

**UNIVERSIDAD DE SEVILLA**

**ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA INFORMÁTICA**

**Grado en Ingeniería Informática – Tecnologías Informáticas**

**Curso 2020/2021**

**OPTIMIZACIÓN DE FUNCIONES MEDIANTE ALGORITMO MULTIOBJETIVO BASADO EN AGREGACIÓN**

Realizado por:

Jaime Emilio Sala Mascort - jasalmas

Jesús Fernández García - jesfergar5

Correo electrónico:

Tu correo Jaime

[jesusfernandez199815@gmail.com](mailto:jesusfernandez199815@gmail.com)

**Sevilla, febrero de 2021**

Tabla de contenido

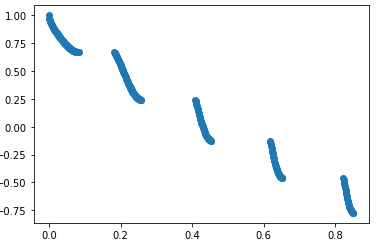
[1. Temática y problema por resolver 3](#_Toc64137684)

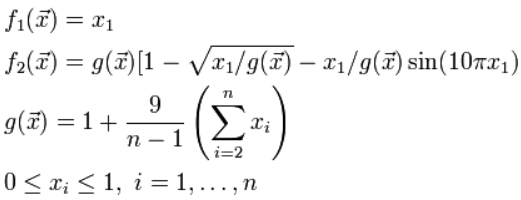
[2. Estructura del código y elementos utilizados 3](#_Toc64137685)

[3. Utilización del programa y ejemplos de uso 3](#_Toc64137686)

[4. Librerías utilizadas (extra indicar como instalarlas) 3](#_Toc64137687)

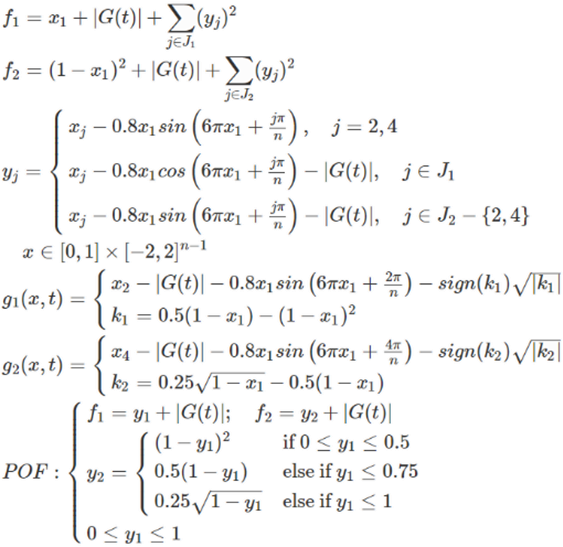
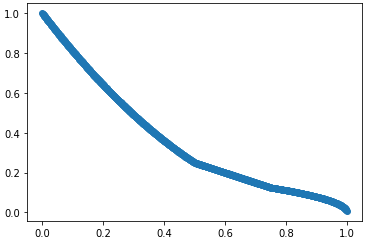
# 1. Temática y problema por resolver

La temática elegida para el trabajo ha sido la resolución de problemas de optimización multiobjetivo con y sin restricciones. En concreto, intentaremos obtener soluciones optimas para las funciones ZDT3 y CF6 en 4 y 16 dimensiones. Estas son sus fórmulas y sus frentes Pareto óptimos:



**Frente Pareto Óptimo ZDT3**

**Función ZDT3**



**Función CF6**

**Frente Pareto Óptimo CF6**

El algoritmo propuesto para buscar soluciones óptimas de estas funciones es un algoritmo evolutivo basado en agregación. Esto consiste en descomponer el problema en varios subproblemas de un solo objetivo. Para cada subproblema a resolver, la función objetivo será una agregación de los objetivos del problema principal. En este trabajo, se utilizará la formulación de Tchebychef:

donde es un vector peso y es un punto de referencia donde cada una de sus componentes está dada por:

Para dotar al algoritmo de manipulación de restricciones, se ha optado por el mecanismo basado en selección, el cual consiste en:

* Si un individuo de la generación y otro de la generación no incumplen las restricciones, la actualización se lleva a cabo de la misma forma que si se tratara de un problema sin restricciones.
* Si un individuo de la generación y otro de la generación incumplen las restricciones, la actualización se realiza sumando los errores cometidos por cada individuo en cada restricción y se selecciona el que tenga un error más pequeño (número más positivo).
* Si un individuo de la generación no cumple las restricciones, pero el individuo de la generación sí, entonces la actualización solo se produce en el propio individuo y en los vecinos que no incumplan las restricciones. La actualización total o parcial del vecindario de un individuo, causaría la perdida de diversidad.

# 2. Estructura del código y elementos utilizados

# 3. Utilización del programa y ejemplos de uso

- Funciones recursivas : algoritmo\_agregacion\_ZDT3, algoritmo\_agregacion\_restricciones

- Funciones por patrones : algoritmo\_agregacion\_ZDT3\_aux, algoritmo\_agregacion\_restricciones\_aux

- Usos de guardas : limitador\_aux, evalua\_cf6

- Usos de listas por comprensión : calc\_subproblemas, puntos\_de\_cruce

- Usos de orden superior : calc\_vecindario, cumple

- Usos de evaluación perezosa :

- Dos de tipos de datos abstractos : Random, Array y gnuplot

- Dos tipos de datos nuevos :

- Módulo : Zdt3 y CF6

# 4. Librerías utilizadas (extra indicar como instalarlas)