



**TECNOLÓGICO
NACIONAL DE MÉXICO®**



**TECNOLOGICO NACIONAL DE MEXICO.
INSTITUTO TECNOLÓGICO DE CULIACÁN.**

***OPTIMIZACIÓN DE COLOCACIÓN DE SENSORES PARA RIEGO EFICIENTE
EN GUASAVE, SINALOA, MEDIANTE ENJAMBRE DE PARTÍCULAS (PSO)***

TOPICOS DE INTELIGENCIA ARTIFICIAL 12-13.

DOCENTE: MORA FELIX ZURIEL DATHAN.

CLAVE DE LA MATERIA: GTD2102

NOMBRE DE LOS ALUMNOS:

- BORBÓN SÁNCHEZ EDGAR - 16171301
- MILLÁN LÓPEZ ANA KAREN - 20170985

UNIDAD: 3.

ACTIVIDAD: PROYECTO DE MAPEO DE OPTIMIZACIÓN DE RIEGO DE ENJAMBRE DE PARTÍCULAS.

CULIACÁN, SINALOA.

Contenido

Descripción del Problema	3
Justificación para Usar el Método PSO (Optimización por Enjambre de Partículas)	3
Diseño del Algoritmo	4
Preprocesamiento y Normalización de Datos	4
Función de Fitness (Función de Costo)	5
Implementación (fitness function):.....	5
Configuración y Ejecución del Enjambre	5
Resultados.....	6
Gráficos de Progreso	6
Mapa Final de Sensores	6
Informe de Consola	7
Interpretación de Zonas No Rentables.....	7
Análisis de Eficiencia.....	7
Análisis Metodológico	7
Análisis Paramétrico	7
Escenario 1 vs Escenario 2.....	8
Escenario 3	8
Escenario 4	8
Conclusiones.....	9
Repositorio	9

Descripción del Problema

En este proyecto se aborda un desafío central en la agricultura de precisión para la **Región de Guasave, Sinaloa**: la optimización de la eficiencia del riego. Dicha región, una llanura costera con altitudes que varían entre 10 y 50 metros, representa una potencia agrícola, destacando en la producción de **Maíz, Tomate y Chile**.

El objetivo fundamental es determinar la **colocación óptima de un número limitado de sensores** en el campo. La optimización de la solución no se mide exclusivamente por la distancia euclidiana, sino por la capacidad de la configuración de sensores para **capturar la variabilidad intrínseca del terreno** y, de esta forma, maximizar la eficiencia hídrica.

Para alcanzar este objetivo, el algoritmo debe considerar variables clave que afectan directamente la retención de humedad en el suelo:

1. **Topografía del Terreno** .- Aunque el terreno es predominantemente plano, existen **micro-variaciones en la elevación** que impactan significativamente la distribución y retención del agua. Estas variaciones son gestionadas en la implementación mediante la variable *Elevacion (m)*.
2. **Variabilidad del Suelo** .- Los suelos de Guasave, si bien son fértiles, enfrentan procesos de degradación. Un factor crítico es el **aumento de la salinidad** en áreas con drenaje deficiente, lo cual afecta directamente la capacidad del suelo para retener humedad. El algoritmo implementado aborda esta problemática utilizando la variable *Salinidad (dS/m)*.
3. **Distribución de Cultivos** .- Los diferentes cultivos tienen distintas necesidades hídricas. Por lo tanto, el algoritmo debe identificar las ubicaciones óptimas de forma independiente para cada tipo de cultivo (*Maíz, Chile y Tomate*).

El problema, entonces, se define como la búsqueda de un conjunto de coordenadas (*Latitud, Longitud, Elevación, Salinidad*) para cada sensor que, en conjunto, **minimicen la diferencia promedio** desde cualquier punto de un cultivo hasta su sensor más cercano, considerando estas cuatro dimensiones críticas.

Justificación para Usar el Método PSO (Optimización por Enjambre de Partículas)

Para abordar este complejo problema de optimización, se seleccionó el algoritmo de **Optimización por Enjambre de Partículas (PSO)**. La elección se justifica por las siguientes razones:

1.- Naturaleza del Espacio de Búsqueda:

El problema presenta un espacio de soluciones continuo y de alta dimensionalidad. Cada sensor requiere la optimización de **4** variables (*Lat, Lon, Elevación, Salinidad*). Al colocar **2** sensores por cultivo, una solución para un solo cultivo ya tiene 8 dimensiones. El espacio de búsqueda es demasiado extenso para métodos exhaustivos.

2.- Eficiencia en Problemas Complejos:

PSO es un algoritmo metaheurístico conocido por su eficiencia para encontrar soluciones robustas y casi óptimas en espacios de búsqueda complejos y no lineales, sin necesidad de conocer la derivada de la función objetivo.

3.- Convergencia Rápida:

A diferencia de otros algoritmos evolutivos, **PSO** a menudo converge más rápidamente hacia una buena solución. Su mecanismo, que combina la experiencia individual de cada **partícula** (solución candidata) con la experiencia colectiva del **enjambre**, permite una exploración y explotación equilibrada.

4.- Facilidad de Implementación:

El concepto de **PSO** es relativamente simple y su implementación es directa, especialmente con el uso de librerías especializadas como *pyswarms*, tal como se recomendó en las especificaciones del proyecto. Esto permite centrarse en el modelado del problema (función de *fitness*) en lugar de en la mecánica del optimizador.

5.- Escalabilidad:

El método es escalable. Si en el futuro se desea aumentar el número de sensores o incluir más variables (ej. *pH, materia orgánica*), el algoritmo puede adaptarse simplemente ajustando el número de dimensiones de la solución.

Diseño del Algoritmo

El diseño de la solución se estructura en los siguientes pasos clave:

Preprocesamiento y Normalización de Datos

1.- Carga de Datos:

Se inicia cargando un archivo *data.csv* que contiene los datos de los puntos del terreno, incluyendo *Latitud, Longitud, Elevacion (m), Salinidad (dS/m)* y el tipo de *Cultivo*.

2.- Normalización:

Las variables para optimizar tienen rangos y unidades muy diferentes (*grados* para coordenadas, *metros* para elevación, *dS/m* para salinidad). Para que el algoritmo de PSO las pondere de manera equitativa, es crucial normalizarlas. Se utiliza *MinMaxScaler* de *scikit-learn* para transformar todas las variables a una escala común de 0 a 1. Esto permite que la **distancia** en cada dimensión sea comparable.

3.- Separación por Cultivo:

El problema se descompone en sub-problemas independientes, uno por cada tipo de cultivo (*Maíz*, *Chile*, *Tomate*), y se crean vectores de datos separados para cada uno.

Función de Fitness (Función de Costo)

La función de *fitness* es el corazón del algoritmo **PSO**, ya que guía al enjambre hacia la mejor solución. Su objetivo es **evaluar la idoneidad (calidad)** de una configuración de sensores propuesta por una partícula.

Implementación (fitness function):

1. Una partícula del enjambre propone una solución como un vector unidimensional (ej., *[lat1, lon1, elev1, sal1, lat2, lon2, elev2, sal2]*).
2. Este vector se remodela a una matriz donde cada fila representa un sensor y sus 4 coordenadas (*numSensores*, 4).
3. Para cada punto de datos del terreno, se calcula la distancia euclidiana a cada uno de los sensores propuestos usando *np.linalg.norm*.
4. Se identifica la distancia mínima (el sensor más cercano a ese punto).
5. Esta distancia mínima se suma a un costo total.
6. El costo total acumulado para todos los puntos del terreno es el valor que retorna la función de *fitness* para esa partícula.

Configuración y Ejecución del Enjambre

Para cada cultivo, se configura y ejecuta un enjambre de partículas independiente.

1.- Parámetros del Enjambre

Se definen los hiperparámetros de **PSO**:

- *n_particles*: 50 partículas.
- *c1* (cognitivo): 0.5. Confianza en su propia mejor solución.
- *c2* (social): 0.3. Confianza en la mejor solución del enjambre.
- *w* (inercia): 0.9. Impulso para seguir en su dirección actual.

2.- Dimensiones y Límites:

- Las dimensiones de cada partícula se calculan como *numSensores* * 4.
- Debido a la normalización, los límites para cada dimensión son simplemente 0 (mínimo) y 1 (máximo).

3.- Optimización Iterativa:

Se utiliza *pyswarms.single.GlobalBestPSO*. El algoritmo se ejecuta en un bucle de **100 iteraciones**, optimizando un paso a la vez. Este enfoque se adopta con el fin de capturar y visualizar el progreso del enjambre cada **10 iteraciones**, permitiendo observar la convergencia de las soluciones propuestas hacia el óptimo.

Resultados

El algoritmo se ejecuta de manera exitosa, produciendo como salida una **propuesta de optimización estratégica** y las coordenadas óptimas para el número de sensores especificado para cada cultivo.

La **propuesta final** se presenta en forma de un plano que representa el campo y la posición geográfica de los cultivos. En este plano, se señalan con una '**X**' las posiciones propuestas que mejor satisfacen los requisitos del problema, teniendo en cuenta la distancia a los cultivos, la altitud del terreno y la salinidad del suelo.

El programa genera una serie de visualizaciones clave que funcionan como reportes de resultados:

Gráficos de Progreso

Por cada cultivo, se generan automáticamente carpetas (*progreso_maiz*, *progreso_chile*, *progreso_tomate*) que contienen imágenes guardadas cada **10 iteraciones**. Estos gráficos muestran la dispersión de los puntos de datos y la mejor posición encontrada por el enjambre hasta ese momento. Estas visualizaciones son efectivas e informativas, ya que permiten observar la convergencia del algoritmo.

Mapa Final de Sensores

Al finalizar las **100 iteraciones**, el programa genera un informe gráfico final consolidado ([Mapa de sensores final.png](#)). Esta visualización es clara, profesional y presenta de manera efectiva el resultado final:

- Muestra todos los puntos de datos de los tres cultivos, cada uno con un color distintivo (rojo para maíz, azul para chile, verde para tomate).

- Superpone las ubicaciones óptimas finales para los sensores de cada cultivo, utilizando marcadores 'X' del mismo color, lo que facilita su identificación.
- Incluye leyendas claras, etiquetas en los ejes y un título informativo, facilitando su comprensión e interpretación.

Informe de Consola

Los **costos finales (fitness)** para cada cultivo y el costo total se imprimen en la consola al finalizar la ejecución, proporcionando una métrica cuantitativa precisa del rendimiento de la solución encontrada.

Interpretación de Zonas No Rentables

Además de identificar las ubicaciones óptimas, el análisis visual de los resultados permite **determinar zonas no rentables para el cultivo o riego**. Se observó que, en casi todas las pruebas, la zona superior derecha del mapa consistentemente quedaba vacía de sensores. Se puede concluir que, bajo las variables analizadas, dicho terreno no es rentable para cultivar o regar.

Análisis de Eficiencia

El uso de **PSO** demostró ser altamente eficiente para este problema. El análisis de eficiencia se divide en **dos partes**: el **análisis metodológico** (la idoneidad del diseño) y el **análisis paramétrico** (el efecto de los hiperparámetros).

Análisis Metodológico

La **normalización de los datos** (utilizando *MinMaxScaler*) fue un paso crítico. Permitió al algoritmo tratar todas las variables (*grados*, *metros*, *dS/m*) con igual importancia, resultando en una convergencia más significativa y coherente.

El modelado del problema a través de una **función de costo (fitness)** basada en la **distancia euclidiana multidimensional** fue efectivo para representar el objetivo de maximizar la cobertura. El algoritmo se ejecuta sin errores y produce resultados óptimos y repetibles. Finalmente, la estructura del código es lógicamente modular, con secciones de carga de datos, función de fitness, bucles de optimización y graficación claramente diferenciados y comentados.

Análisis Paramétrico

Se ejecutó el algoritmo **4 veces** con distintos hiperparámetros para observar cómo la efectividad aumenta o disminuye dependiendo de los valores asignados.

La siguiente tabla resume los escenarios de prueba:

Escenario	Parámetros (c1, c2, w, p, i)	Costo Maíz (Fi- nal)	Costo Chile (Fi- nal)	Costo To- mate (Fi- nal)	Costo Total (Suma)
1	(0.5, 0.3, 0.9, 50, 100)	15.81	17.43	14.48	47.72
2	(1.5, 1.5, 0.7, 50, 100)	15.8	17.4	14.43	47.63
3	(0.5, 2.5, 0.4, 50, 100)	15.7	17.37	14.43	47.5
4	(1.5, 1.5, 0.7, 100, 200)	16.01	17.58	14.63	48.22

De esta tabla se pueden entender las siguientes observaciones:

Escenario 1 vs Escenario 2

En el **escenario 2**, los coeficientes cognitivos (**c1**) y social (**c2**) son más altos. Esto indica que las partículas se agruparán mucho más rápido. Aunque los costos finales son muy similares, el **escenario 2** tiende a alcanzar la solución ideal en menos iteraciones.

Escenario 3

En este caso, el coeficiente social (**c2 = 2.5**) es mucho más alto que el personal (**c1 = 0.5**). Esto fuerza a que cada partícula confíe más en la comunidad que en sí misma. Se limita la "curiosidad" o exploración individual para forzar una convergencia más rápida hacia el valor global que ya se considera bueno. Este escenario produjo el **costo total más bajo (47.50)**.

Escenario 4

En esta prueba se duplicaron tanto el número de partículas como las iteraciones (aumentando significativamente el tiempo de ejecución). Curiosamente, esta prueba produjo un **resultado peor (costo total de 48.22)** que las demás. Esto resalta la naturaleza del algoritmo: dado que las partículas se sueltan de manera aleatoria, es posible que en esta prueba **el punto de partida inicial haya sido mucho peor que en los otros escenarios**. Incluso con el doble de recursos, el trabajo a realizar fue más complejo debido a este punto de partida aleatorio.

Conclusiones

Este proyecto ha demostrado con éxito la aplicación funcional del algoritmo de **Optimización por Enjambre de Partículas (PSO)** para resolver un problema complejo del mundo real: la optimización de la colocación de sensores para el riego eficiente en la región agrícola de Guasave, Sinaloa.

El algoritmo logró cumplir con el objetivo principal al modelar de manera efectiva un problema multidimensional. Se integraron con éxito las variables clave especificadas: la **topografía** (modelada como *Elevacion (m)*), la **distribución de cultivos** (separando el análisis por *Maíz, Chile y Tomate*) y la **variabilidad del suelo** (representada por la *Salinidad (dS/m)*).

Un paso metodológico crucial fue la **normalización de los datos**, que permitió al algoritmo de **PSO** comparar de manera equitativa variables con unidades heterogéneas (*grados, metros y dS/m*). La función de *fitness (costo)* implementada, basada en la distancia euclidiana multidimensional, guio eficazmente al enjambre hacia soluciones que minimizan la distancia total desde cada punto del cultivo hasta su sensor más cercano.

La implementación no solo es funcional, sino que también produce resultados claros y accionables, como los mapas finales de sensores, que proporcionan una guía visual directa para la instalación en el campo. Se concluye que el proyecto es una aplicación exitosa de una técnica de inteligencia artificial bio-inspirada y sienta las bases para una herramienta práctica que puede generar un impacto tangible en la eficiencia del riego y la agricultura de precisión en la región.

Repositorio

<https://github.com/AnaKMLopez/ITCAKML-Topicos-IA>