig("{}-({}, {})-{}.png".format(category, edo, mpio, namePlot),
bbox_inches='tight')
        plt.clf()
    else:
        plt.show()
        plt.clf()
    plt.close()


def keys(category, scale):
    """Return a table edo or mpio.

    Return a talbe with place, edo, mpio, count, pf"""
    cwd = getcwd()
    chdir(reference_folder)
    chdir("Data")
    keys = pd.read_csv("TableSorted-{}-{}.csv".format(scale, category))
    if scale == "edo":
        keys.loc[:, "mpio"] = [None] * keys.mpio.size
    if scale == "country":
        keys.loc[:, "edo"] = [0] * keys.mpio.size
        keys.loc[:, "mpio"] = [0] * keys.mpio.size
    keys["place"] = [name_place(edo, mpio) for edo, mpio in zip(keys.edo,
keys.mpio)]
    chdir(cwd)
    return keys


def keysSum(category, scale, hypothesis=1):
    """Return a table of the results in optimize_cumulative_sum"""
    cwd = getcwd()
```

```
    chdir(reference_folder)
    chdir("Results")
    keys = pd.read_csv("cumulative_sum_separator.csv")
    keys = keys[keys.loc[:, "category"] == category]
    keys = keys[keys.loc[:, "scale"] == scale]
    # keys = keys[keys.loc[:, "hypothesis"] == int(hypothesis)]
    if scale == "edo": keys.loc[:, "mpio"] = [None] * keys.mpio.size
    keys["place"] = [name_place(edo, mpio) for edo, mpio in zip(keys.edo,
keys.mpio)]
    chdir(cwd)
    return keys


def keyResults(scale="mpio", category=None, edo=None, mpio=None, criteria="sum",
uniform="False", hypothesis=1,
               results_file="results.csv"):
    """This takes keys form the file results.csv where the parameters have
computed.

    This keys are meant to be used for the computation of the integral of the
distribution."""
    cwd = getcwd()
    chdir(reference_folder)
    chdir("Data")
    keys = pd.read_csv(results_file)
    keys = keys[keys.loc[:, "scale"] == scale]
    keys = keys[keys.loc[:, "uniform"].apply(str) == str(uniform)]
    keys = keys[keys.loc[:, "category"] == category]
    keys = keys[keys.loc[:, "criteria"] == criteria]
    keys = keys[keys.loc[:, "hypothesis"] == int(hypothesis)]

    if scale == "country":
        keys.loc[:, "edo"] = [0] * keys.mpio.size
        keys.loc[:, "mpio"] = [0] * keys.mpio.size
    elif scale == "edo":
        # keys.loc[:, "mpio"] = [None] * keys.mpio.size   # alternative
        keys.loc[:, "mpio"] = [0] * keys.mpio.size
        keys.loc[:, "s"] = keys.loc[:, "s"].apply(float)
    else:   # mpio
        keys.loc[:, "mpio"] = keys.loc[:, "mpio"].apply(int)
        keys.loc[:, "s"] = keys.loc[:, "s"].apply(float)

    if edo is not None:
        keys = keys[keys.loc[:, "edo"] == edo]
    if mpio is not None:
        keys = keys[keys.loc[:, "mpio"] == mpio]

    keys["place"] = [name_place(edo, mpio) for edo, mpio in zip(keys.edo,
keys.mpio)]
    chdir(cwd)
    return keys


def CutData(intvl: list, tm: list, pf: list) -> list:
    Data = []
```

```
    M = np.array((tm, pf))
    M = M.T
    for i in range(len(intvl) - 1):
        idx1 = tm.index(intvl[i])
        idx2 = tm.index(intvl[i + 1])
        Data.append(M[idx1:idx2].tolist())
    Data[-1].append(M[-1].tolist())
    return Data


def fit_selection(xdata, ydata):
    """This select the correct parameters for p0"""
    try:
        popt1, pcov1 = scipy.integrate.curve_fit((lambda x, B, m, C: B * x ** m
+ C), xdata, ydata, p0=(1, 6, 1))
        # print("part 1")
    except:
        popt1, pcov1 = None, None
        # print("fail 1")
    if popt1 is None:
        try:
            popt2, pcov2 = scipy.integrate.curve_fit((lambda x, B, m, C: B * x
** m + C), xdata, ydata, p0=(+1, -1, 1))
            # print("part 2")
        except:
            popt2, pcov2 = None, None
            # print("fail 2")
    # if popt2 is  None:
    try:
        popt3, pcov3 = scipy.integrate.curve_fit((lambda x, B, m, C: B * x ** m
+ C), xdata, ydata, p0=(-1, -1, 1))
        # print("part 3", np.sqrt(np.diag(pcov3)))
    except:
        popt3, pcov3 = None, None
        # print("fail 3")
    try:
        popt4, pcov4 = scipy.integrate.curve_fit((lambda x, B, m, C: B * x ** m
+ C), xdata, ydata, p0=(1, 1, -1))
        # print("part 4", np.sqrt(np.diag(pcov4)))
    except:
        popt4, pcov4 = None, None
        # print("fail 4")

    if popt1 is not None:
        popt = popt1
        pcov = pcov1
        # print("selected 1", np.sqrt(np.diag(pcov1)))
    elif popt2 is not None:
        popt = popt2
        pcov = pcov2
        # print("selected 2", np.sqrt(np.diag(pcov2)))
    elif popt3 is not None:
        popt = popt3
        pcov = pcov3
        # print("selected 3", np.sqrt(np.diag(pcov3)))
```

```python
    elif popt4 is not None:
        popt = popt4
        pcov = pcov4
        # print("selected 4", np.sqrt(np.diag(pcov4)))
    elif [popt1, popt2, popt3] == [None, None, None]:
        print("None of the worked.")

    # print("-----------------")
    return popt, pcov
```

Producción de resultados

En esta sección solo se agrega un archivo de los varios que producen estas gráficas para no hacer mas extenso este anexo. Los demas archivos que producen este tipo de gráficas tienen el mismo tipo de estructura. Se pueden rehacer los demas usando los módulos.

InterEvent_Distribution-OneLine.py

```python
"""This is a file optimize the parameters and returns table of R_2^2's a
file."""

import routine as rt
import names
import os

category = ["Ejecuciones", "Todo", "Agresiones", "Enfrentamientos"]
scale = ["mpio", "edo", "country"]
hypothesis = ["0"]# , "1", "2"]
uniform = [False] #, True]
criteria = ["sum"] #, "tail", "sum"]

for hyp in hypothesis:
    for uni in uniform:
        for cri in criteria:
            for scl in scale:
                # making folder
                os.chdir(rt.reference_folder)
                os.chdir("Results")
                folder = "IEDS-OneLine-" + str(scl)

                # check directory  Is it necessary? Just for the first time.
                try:
                    os.makedirs(folder)
                except FileExistsError:
                    pass

                os.chdir(folder)

                for cat in category:
                    # directory
```

```python
                    cwd = os.getcwd()
                    os.chdir(rt.reference_folder)
                    os.chdir("Data")

                    # list of keys
                    keys = rt.keyResults(scale=scl, category=cat)
                    # keys = keys[keys.loc[:, r"R_1^2"] >= 0.7]

                    if scl == "mpio":
                        keys = keys[:20]

                    os.chdir(cwd)

                    for edo, mpio, b, s, sep in zip(keys.edo, keys.mpio,
keys.bins, keys.s, keys.sep):
                        # selecting results
                        # if scl == "mpio":
                        #     if edo != 25 or mpio != 18:
                        #         continue
                        # elif scl == "edo":
                        #     if edo != 25:
                        #         continue

                        # adding results
                        if scl is "country": s = range(1, 6)

                        # hypothesis
                        namePlace = names.name_place(edo, mpio)
                        tm, pf = rt.import_data(cat, edo, mpio)
                        if hyp == "0":
                            tm, pf = rt.same_event_0(tm, pf)
                        elif hyp == "1":
                            tm, pf = rt.same_event_1(tm, pf)
                        elif hyp == "2":
                            tm, pf = rt.same_event_2(tm, pf)

                        sep = tuple()

                        os.chdir(cwd)
                        rt.PlotInterEventDistribution(tm, pf, cat, namePlace,
bins="doane", s=s, separator=sep,
                                                                  save=True, scaled=True,
uniform=uni)
```