< >
MEMORIA ASC
COMPETICIÓN BLOQUE 3

28/10/2023

Autor:
Daniel Gallardo Martos, *dangalmar@alum.us.es*

Tabla de contenidos

1. Introducción

En este estudio, se investiga y compara un algoritmo multiobjetivo basado en agregación con el ampliamente utilizado NSGA-II (Non-dominated Sorting Genetic Algorithm II). El objetivo principal es evaluar la eficiencia y efectividad de ambos enfoques en la resolución de problemas de optimización multiobjetivo, además de analizar cómo la elección del algoritmo influye en la calidad de las soluciones obtenidas. Esta investigación tiene como propósito proporcionar una visión más clara de las ventajas y desventajas de los algoritmos de agregación en comparación con NSGA-II. La implementación se realiza en python y las gráficas por la librería matplotlib.

2. Contenido

2.1. Explicación de implementación

A continuación se explicarán cada una de las partes que se realizaron para, en conjunto, poder entender el porqué de las decisiones tomadas durante la implementación.

1. Inicialización
   En esta parte de la implementación que comienza con la inicialización de parámetros y termina con la creación de la generación inicial y su posterior evaluación se tomaron 2 decisiones que merece la pena comentar.
   La primera decisión se refiere a la generación de los vectores pesos. A la hora de generar los N vectores si N tiene un valor grande como 30 se pone de manifiesto el problema de coma flotante en la que se pierde precisión por lo que la condición de equidistancia para estos vectores no se cumpliría al acumularse esa diferencia a medida que se generan. Para solucionarlo simplemente redondeamos esa diferencia (a 12 decimales por ejemplo) y ya tendríamos vectores equidistantes.
   La segunda decisión se refiere a la generación de los vectores vecinos. Para agilizar el proceso se utilizó la librería Numpy para el cálculo de la distancia euclidiana y así disminuir el tiempo de ejecución. El resultado fue guardado en un diccionario con clave vector peso i, y como valor la lista de vectores vecinos B(i).

2. Reproducción
   En esta parte de la implementación que consiste en generar un nuevo individuo a partir de tres individuos vecinos se utilizaron los operadores evolutivos propuestos en la competición, es decir, el algoritmo de evolución diferencial con los parámetros recomendados CR = 0.5 y F = 0.5 y el operador de mutación gaussiana. Se utilizó Numpy tanto para la normalización al espacio de búsqueda como para la función de densidad de probabilidad de Gauss. Durante el análisis jugué con los valores de CR y F y llegué a la conclusión de que ante valores muy bajos o muy altos de CR la convergencia de soluciones al Frente de Pareto se ralentiza significativamente, como era de esperar. Además, si bajaba F había poca dispersión entre las soluciones no dominadas y una convergencia más lenta, debido a la baja exploración.

3. Actualización de vecinos
   Esta parte de la implementación va después de la evaluación de la nueva solución y la actualización de Z, dos partes de la implementación que no es necesario comentar debido a que se realizó según pone en la descripción de las funciones, a la simplicidad de las funciones y a que no se tomó ninguna decisión que se deba comentar.

   Para la actualización de vecinos se aplicó la formulación de Tchebycheff e intenté un cambio en la actualización que creía que podría mejorar el algoritmo pero terminó empeorando por lo que lo terminé cambiando. Debido a que podía haber varias actualizaciones de un mismo individuo, lo que podría derivar en menos diversidad de población, introduje un break después de la primera actualización, pero resultó empeorar el algoritmo debido a que reducía drásticamente la convergencia y no mejoraba tanto la diversidad de soluciones no dominadas.

4. Aplicación de restricciones
   A la hora de aplicar las restricciones de la función objetivo en cf6 probé tanto la penalización por penalty como la penalización por selección, ambas con sus ventajas e inconvenientes que pudieron observarse en la implementación y en la evaluación.

   En la penalización por penalty había una gran desventaja con respecto a la penalización por selección que se aprecia en la implementación y era el ajuste de pesos debido a que para que saliese una evaluación con resultados realmente buenos había que encontrar los pesos exactos lo que puede llegar a ser tedioso.

   En este caso no fue realmente complicado ya que el conjunto de valores que sacaban las funciones objetivos estaban en el rango de los valores que salían a la hora de evaluar las restricciones por lo que bastaba con sumar los valores absolutos de las restricciones en las funciones objetivos para que tuviese en cuenta las restricciones a la hora de minimizar sin dejar de tener en cuenta la función objetivo.

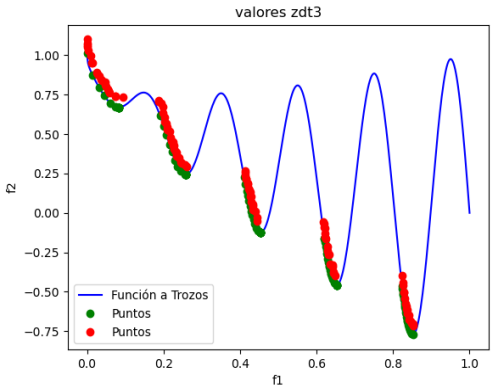
   La penalización por selección tiene la ventaja de que no hay que elegir pesos ya que se regula por sí sola al elegir la que incumpla menos restricciones y en caso de que ambas la cumplan se utiliza la formulación de Tchebycheff. Aunque tiene la desventaja de que suele funcionar peor que la de penalización por penalty si se eligen los pesos exactos.

2.2. Comparación y análisis función zdt3

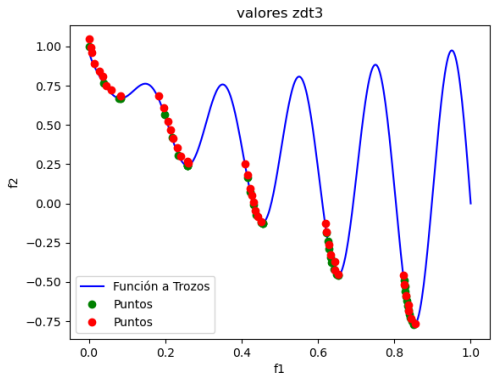
En las gráficas mostradas referentes a los Frentes de Pareto los puntos rojos son las soluciones del algoritmo NSGAII, mientras que los verdes son los de mi algoritmo implementado.

2.2.1. 10.000 evaluaciones

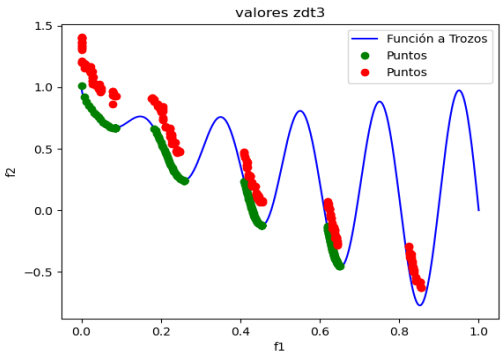
Dentro del marco de estas 10.000 evaluaciones se han realizado análisis y comparaciones con distinto número de subproblemas y generaciones. Se realizaron 10 ejecuciones de cada uno de los casos.
Se realizaron ejecuciones con 100 subproblemas y 100 generaciones; con 40 subproblemas y 250 generaciones; y con 200 subproblemas y 50 generaciones.



Gráfica que muestra el frente de Pareto con 100 subproblemas y 100 generaciones.
Tras hacer 10 ejecuciones la conclusión a la que se llega es que ambas funcionan bastante bien a la hora de conseguir un buen conjunto de soluciones no dominadas pero mi implementación converge ligeramente mejor.

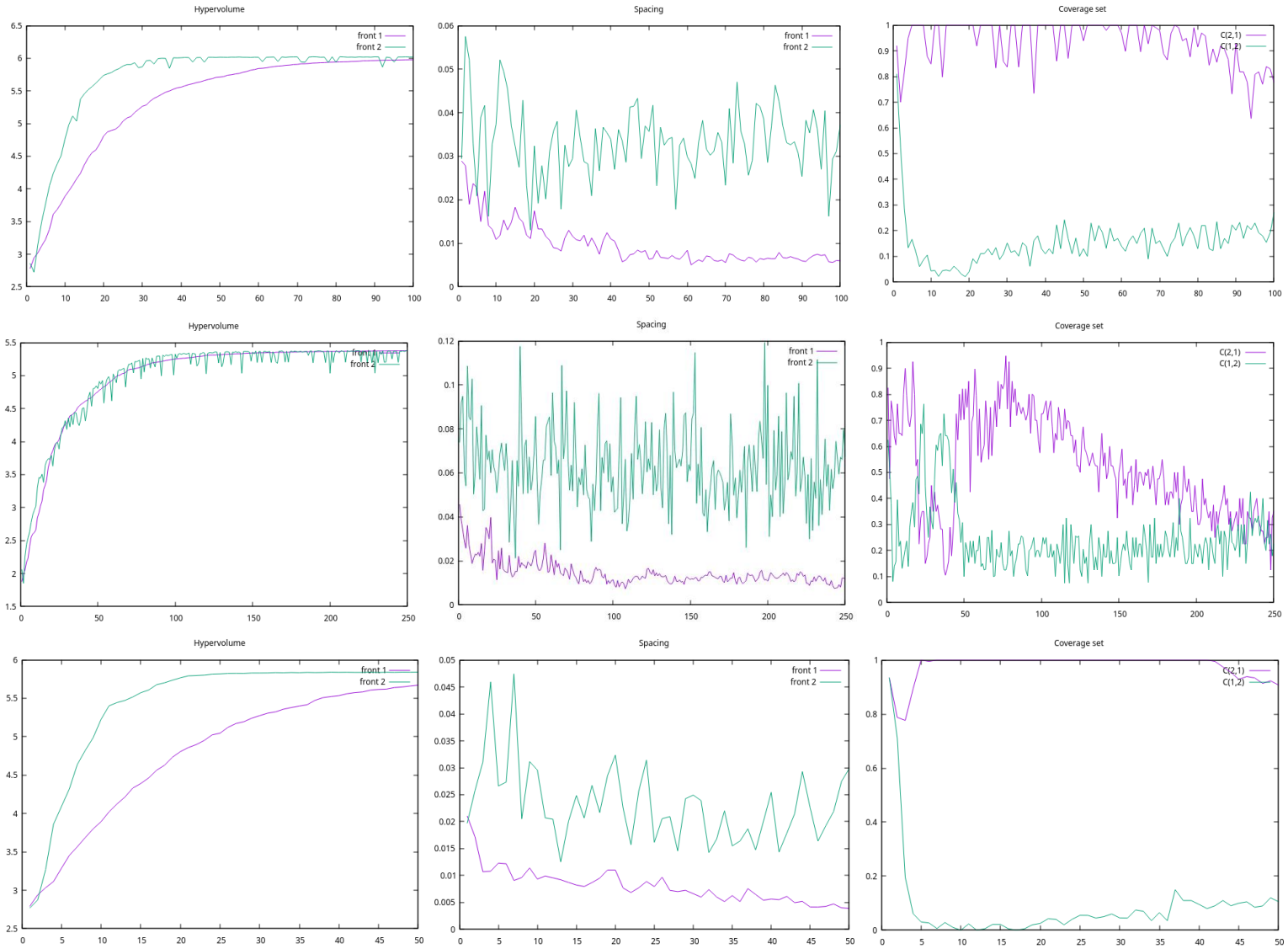


Gráfica que muestra el frente de Pareto con 40 subproblemas y 250 generaciones.
Tras hacer 10 ejecuciones la conclusión a la que se llega es que cuando se suben las generaciones ambos dan soluciones no dominadas igual de buenas. Cabe destacar que en algunas ejecuciones mi implementación no llega al máximo absoluto debido a que el algoritmo no ha explorado tanto por el medio de búsqueda como NSGAII.



Gráfica que muestra el frente de Pareto con 200 subproblemas y 50 generaciones.
Tras hacer 10 ejecuciones y analizar los resultados reforzamos lo concluído anteriormente, es decir, mi implementación converge mucho más rápido que NSGAII y se aprecia más con pocas generaciones, pero explora menos por lo que con pocos subproblemas no puede llegar a todos los mínimos.

A continuación se muestran las gráficas referentes al hipervolumen, el espaciado y la cobertura de las soluciones no dominadas.Las funciones moradas hacen referencia a las del algoritmo NSGAII mientras que las funciones verdes serían las de mi implementación.



Las tres primeras hacen referencia a 100 subproblemas y 100 generaciones, la segunda a 40 subproblemas y 250 generaciones, y la tercera a 200 subproblemas y 50 generaciones. Las gráficas mostradas son las gráficas de las ejecuciones que daban los resultados que se acercaban más a la media de los resultados obtenidos de las 10 ejecuciones evaluadas.

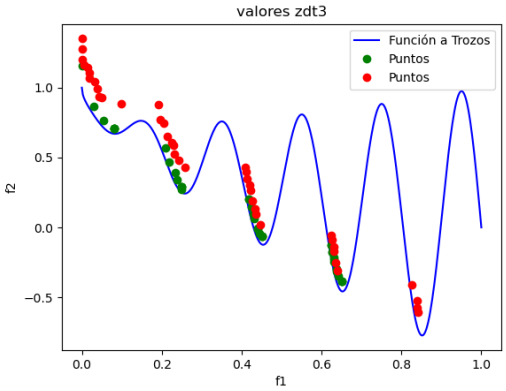
Analizando el hipervolumen en las gráficas podemos observar una clara diferencia en la forma de las funciones mientras crecen a lo largo de las generaciones. Resulta que el algoritmo que yo implementé presenta muchos picos y valles cuando se tiene un número de subproblemas bajo. Los picos y valles pueden indicar que el algoritmo genera soluciones que divergen al Frente de Pareto. Esto podría explicar también la diferencia de spacing entre las gráficas.

En cuanto a la cobertura vemos que el algoritmo implementado por mi tiene mejores soluciones cuando el número de generaciones es pequeño. Esto tiene todo el sentido del mundo debido a que ya habíamos visto como mi algoritmo converge más rápido aunque no

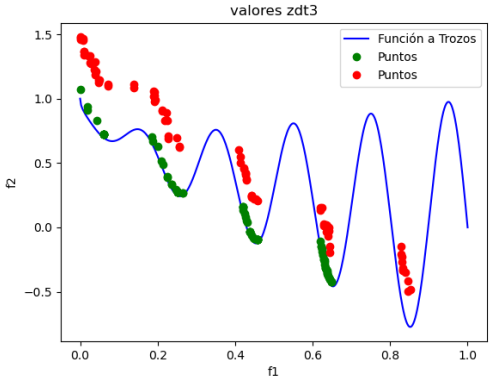
llegue a todos los mínimos cuando el número de subproblemas es bajo. En el momento en el que el número de generaciones es suficientemente grande las coberturas se estabilizan como cabría esperar.

2.2.2. 4.000 evaluaciones

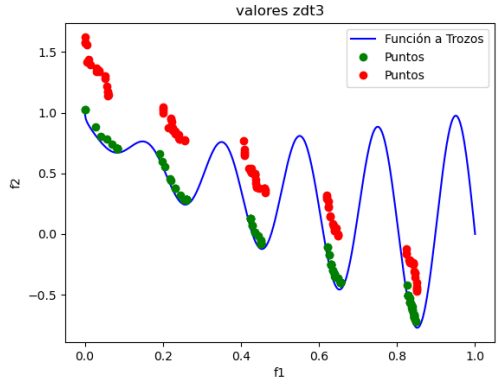
Dentro del marco de estas 4.000 evaluaciones se realizaron ejecuciones con 40 subproblemas y 100 generaciones; con 80 subproblemas y 50 generaciones; y con 100 subproblemas y 40 generaciones.Tras hacer 10 ejecuciones de cada tipo la conclusión a la que llegamos es la misma a la que se llegó con los Frentes de Pareto de las 10.000 ejecuciones.



Gráfica que muestra el frente de Pareto con 40 subproblemas y 100 generaciones.

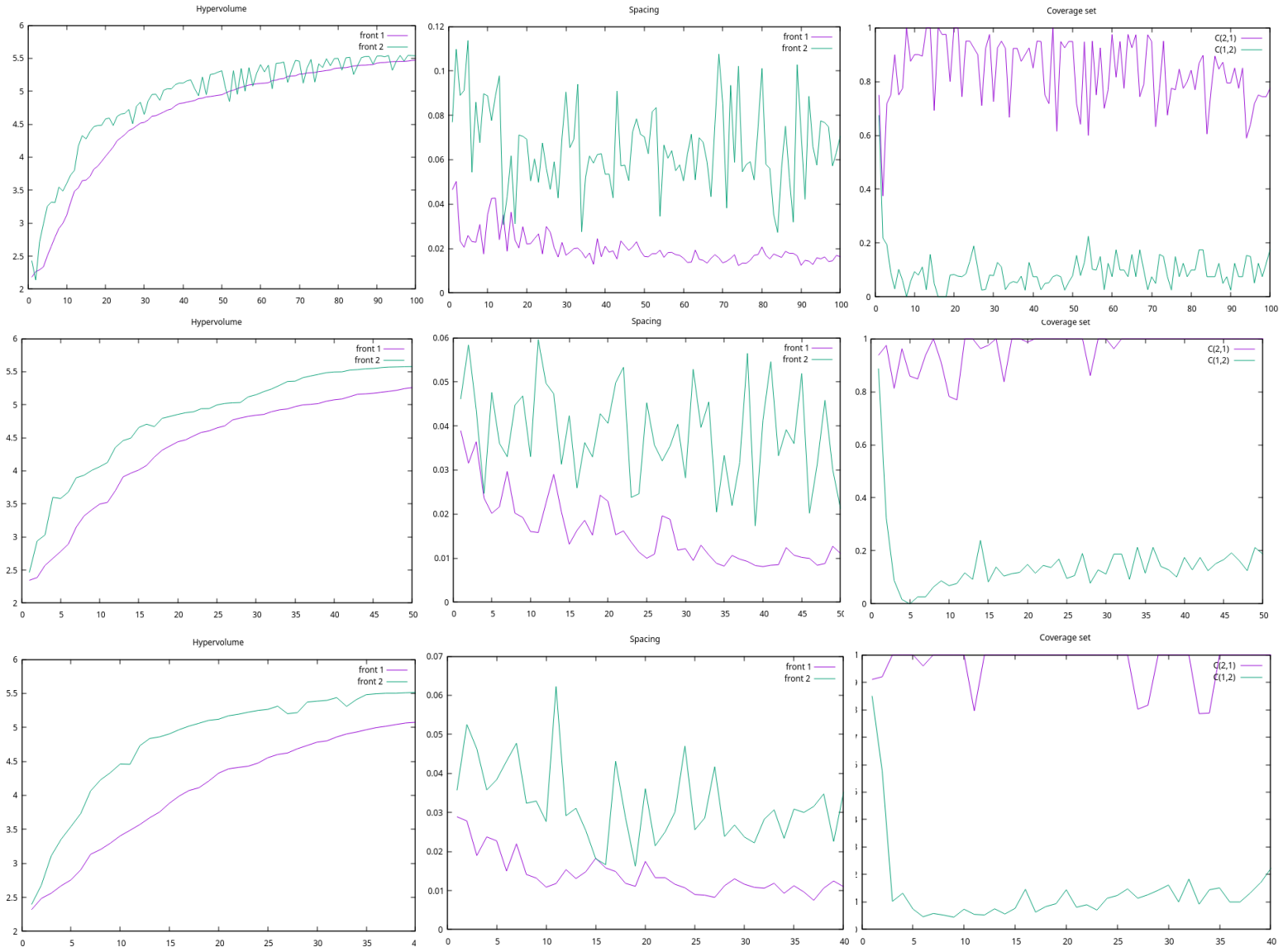


Gráfica que muestra el frente de Pareto con 80 subproblemas y 50 generaciones.



Gráfica que muestra el frente de Pareto con 100 subproblemas y 40 generaciones.

A continuación se muestran las gráficas referentes al hipervolumen, el espaciado y la cobertura de las soluciones no dominadas.



Las tres primeras hacen referencia a 40 subproblemas y 100 generaciones, la segunda a 80 subproblemas y 50 generaciones, y la tercera a 100 subproblemas y 40 generaciones. Las gráficas mostradas son las gráficas de las ejecuciones que daban los resultados que se acercaban más a la media de los resultados obtenidos de las 10 ejecuciones evaluadas.

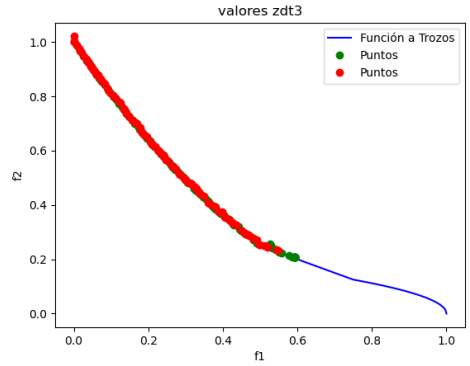
Analizando tanto el hipervolumen como el espaciado y la cobertura llegamos a las mismas conclusiones que con las métricas de las 10.000 ejecuciones. Además, al haber menos ejecuciones queda de manifiesto de una forma más clara la diferencia en la convergencia en el hipervolumen y la cobertura.

2.3. Comparación y análisis función cf6 con 4 dimensiones

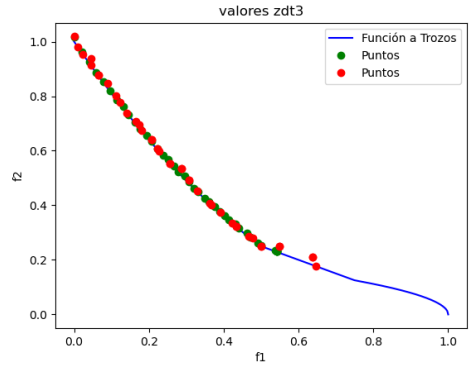
Como los resultados fueron muy similares, analizaré las gráficas con la penalización por selección

2.3.1. 10.000 evaluaciones

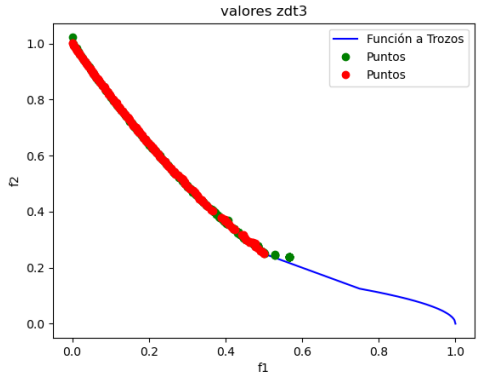
Dentro del marco de estas 10.000 evaluaciones se han realizado análisis y comparaciones con los mismos casos que la función zdt3.



Gráfica que muestra el frente de Pareto con 100 subproblemas y 100 generaciones.
Tras hacer 10 ejecuciones la conclusión a la que se llega es que ambas funcionan bastante bien a la hora de conseguir un buen conjunto de soluciones no dominadas aunque ninguna de ellas cubre la porción de Frente que va desde 0.6 a 1 con satisfacción.

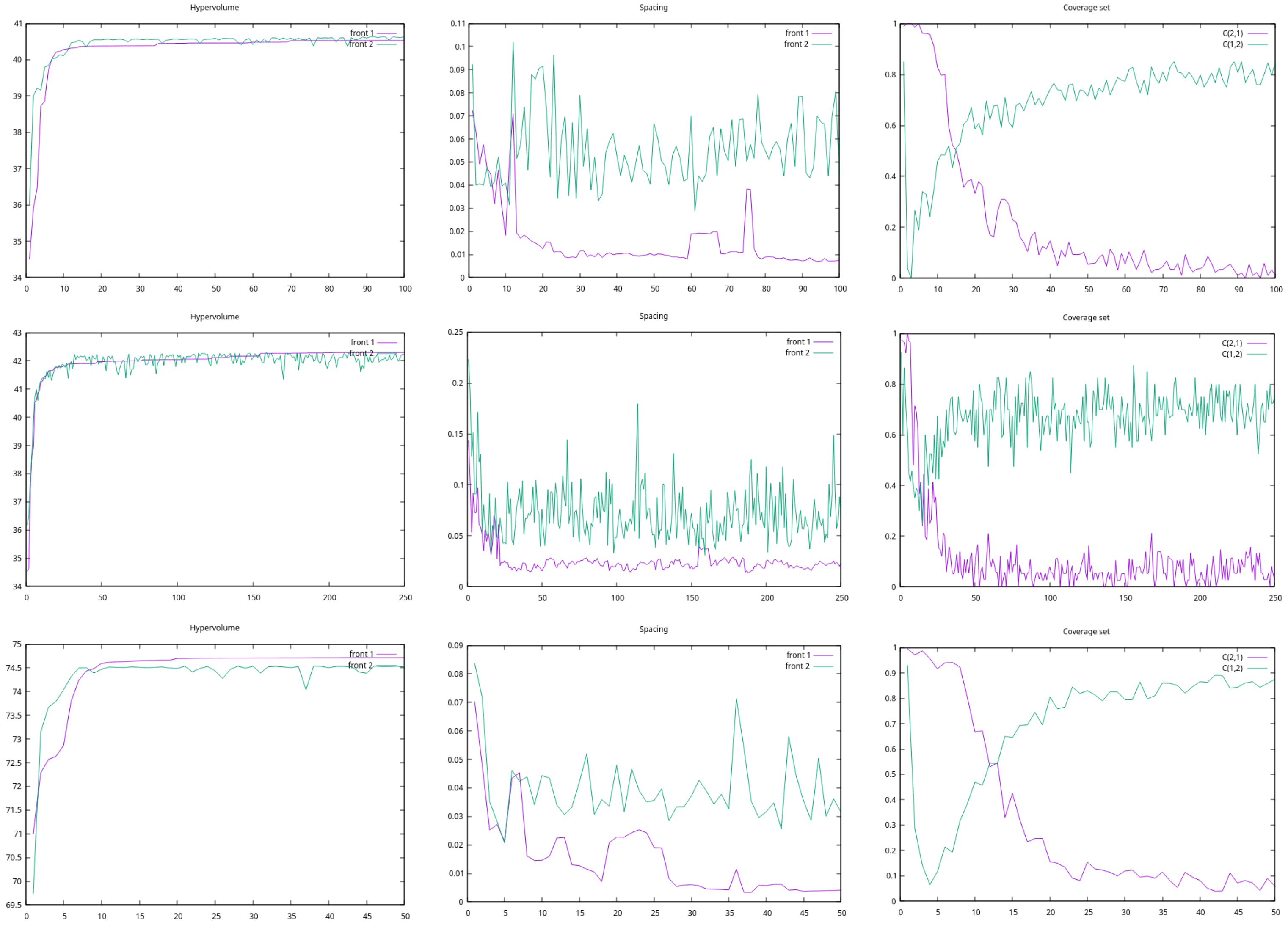


Gráfica que muestra el frente de Pareto con 40 subproblemas y 250 generaciones.
Tras hacer 10 ejecuciones la conclusión a la que se llega es la misma que la del caso anterior.



Gráfica que muestra el frente de Pareto con 200 subproblemas y 50 generaciones.
Tras hacer 10 ejecuciones la conclusión a la que se llega es la misma que en los casos anteriores.

A continuación se muestran las gráficas referentes al hipervolumen, el espaciado y la cobertura de las soluciones no dominadas.



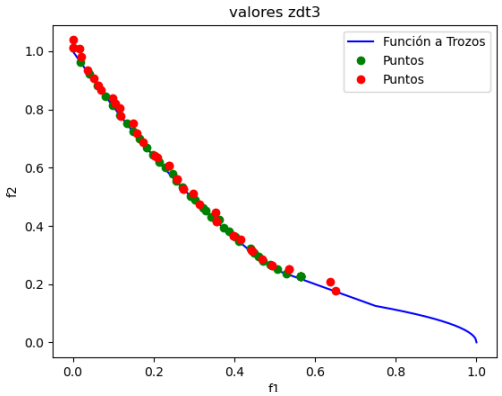
Las tres primeras hacen referencia a 100 subproblemas y 100 generaciones, la segunda a 40 subproblemas y 250 generaciones, y la tercera a 200 subproblemas y 50 generaciones. Las gráficas mostradas son las gráficas de las ejecuciones que daban los resultados que se acercaban más a la media de los resultados obtenidos de las 10 ejecuciones evaluadas.

Analizando el hipervolumen en las gráficas podemos observar que la convergencia es prácticamente igual, lo único en lo que difieren son los valles y picos como ya se vio en el análisis de los resultados de zdt3. El spacing también se mantiene igual.

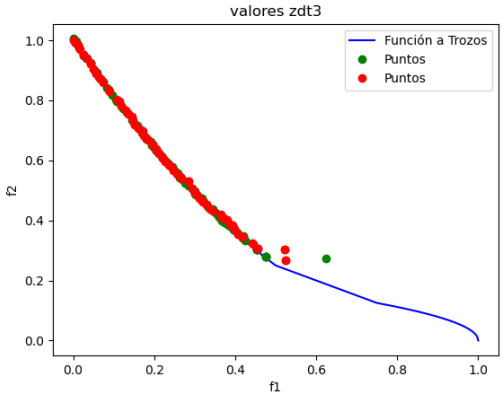
En cuanto a la cobertura sí que vemos una importante diferencia y es que hay un gran número de soluciones del algoritmo de NSGAII que dominan a las de mí implementación, algo que al analizar el Frente de Pareto no se apreciaba muy bien ya que esa diferencia de dominancia es muy pequeña.

2.3.2. 4.000 evaluaciones

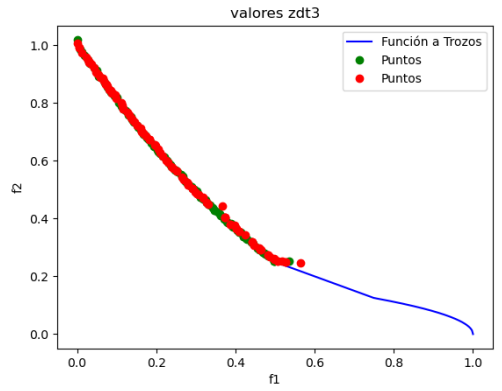
Dentro del marco de estas 4.000 evaluaciones se realizaron ejecuciones con 40 subproblemas y 100 generaciones; con 80 subproblemas y 50 generaciones; y con 100 subproblemas y 40 generaciones.Tras hacer 10 ejecuciones de cada tipo la conclusión a la que llegamos es la misma a la que se llegó con los Frentes de Pareto de las 10.000 ejecuciones.



Gráfica que muestra el frente de Pareto con 40 subproblemas y 100 generaciones.

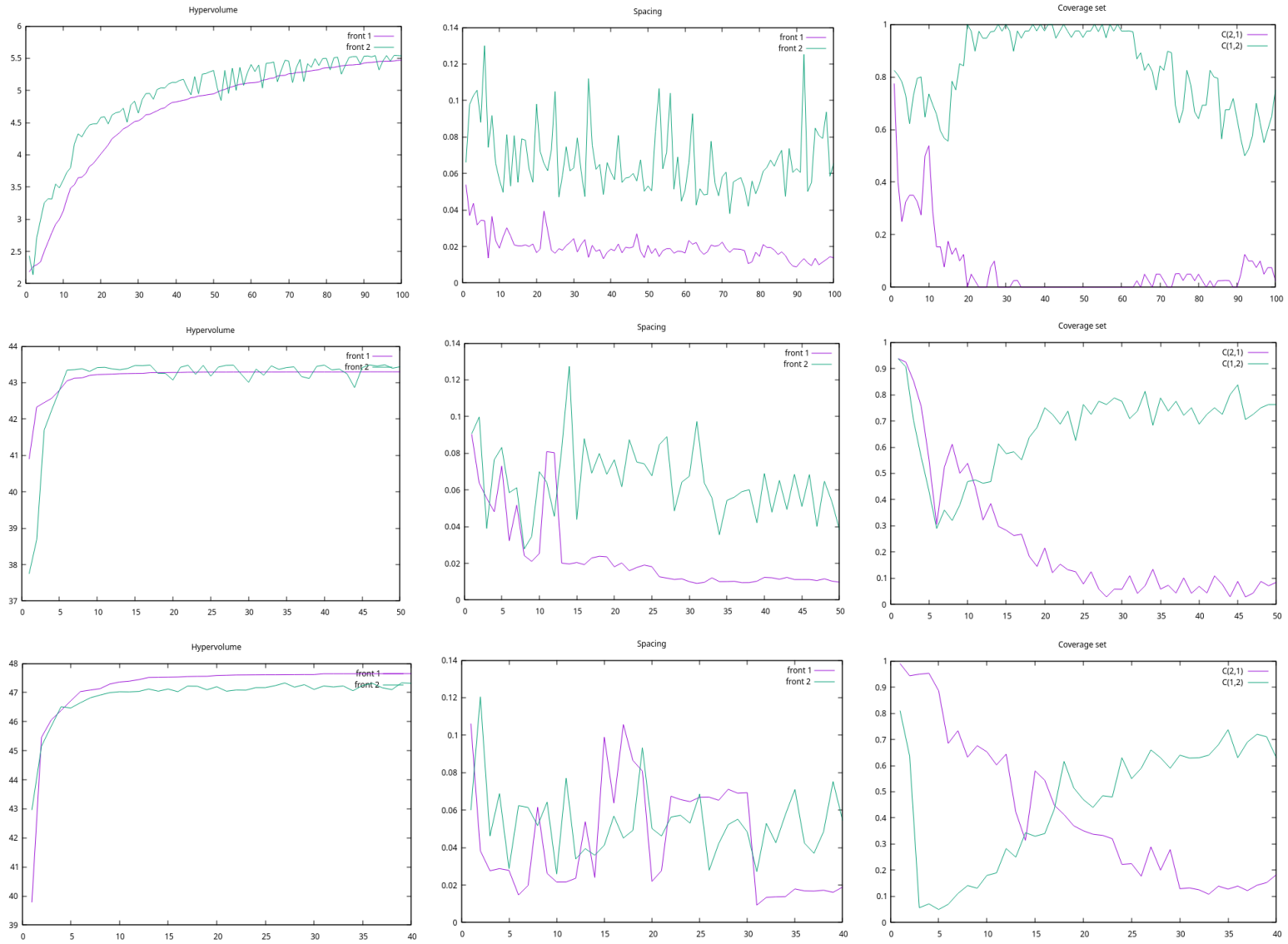


Gráfica que muestra el frente de Pareto con 80 subproblemas y 50 generaciones.



Gráfica que muestra el frente de Pareto con 100 subproblemas y 40 generaciones.

A continuación se muestran las gráficas referentes al hipervolumen, el espaciado y la cobertura de las soluciones no dominadas.



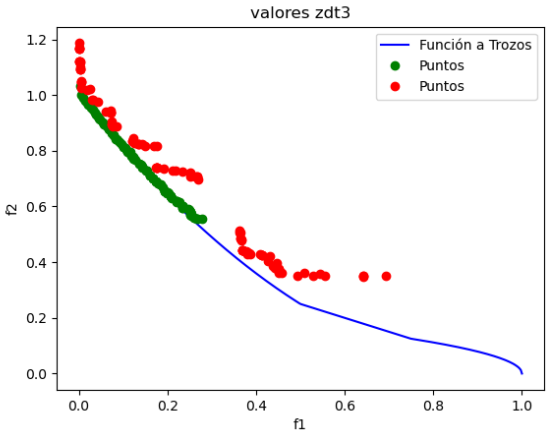
Las tres primeras hacen referencia a 40 subproblemas y 100 generaciones, la segunda a 80 subproblemas y 50 generaciones, y la tercera a 100 subproblemas y 40 generaciones. Las gráficas mostradas son las gráficas de las ejecuciones que daban los resultados que se acercaban más a la media de los resultados obtenidos de las 10 ejecuciones evaluadas.

Analizando tanto el hipervolumen como el espaciado y la cobertura llegamos a las mismas conclusiones que con las métricas de las 10.000 ejecuciones.

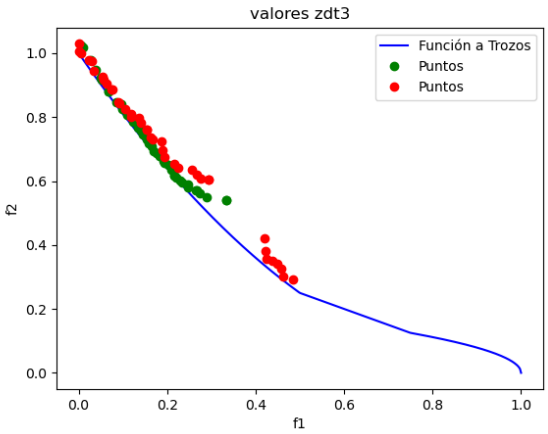
2.4. Comparación y análisis función cf6 con 16 dimensiones

2.4.1. 10.000 evaluaciones

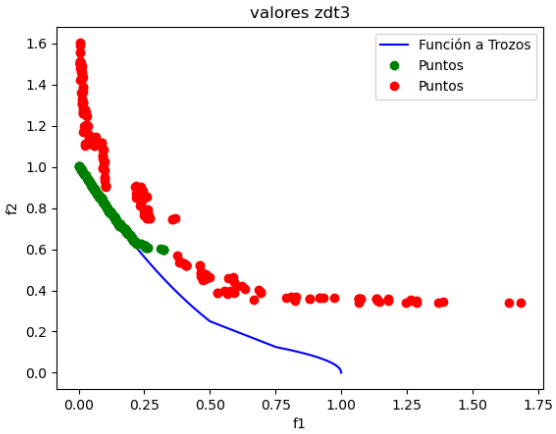
Como los resultados fueron muy similares, analizaré las gráficas con la penalización por selección.



Gráfica que muestra el frente de Pareto con 100 subproblemas y 100 generaciones. Tras hacer 10 ejecuciones la conclusión a la que se llega es que mi implementación converge mejor aunque se concentra demasiado en un trozo del frente óptimo.

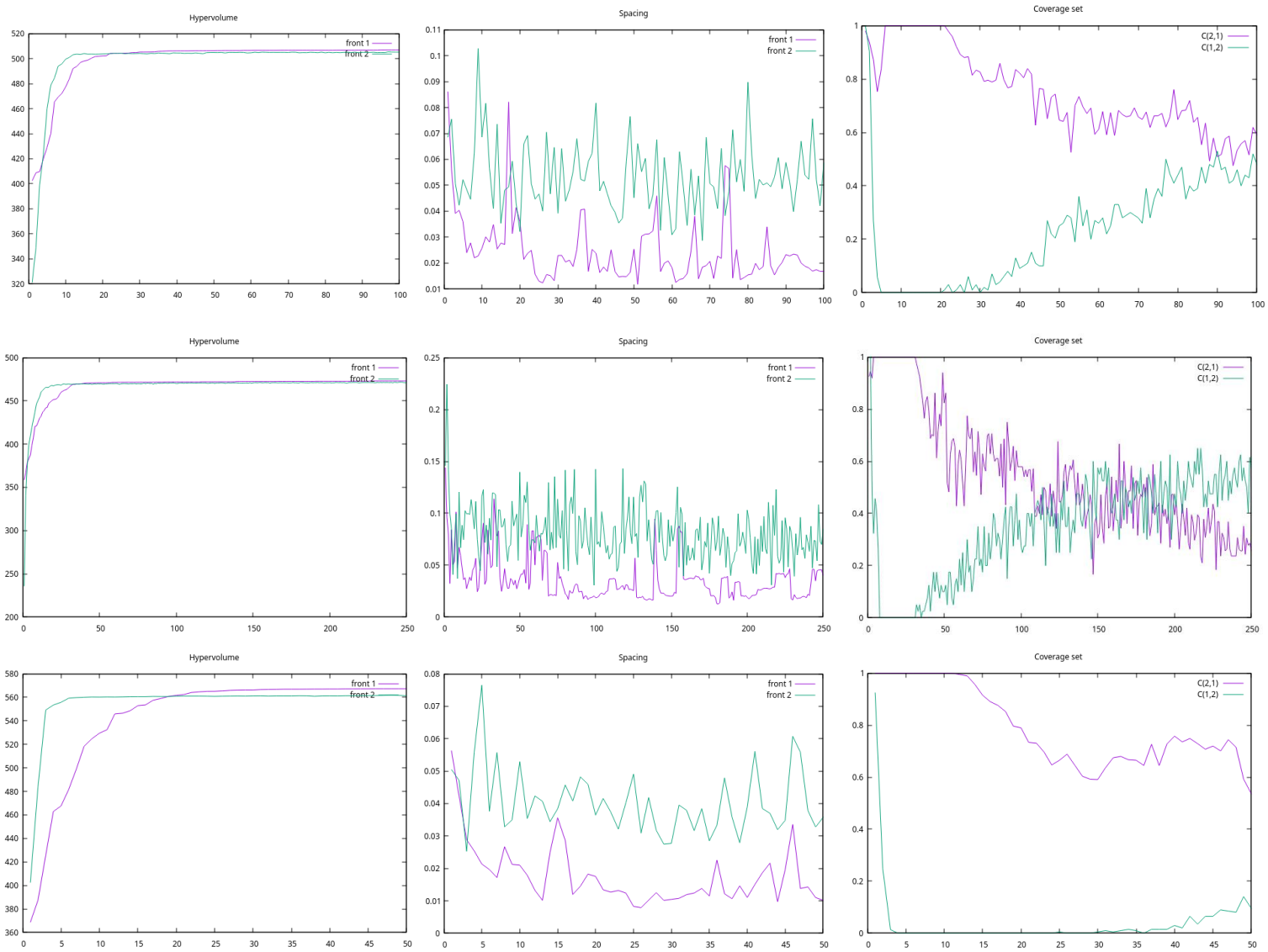


Gráfica que muestra el frente de Pareto con 40 subproblemas y 250 generaciones. Tras hacer 10 ejecuciones la conclusión a la que se llega es que al aumentar las generaciones el algoritmo NSGAII converge mejor.



Gráfica que muestra el frente de Pareto con 200 subproblemas y 50 generaciones. Tras hacer 10 ejecuciones y analizar los resultados reforzamos lo concluído anteriormente, es decir, mi implementación converge mucho mejor aunque se concentre en una región óptima pequeña y el algoritmo NSGAII necesita más generaciones para hacerlo. Además, el algoritmo NSGAII cubre más espacio aunque no converja tanto.

A continuación se muestran las gráficas referentes al hipervolumen, el espaciado y la cobertura de las soluciones no dominadas.



Las tres primeras hacen referencia a 100 subproblemas y 100 generaciones, la segunda a 40 subproblemas y 250 generaciones, y la tercera a 200 subproblemas y 50 generaciones. Las gráficas mostradas son las gráficas de las ejecuciones que daban los resultados que se acercaban más a la media de los resultados obtenidos de las 10 ejecuciones evaluadas.

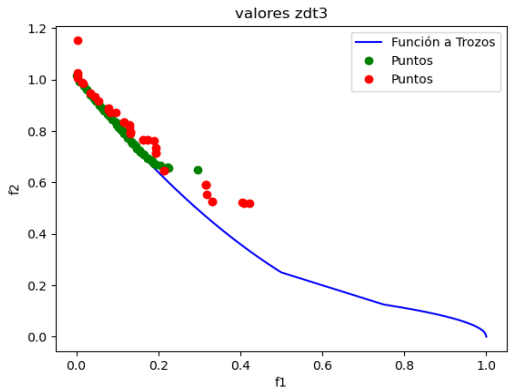
Analizando el hipervolumen en las gráficas podemos observar que la convergencia de mi implementación es mayor, como era de esperar, pero el algoritmo NSGAII ofrece una mayor diversidad de soluciones no dominadas que cubren un mayor Frente de Pareto lo que hace que se equiparen los hipervolumenes.

En cuanto a la cobertura vemos que debido a la gran convergencia de mi implementación, esta empieza teniendo una mayor cobertura, pero a medida que pasan las generaciones las cobertura va cambiando hasta que se tornan los valores debido a lo antes

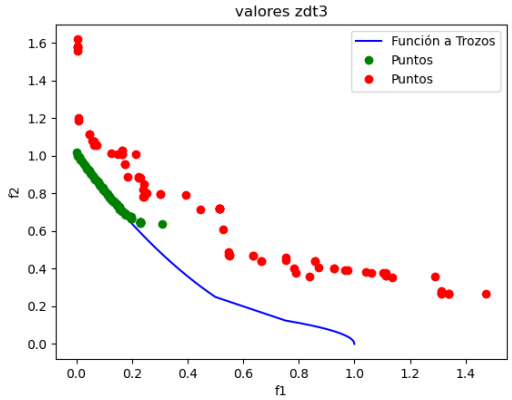
mencionado. Si el número de subproblemas inicial aumenta el cambio de cobertura es más grande como era de esperar.

2.4.2. 4.000 evaluaciones

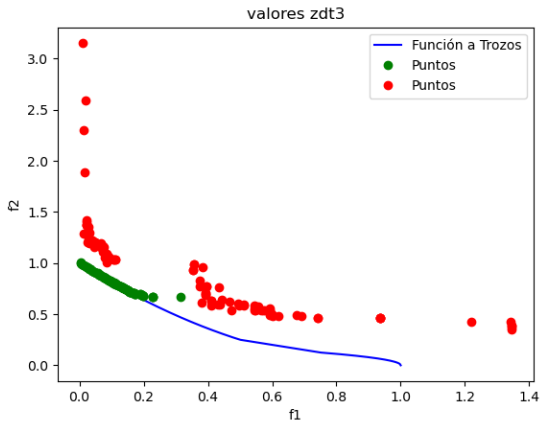
Dentro del marco de estas 4.000 evaluaciones se realizaron ejecuciones con 40 subproblemas y 100 generaciones; con 80 subproblemas y 50 generaciones; y con 100 subproblemas y 40 generaciones. Tras hacer 10 ejecuciones de cada tipo la conclusión a la que llegamos es la misma a la que se llegó con los Frentes de Pareto de las 10.000 ejecuciones.



Gráfica que muestra el frente de Pareto con 40 subproblemas y 100 generaciones.

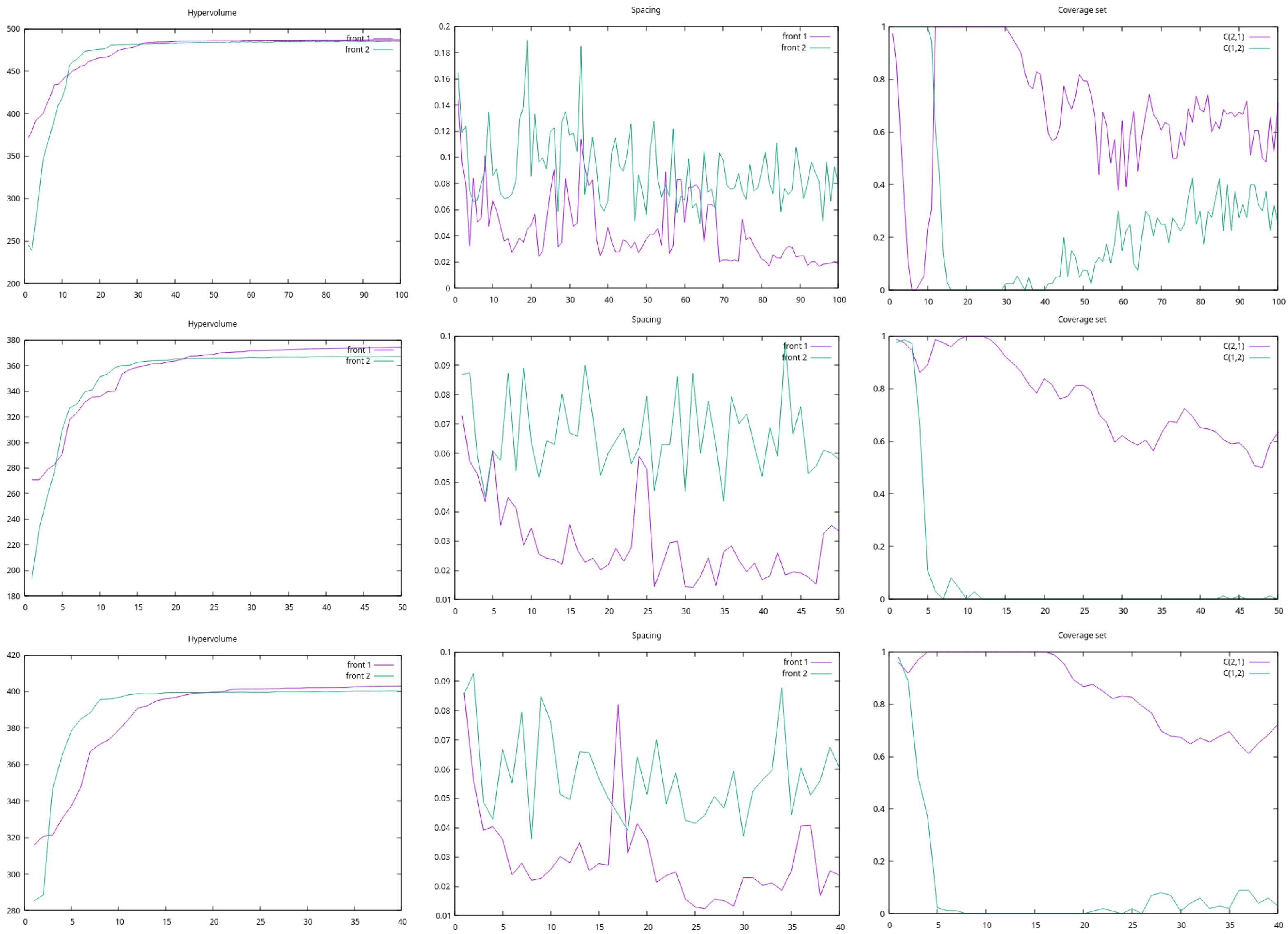


Gráfica que muestra el frente de Pareto con 80 subproblemas y 50 generaciones.



Gráfica que muestra el frente de Pareto con 100 subproblemas y 40 generaciones.

A continuación se muestran las gráficas referentes al hipervolumen, el espaciado y la cobertura de las soluciones no dominadas.



Las tres primeras hacen referencia a 40 subproblemas y 100 generaciones, la segunda a 80 subproblemas y 50 generaciones, y la tercera a 100 subproblemas y 40 generaciones. Las gráficas mostradas son las gráficas de las ejecuciones que daban los resultados que se acercaban más a la media de los resultados obtenidos de las 10 ejecuciones evaluadas.

Analizando tanto el hipervolumen como el espaciado y la cobertura llegamos a las mismas conclusiones que con las métricas de las 10.000 ejecuciones.

3. Conclusiones

A partir de los resultados obtenidos, se puede concluir que en problemas de optimización multiobjetivo, mi algoritmo basado en agregación demuestra una notable ventaja en términos de convergencia eficiente, especialmente en escenarios con un número limitado de generaciones o subproblemas. Sin embargo, muestra una exploración más restringida del espacio de soluciones no dominadas. Por otro lado, NSGA-II, aunque requiere más generaciones para alcanzar la convergencia, destaca por su capacidad para generar una mayor diversidad de soluciones no dominadas, lo que resulta en una mejor cobertura del espacio de Pareto. La elección entre estos enfoques debe basarse en las necesidades específicas del problema de optimización, ya que ambos tienen sus propias ventajas y desventajas.