

BLOQUE III. COMPUTACIÓN EVOLUTIVA

MEMORIA

1. Descripción de la implementación

En este proyecto se ha realizado una implementación básica del algoritmo evolutivo multiobjetivo basado en agregación para resolver dos problemas de minimización llamados ZDT3 Y CF6 en distintas dimensiones. En todo momento se han seguido las recomendaciones de implementación del profesor. El algoritmo se basa en transformar el problema multiobjetivo en un problema monobjetivo cuya función de coste es la suma de las funciones de coste del problema original ponderadas.

El lenguaje escogido para el desarrollo ha sido python debido a mi amplia experiencia utilizándolo. El código se distribuye en una carpeta src donde se implementa una clase para cada problema, en el que se desarrollan las funciones a optimizar, un archivo para el desarrollo de las funciones de reproducción y otro para la implementación del algoritmo principal. Luego tenemos otros archivos que son utilizados para automatizar la obtención de resultados, ejecuciones, gráficas y métricas de comparación con el algoritmo NSGAII. Además de incluir en la entrega todo el código, también se encuentra en un repositorio de github cuya dirección es <https://github.com/MarianodelRio/evolutionaryComputation>

Primero, se inicializa el algoritmo creando una población aleatoria siguiendo una distribución uniforme en el espacio de búsqueda de cada problema, cada individuo se le asocia un vector de peso (que será el que pondre la función objetivo), los cuales han sido creados equiespaciados para ocupar todas las posibles ponderaciones del problema. Entonces, se crea un vecindario, es decir, se asocia a cada vector de peso los T vectores más cercanos y un vector de referencia que será el mínimo valor encontrado de cada función del problema.

Tras ello, comienza el proceso iterativo. En cada paso, se reproduce la población con los mecanismos de cruce y mutación explicados en el enunciado del trabajo. Para cada individuo, se aplica el cruce de evolución diferencial y al nuevo individuo generado se le aplica la mutación gaussiana. El método de crossover básico ha sido implementado y probado en lugar del de evolución diferencial pero no se consigue apenas diversidad, por eso se ha elegido el que se recomienda. Tras ello, evaluamos los nuevos individuos, actualizamos el vector de referencia y el vecindario de cada individuo tal y como se explica en el enunciado. Los parámetros tienen como valor los recomendados, salvo el número de vecinos en CF6 que se mantiene a 4 para distintas poblaciones debido a que el algoritmo muestra un mejor comportamiento que aumentando este número.

En cuanto a la resolución del problema con restricciones, se ha optado por añadir una penalización a la función monobjetivo, que es la suma de los valores absolutos de las restricciones. todo multiplicado por un factor de penalización. Para una generalización del algoritmo, en el caso de no tener restricciones como el problema zdt3, añadimos 0.0 en la columna de restricciones, lo que hace que aunque haya un factor positivo, no se tenga penalización. El segundo método recomendado no se ha implementado debido a que considero que es equivalente al de penalizar con un factor de penalización, lo cual si ha sido probado haciendo ejecuciones variando este parámetro. Se ha encontrado que un factor de 30 en CF6 de 4 dimensiones y un factor de 10 en CF6 de 10 dimensiones muestra un buen comportamiento.

Por último, para visualizar el proceso, se ha desarrollado una función que va iterando y a la vez dibuja los individuos en cada momento junto con el frente ideal, lo cual nos ayuda a ver de forma intuitiva (al ser estocástico no podemos asegurarla en pocas ejecuciones) si nuestro algoritmo va a converger o hay algo que no funciona.

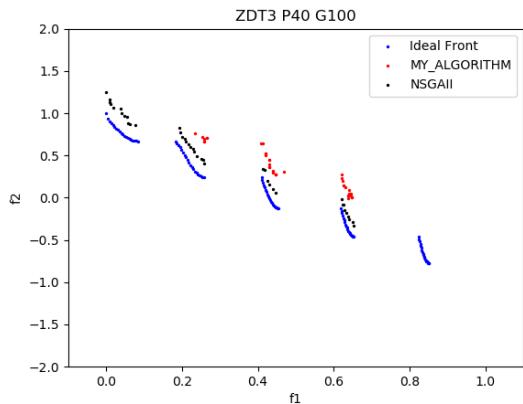
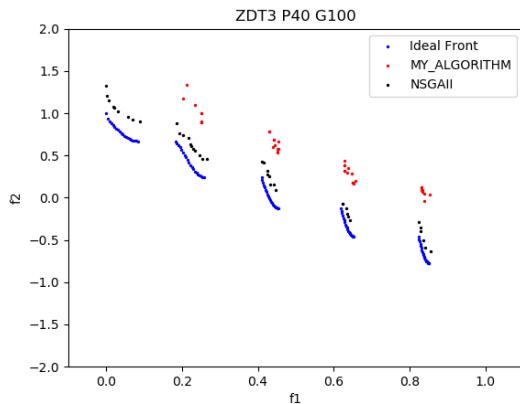
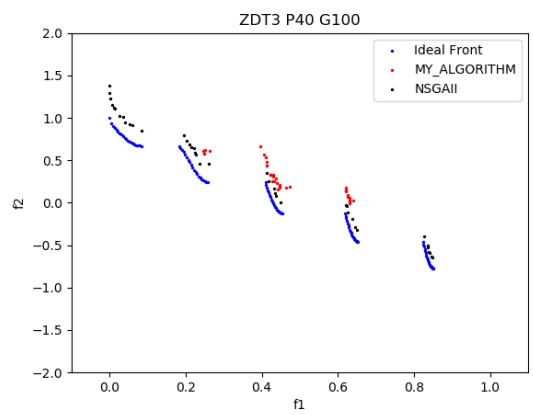
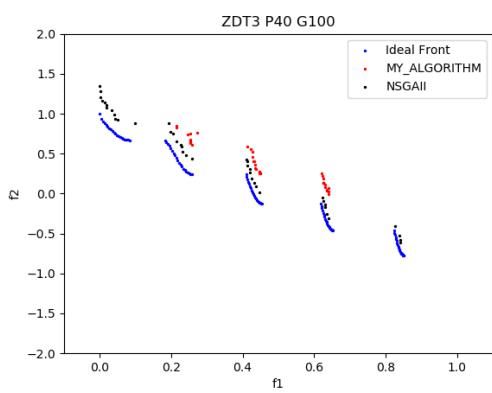
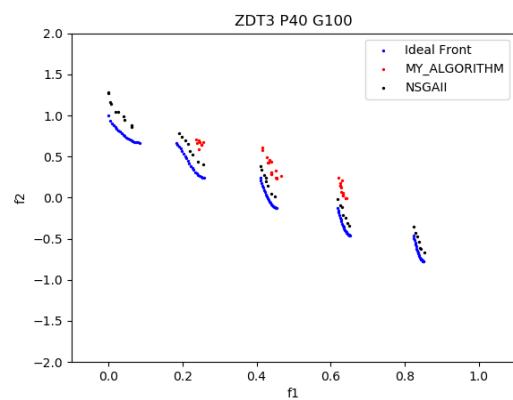
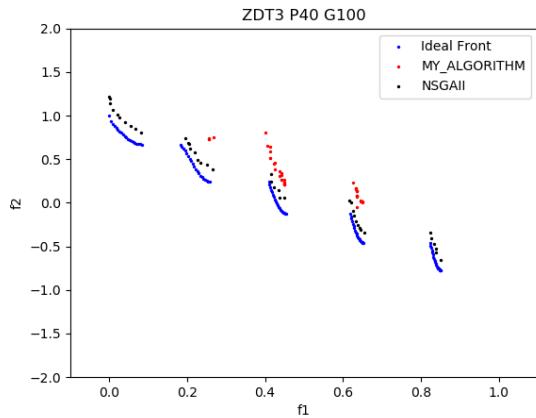
2. Resultados comparativos

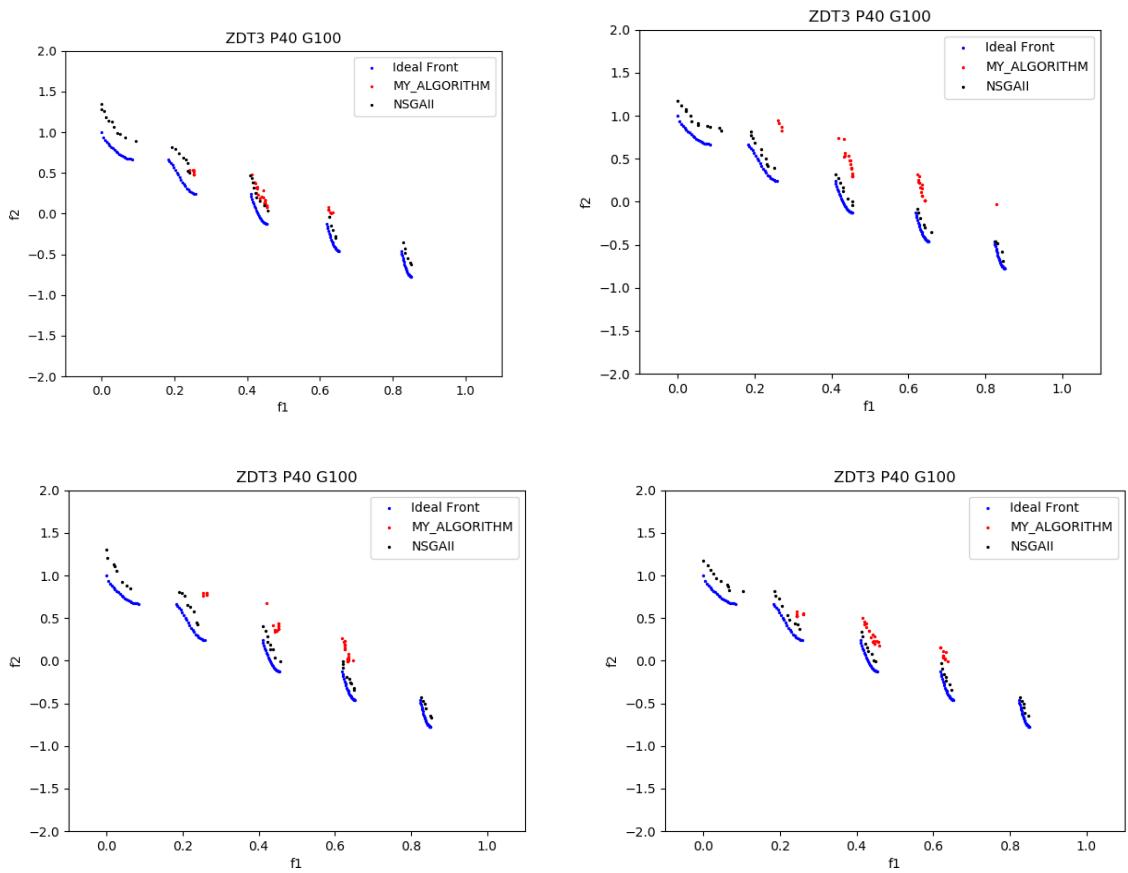
En este nuevo apartado se hará una comparación detallada de las soluciones obtenidas por cada algoritmo (agregación y NSGAII) en cada uno de los problemas nombrados incluyendo métricas de comparación (en concreto spacing y hipervolumen)

Atendiendo al número de evaluaciones, se han escogido las mismas combinaciones de población y generaciones que el profesor escoge para el algoritmo NSGAII para poder hacer una comparativa justa. Se ha optado por realizar 10 ejecuciones con semillas distintas para cada combinación para poder sacar conclusiones reales debido a que el algoritmo es estocástico. Con los datos obtenidos se calculan las métricas para cada ejecución y se representan todas en una gráfica comparativa de ambos algoritmos, esto para cada métrica.

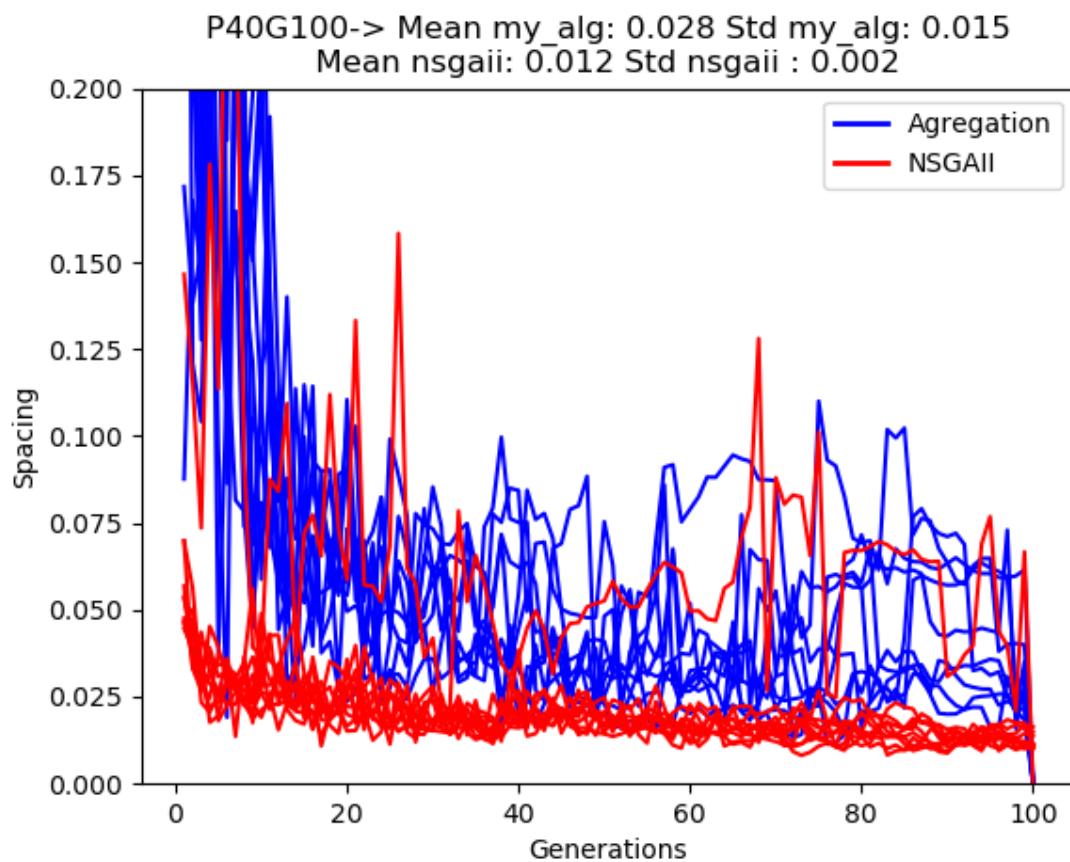
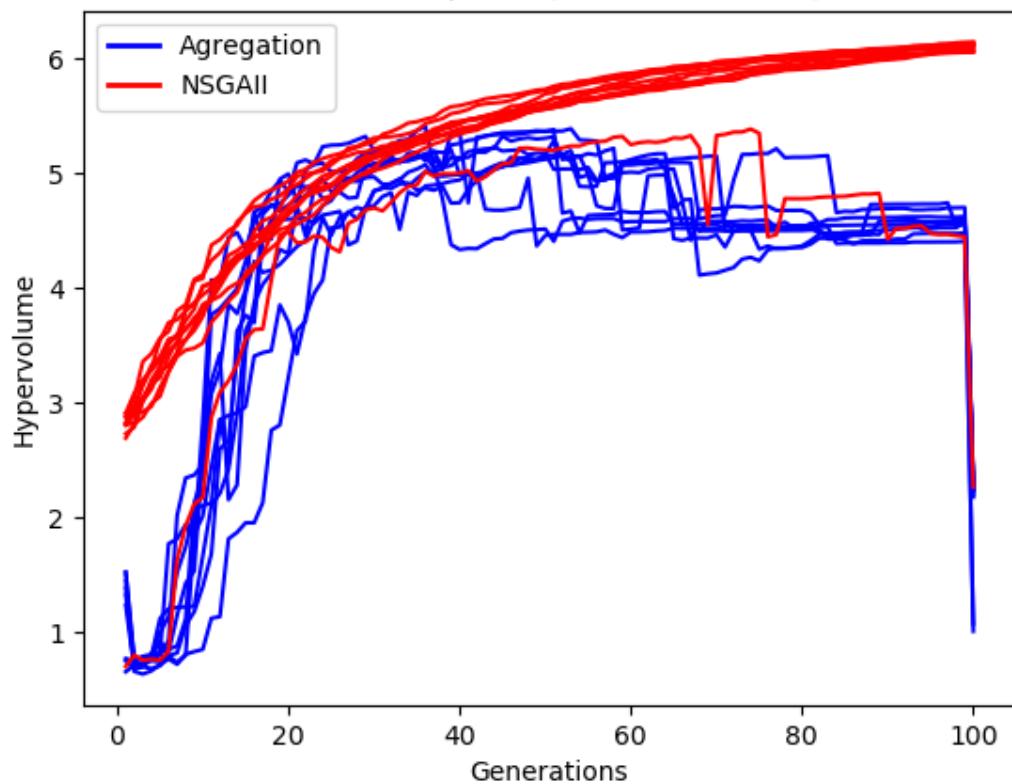
Problema ZDT3 en 30 dimensiones

40 individuos y 100 generaciones

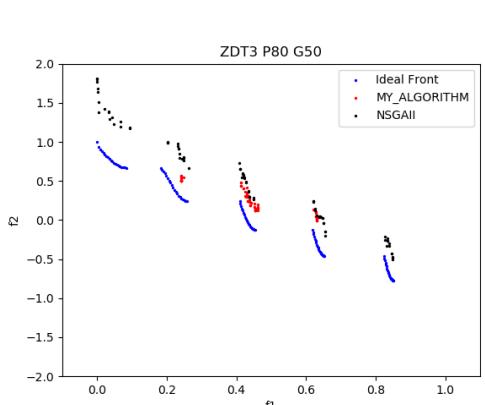
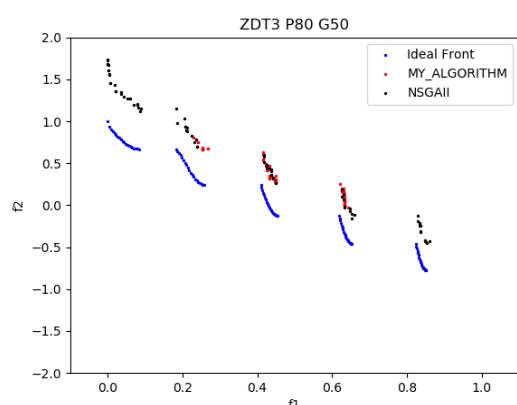
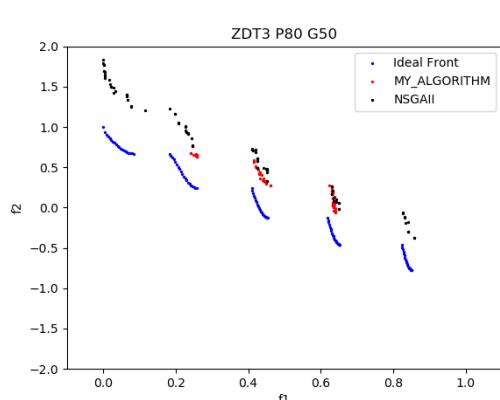
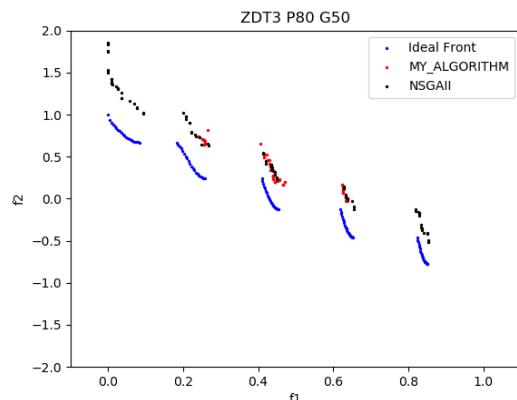
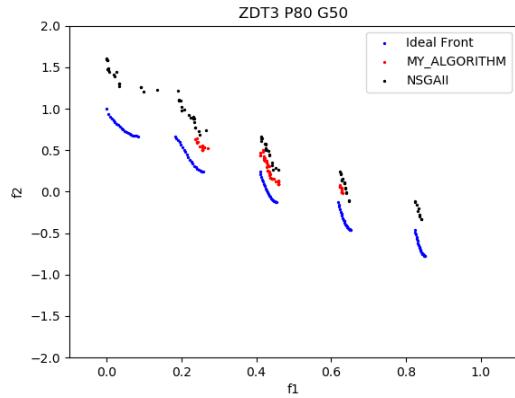
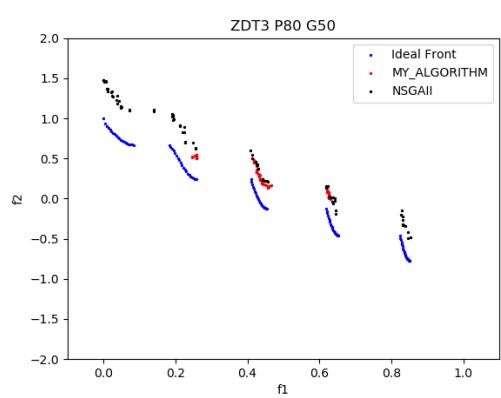


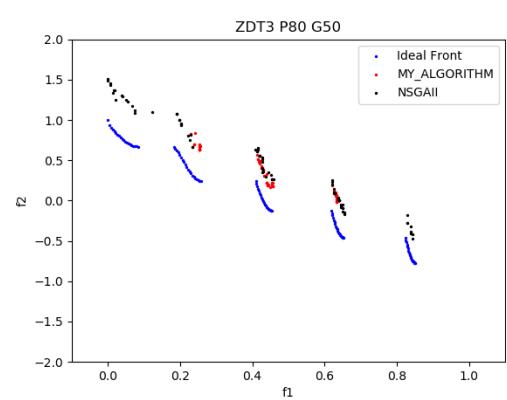
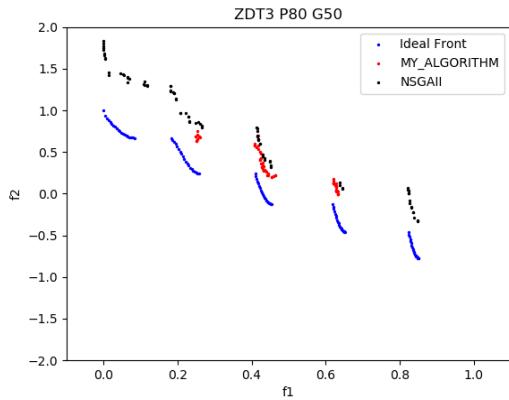
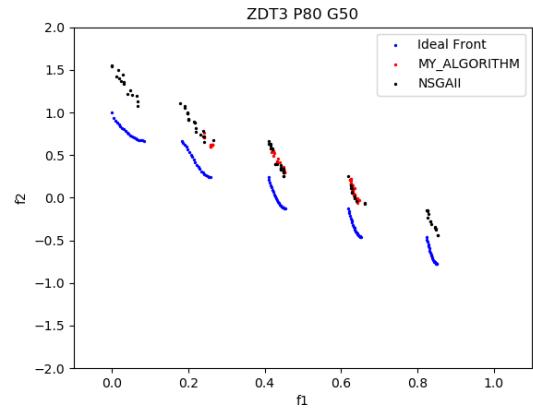
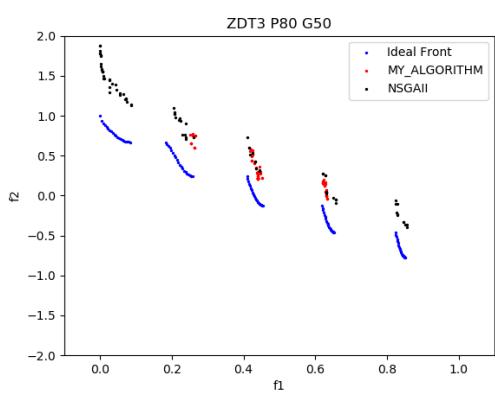


P40G100-> Mean my_alg: 1.781 Std my_alg: 0.760
Mean nsgaii: 6.105 Std nsgaii : 0.029
Reference point: [1. 6.232157]

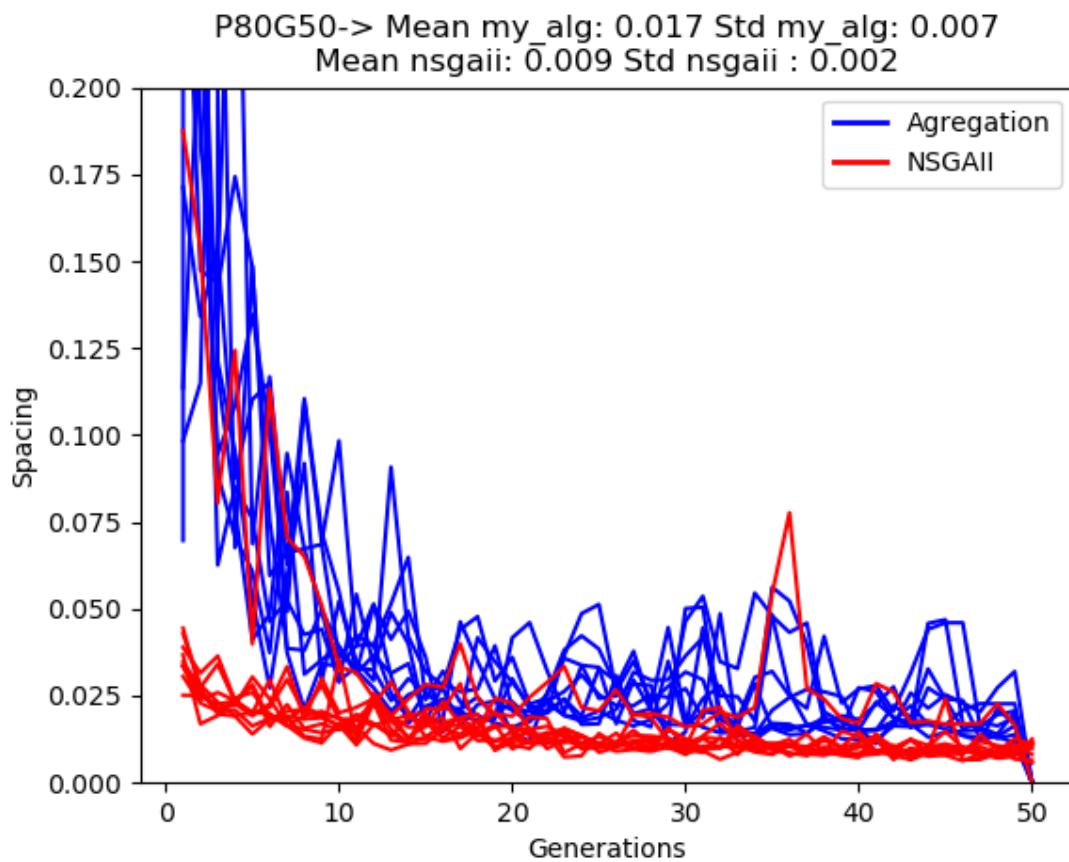
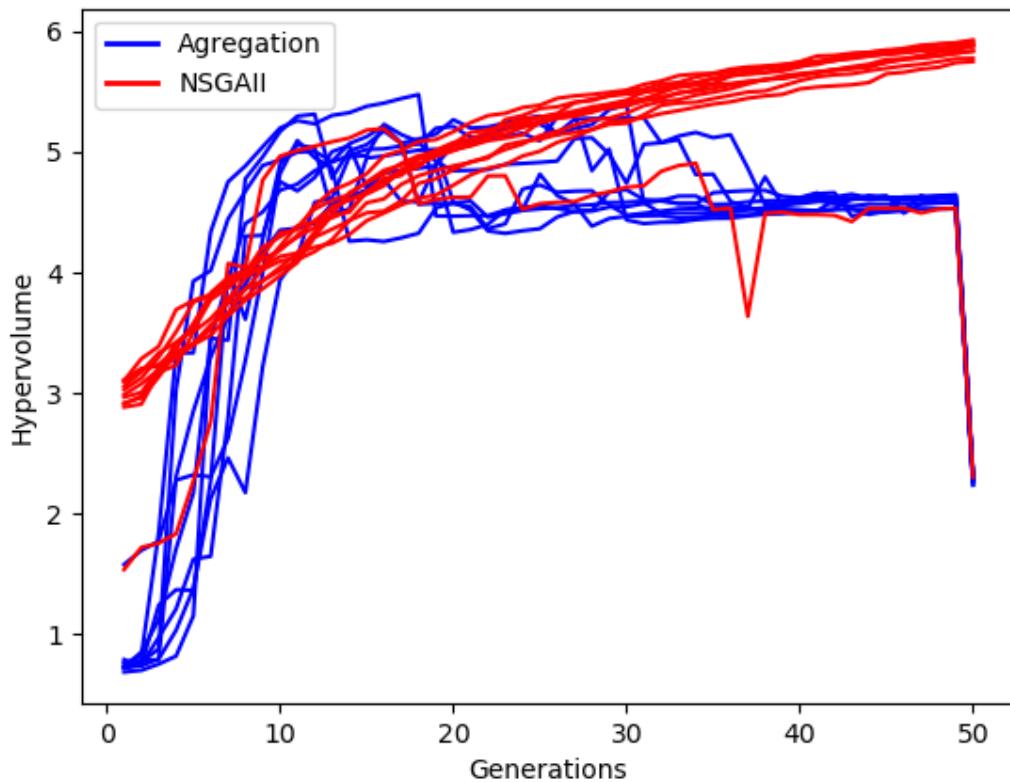


80 individuos y 50 generaciones

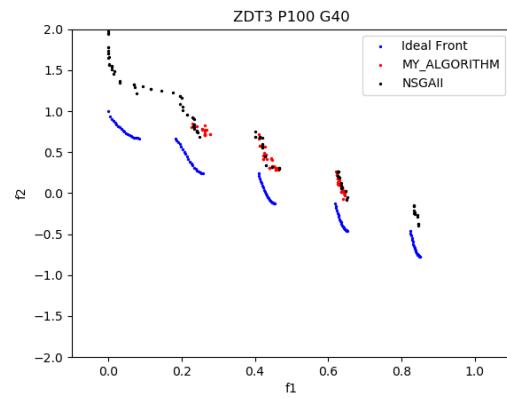
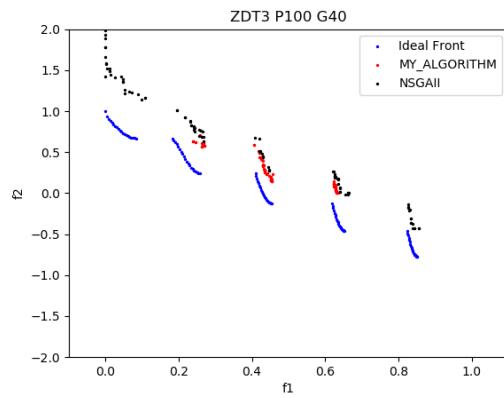
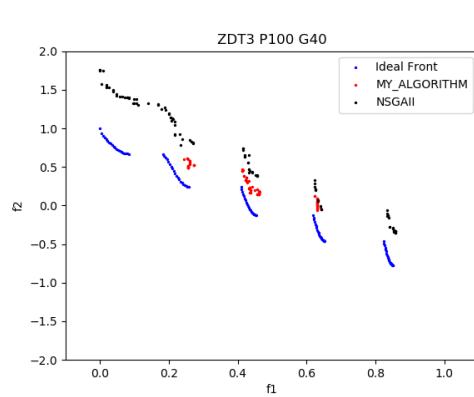
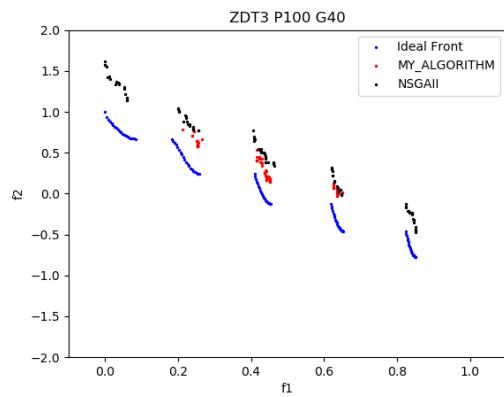
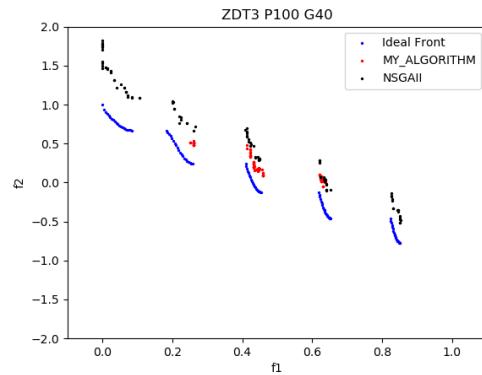
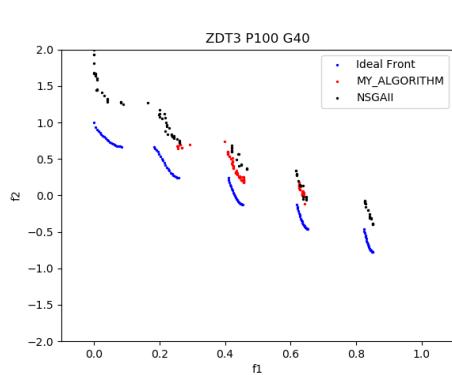


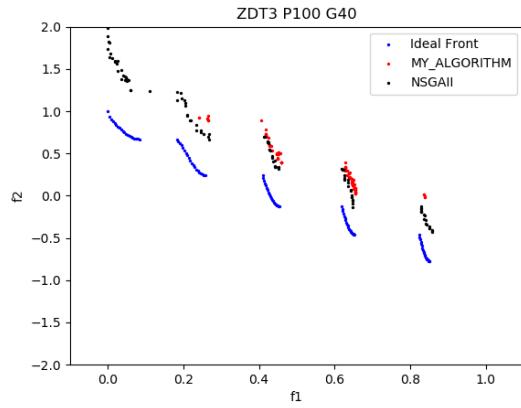
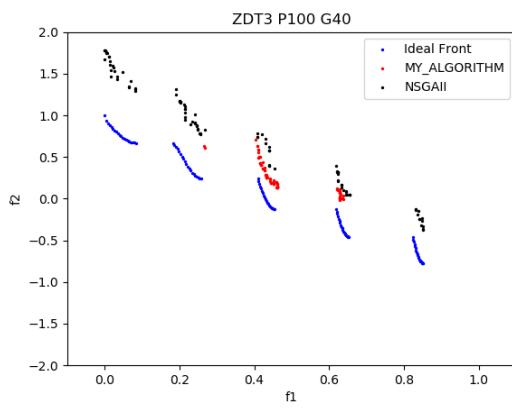
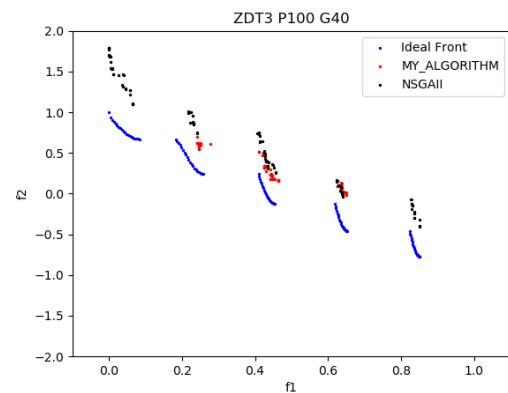
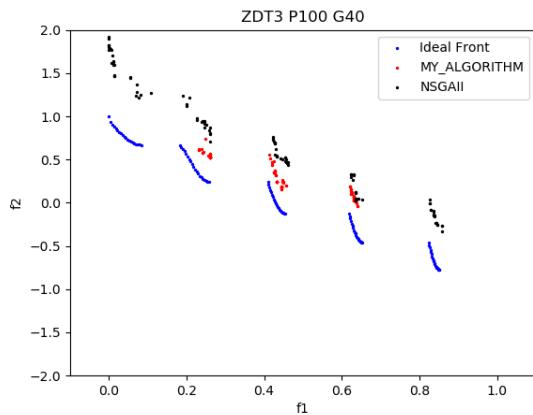


P80G50-> Mean my_alg: 2.052 Std my_alg: 0.684
Mean nsgaii: 5.857 Std nsgaii : 0.055
Reference point: [1. 6.232157]

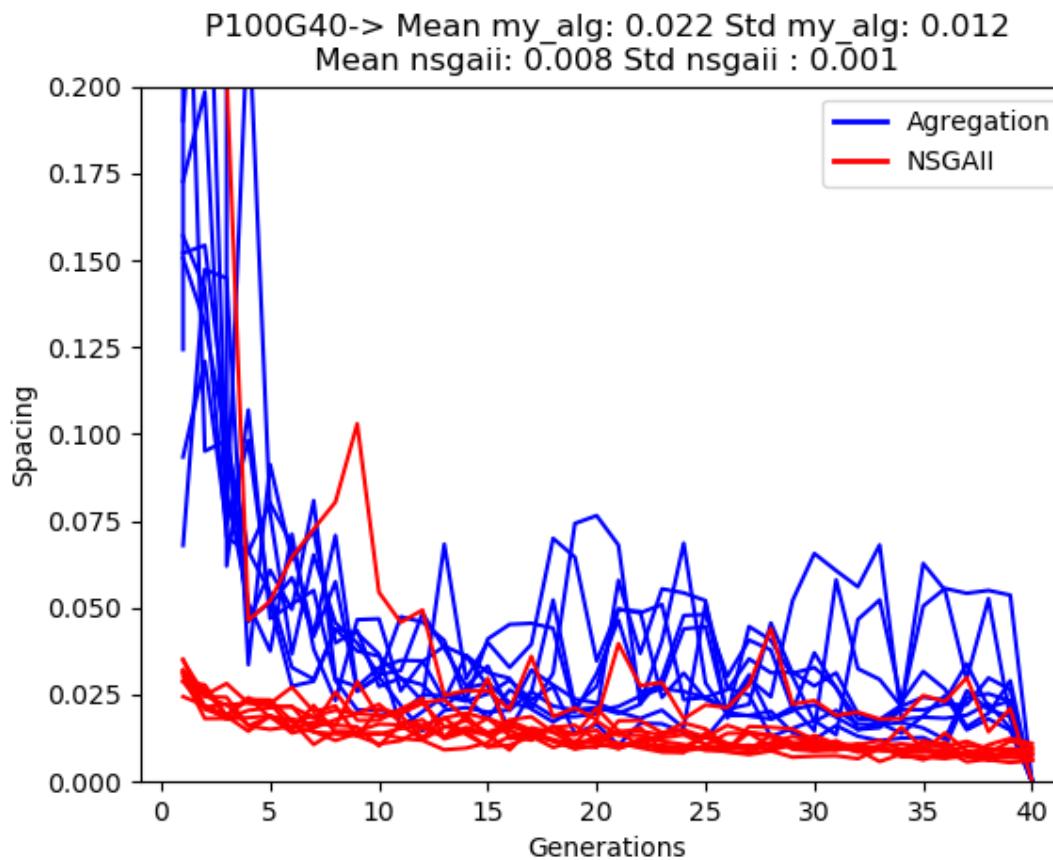
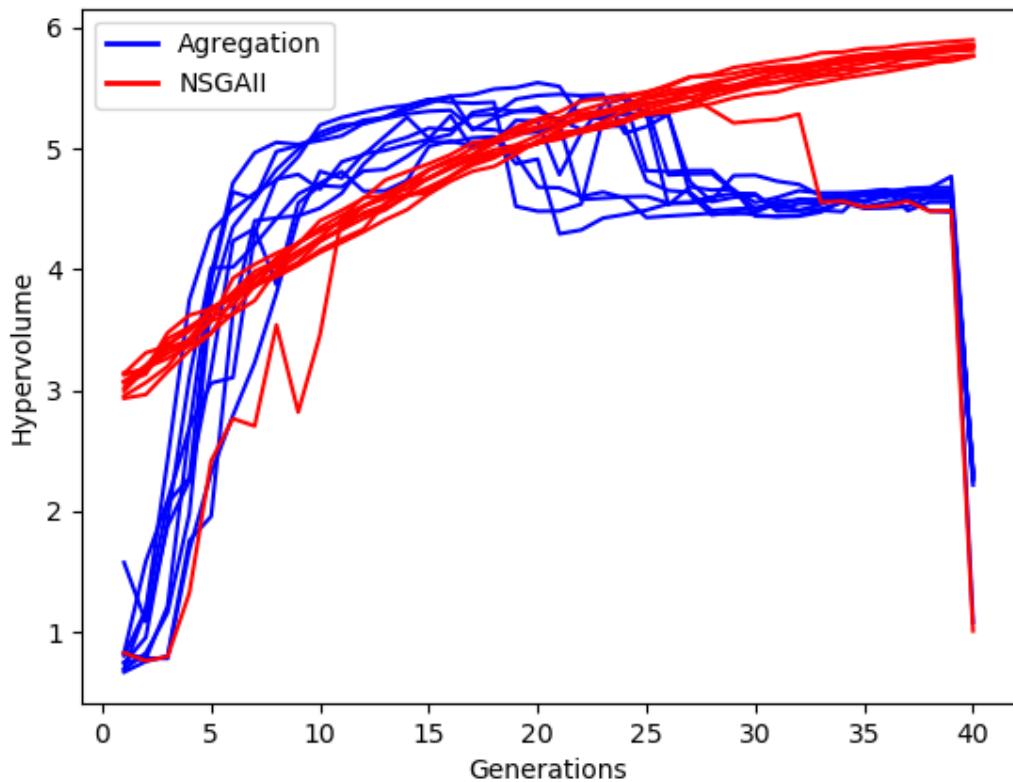


100 individuos y 40 generaciones

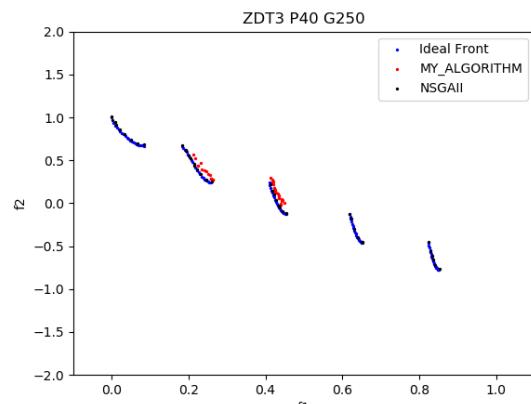
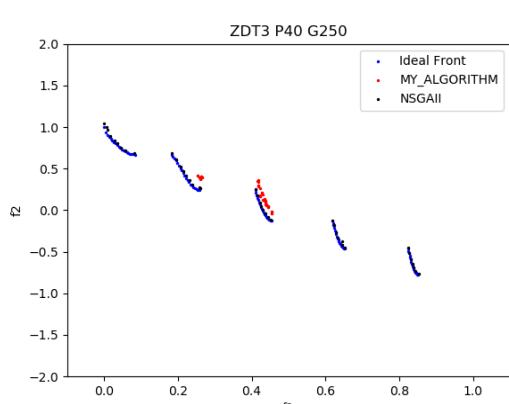
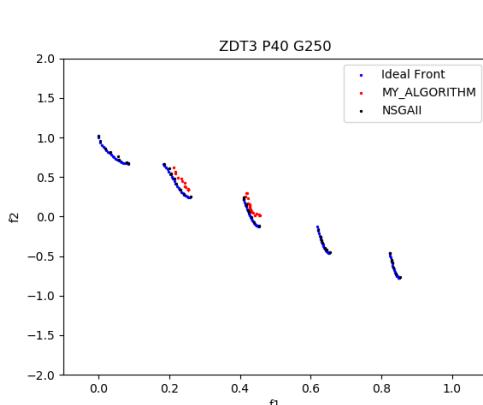
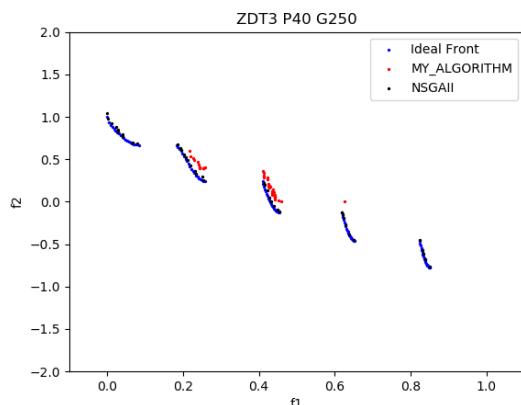
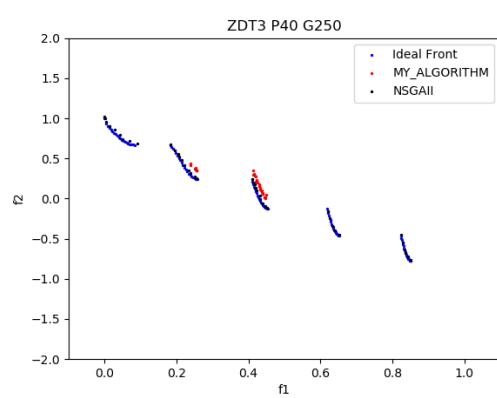
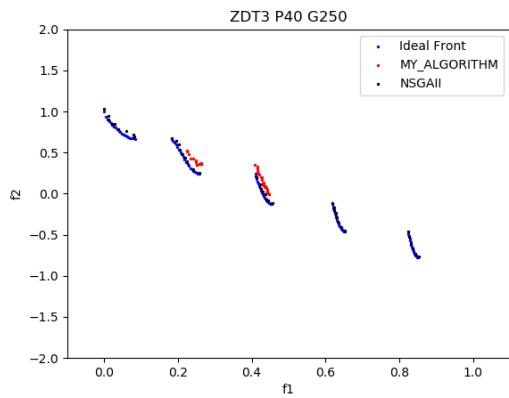


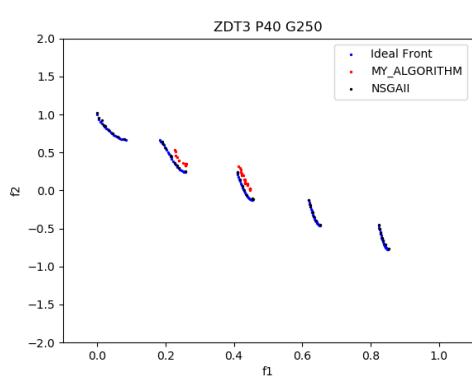
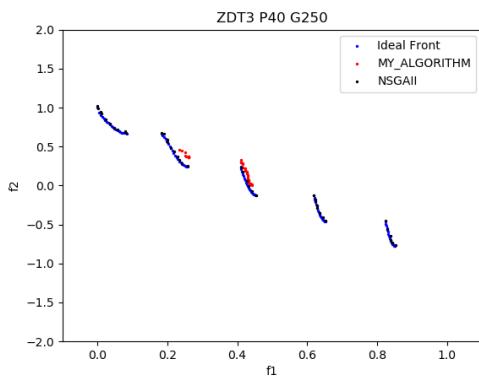
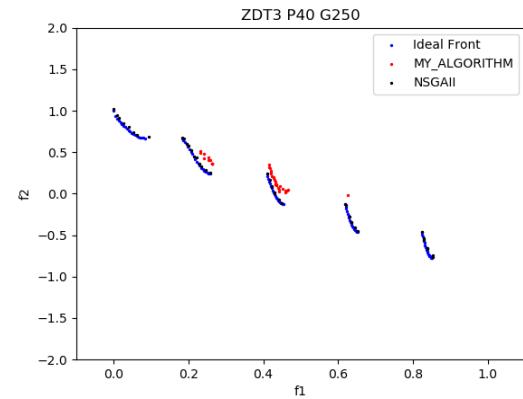
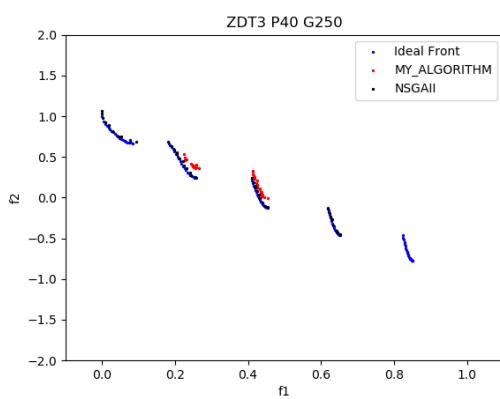


P100G40-> Mean my_alg: 1.933 Std my_alg: 0.737
Mean nsgaii: 5.824 Std nsgaii : 0.043
Reference point: [0.999585 6.250082]

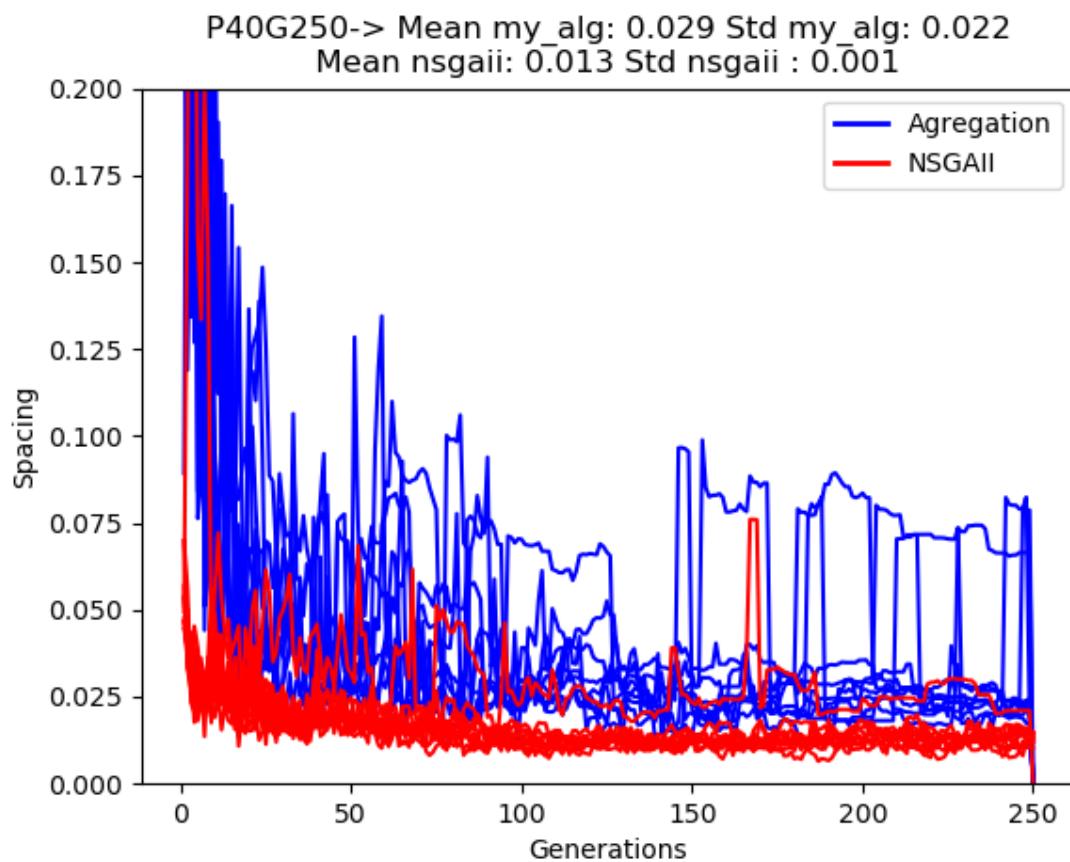
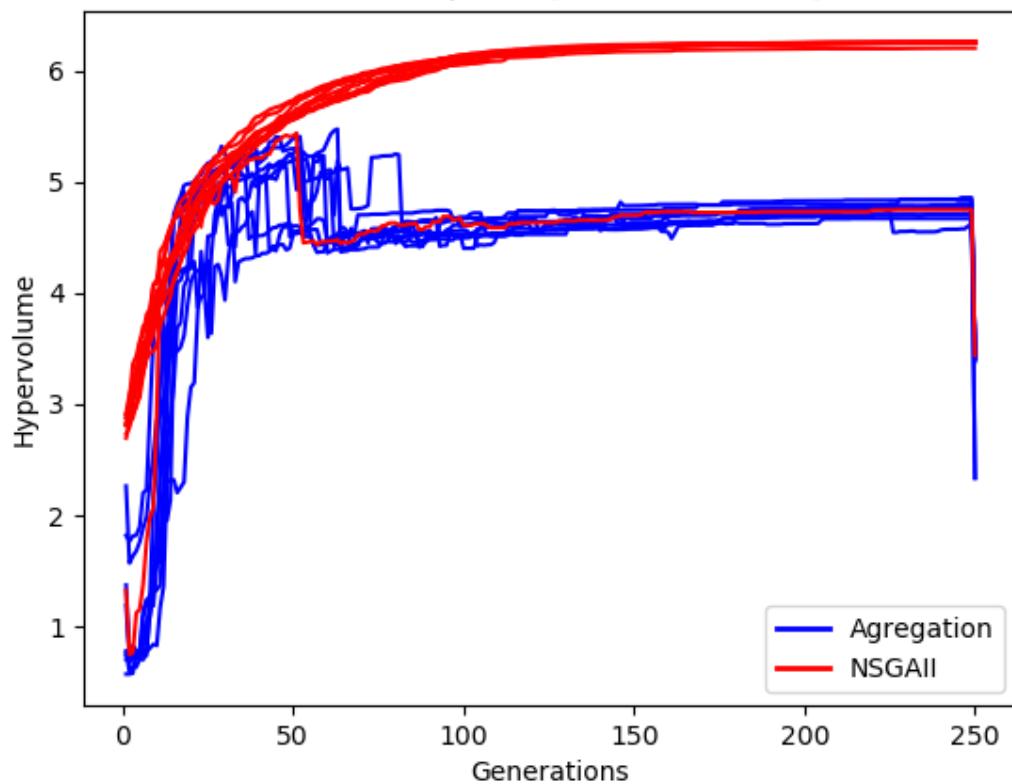


40 individuos y 250 generaciones

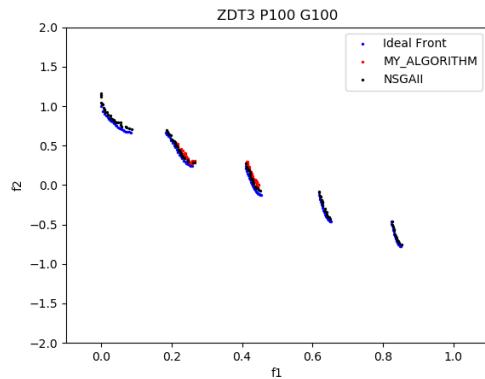
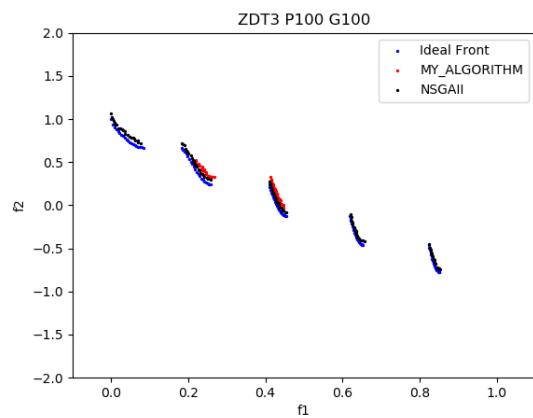
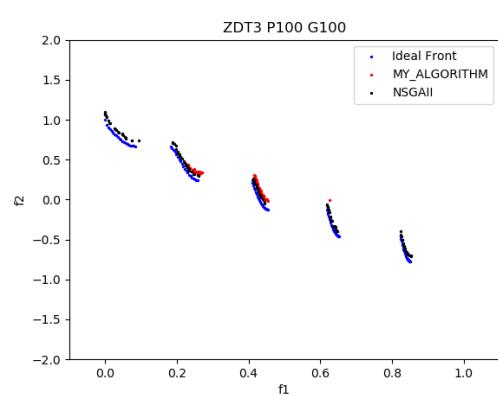
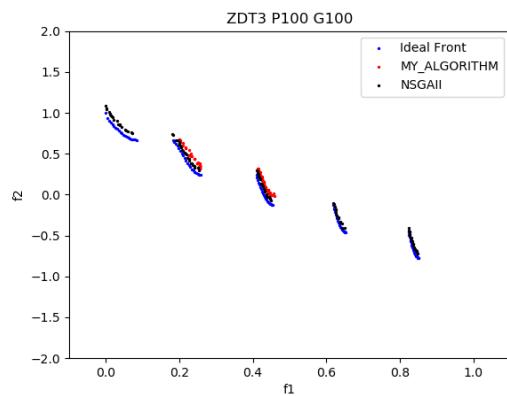
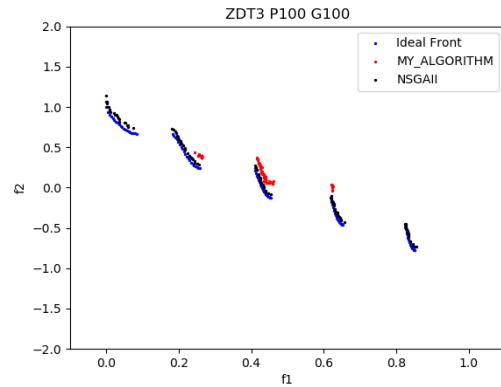
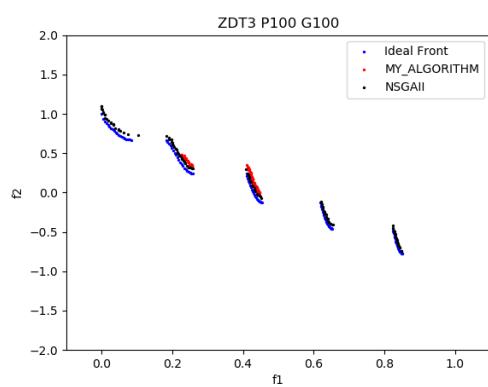


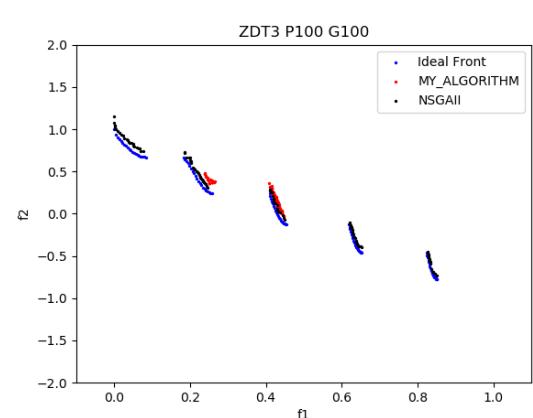
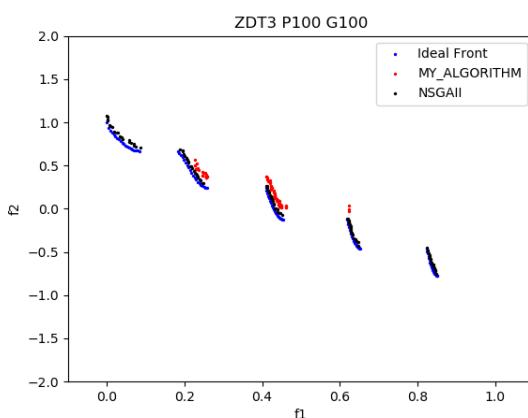
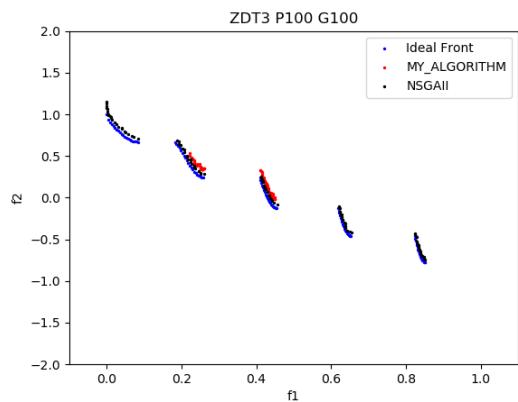
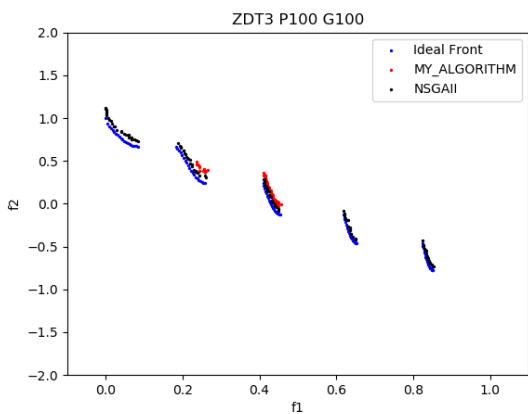


P40G250-> Mean my_alg: 2.865 Std my_alg: 1.047
Mean nsgaii: 6.251 Std nsgaii : 0.015
Reference point: [1. 6.232157]

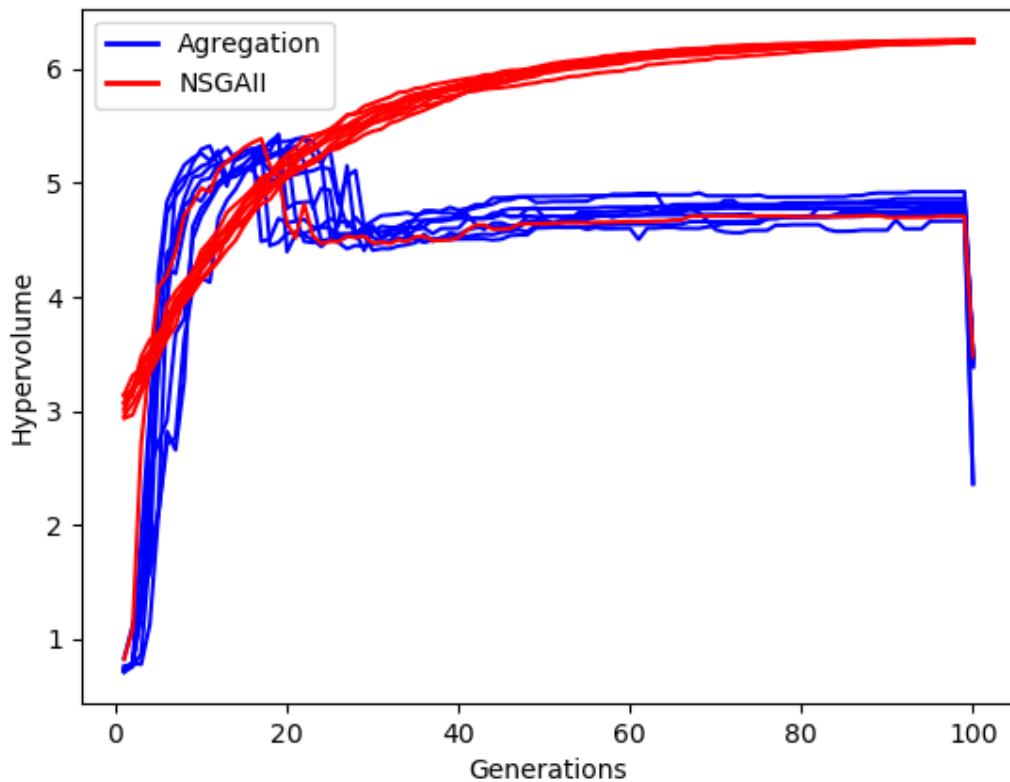


100 individuos y 100 generaciones

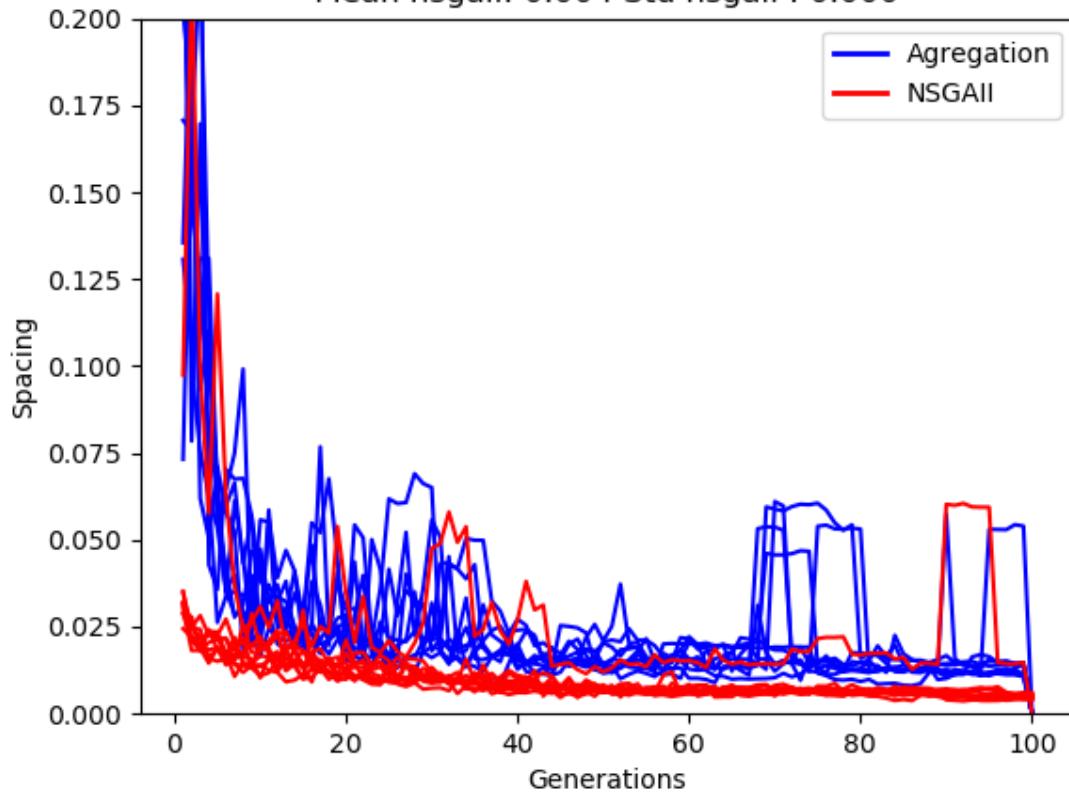




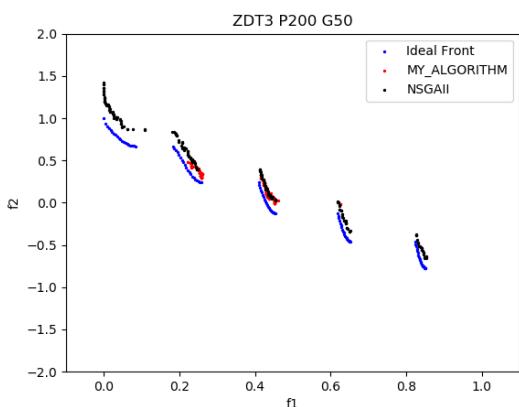
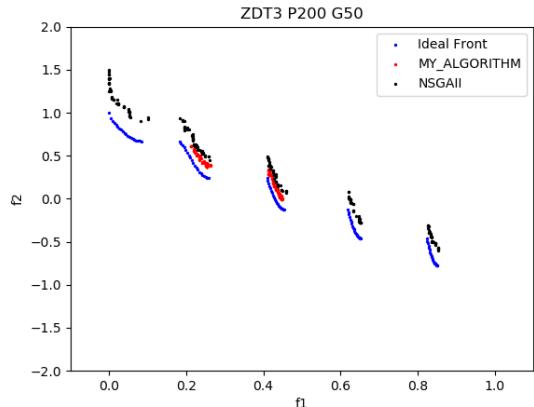
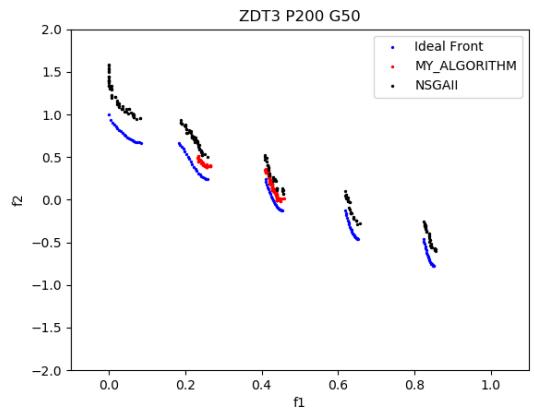
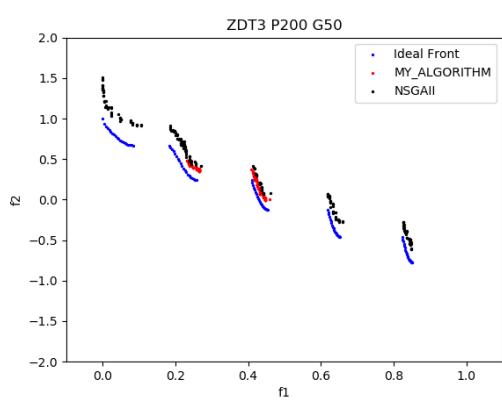
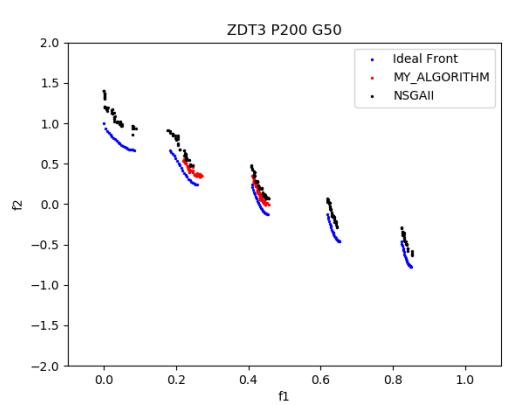
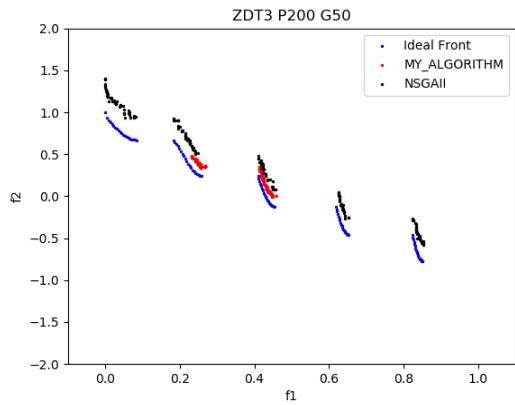
P100G100-> Mean my_alg: 2.878 Std my_alg: 1.049
Mean nsgaii: 6.240 Std nsgaii : 0.008
Reference point: [1. 6.250082]

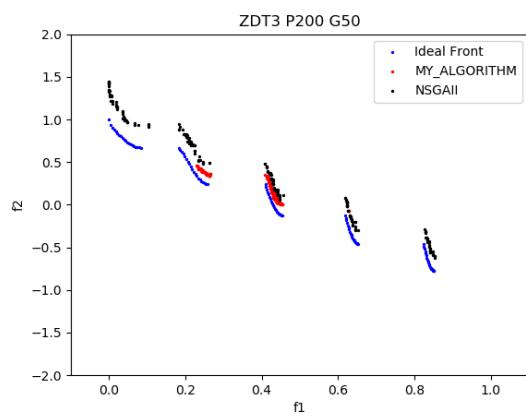
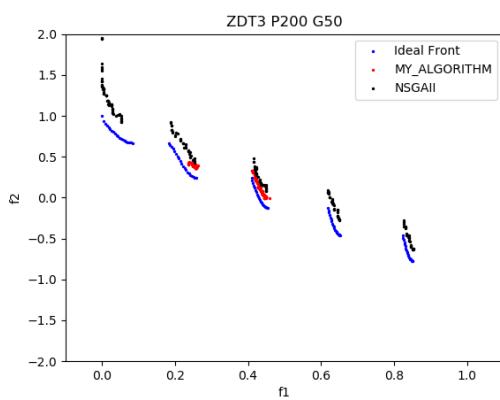
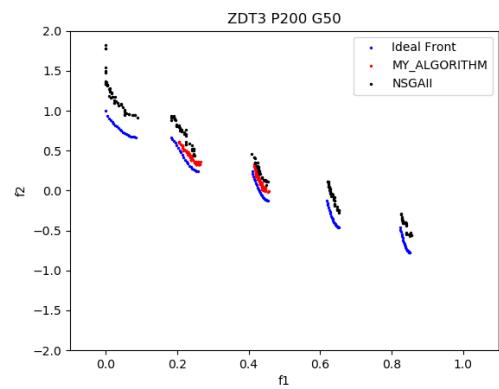
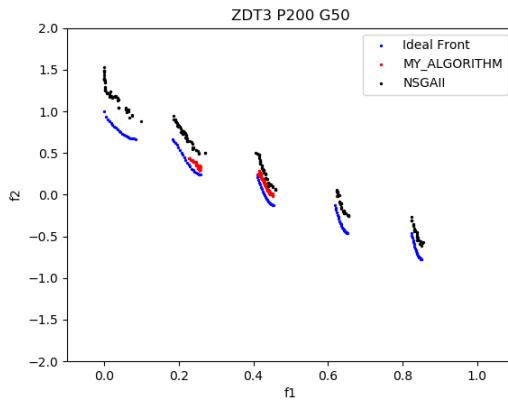


P100G100-> Mean my_alg: 0.015 Std my_alg: 0.013
Mean nsgaii: 0.004 Std nsgaii : 0.000

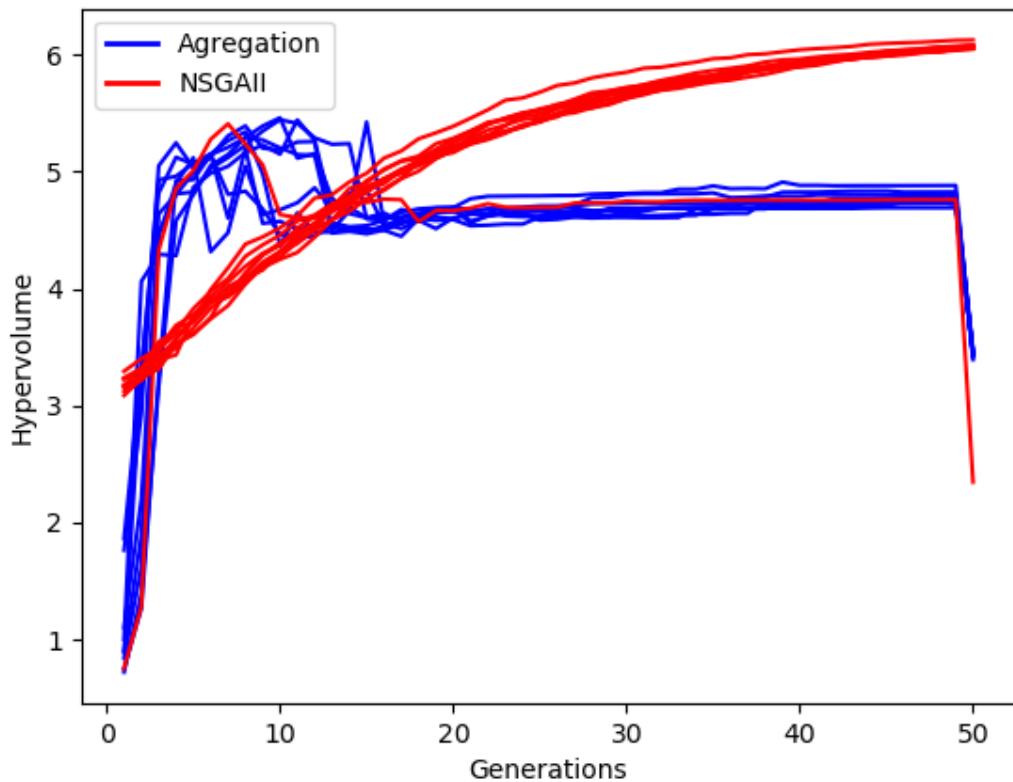


200 individuos y 50 generaciones

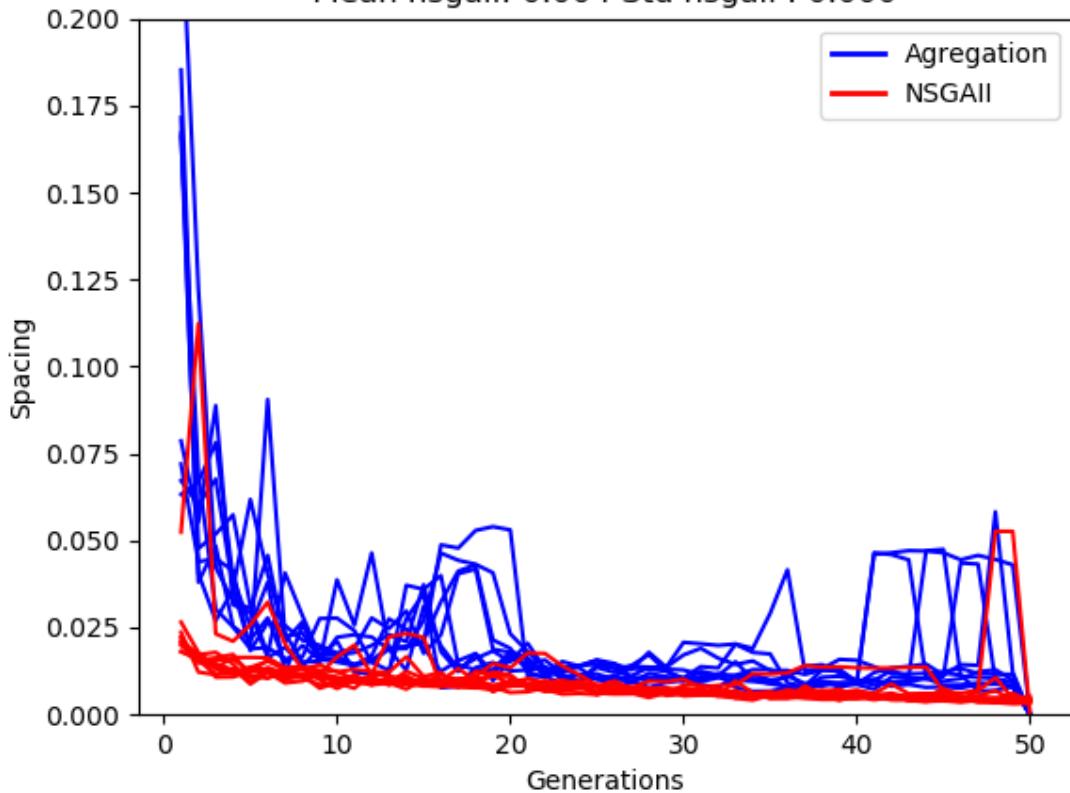




P200G50-> Mean my_alg: 3.091 Std my_alg: 1.030
Mean nsgaii: 6.072 Std nsgaii : 0.020
Reference point: [1. 6.233711]



P200G50-> Mean my_alg: 0.011 Std my_alg: 0.010
Mean nsgaii: 0.004 Std nsgaii : 0.000



Conclusiones

A priori observamos que tanto el algoritmo basado en agregación como el algoritmo NSGAII encuentran buenas soluciones que se acercan a una pequeña distancia al frente ideal.

Sobre todo, vemos esta convergencia en las combinaciones de diez mil evaluaciones, lo cual es evidente porque tenemos mayor reproducción y por tanto mejora de la población.

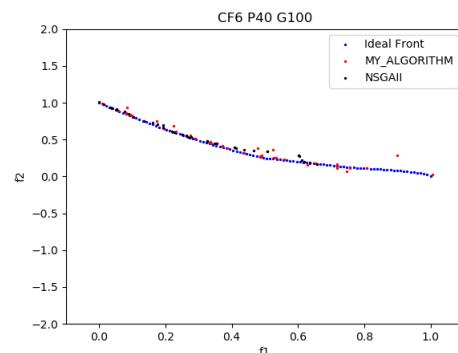
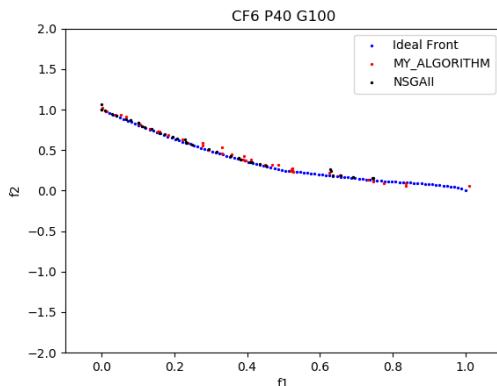
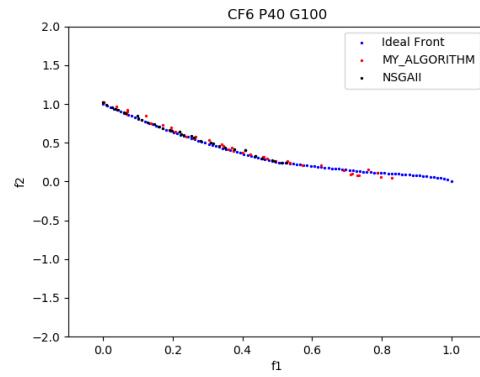
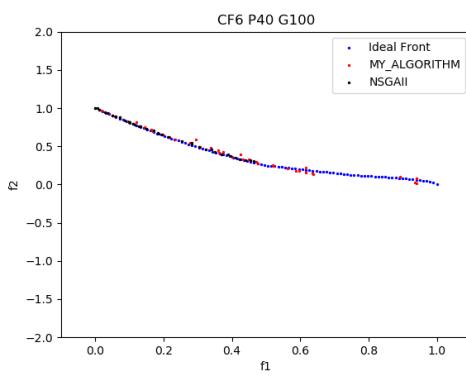
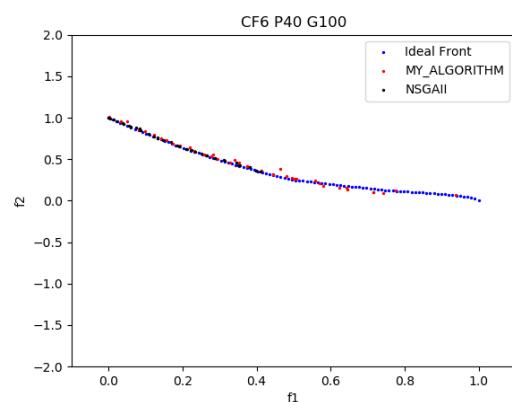
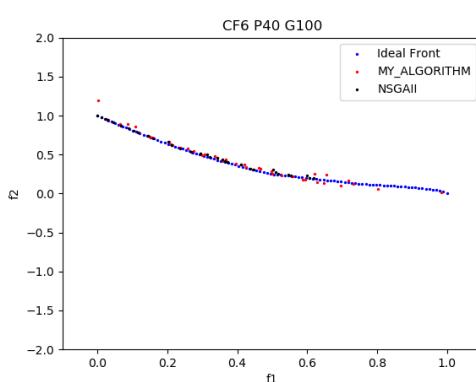
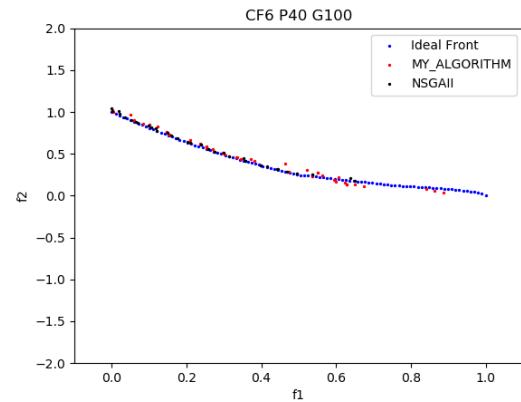
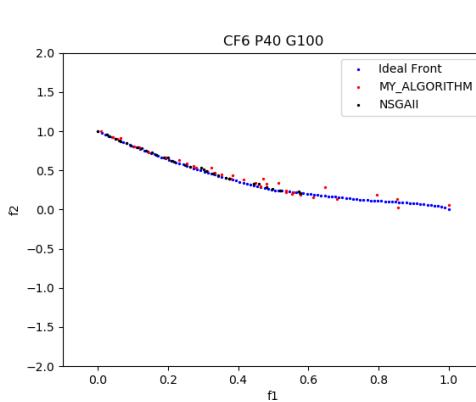
En cuanto a comparación, observamos que para 4000 evaluaciones, el algoritmo de agregación encuentra algunas mejores soluciones, aunque con muy poca diversidad (En casi todos los ejemplos solo converge a 3 de las 5 partes) y aunque NSGAII no llega a superar las mejores soluciones de agregación, consigue una población con mucha diversidad quedándose cerca las partes del frente en prácticamente todas las ejecuciones. Esto lo podemos ver reflejado en las gráficas de spacing, que son mejores para NSGAII por esta diversidad (menor valor implica mejor spacing) y en hipervolumen se demuestra que aunque agregación consigue algunas soluciones concretas mejores, en conjunto NSGAII produce mejor población de soluciones.

Para 10000 evaluaciones vemos que NSGAII afina y consigue llegar a todas las partes del frente de manera muy aproximada. Agregación también lo consigue pero sigue habiendo el mismo problema de muy poca diversidad de soluciones, llegando incluso a solo 1 o 2 partes del frente. Esto se vuelve a demostrar en las gráficas de spacing e hipervolumen.

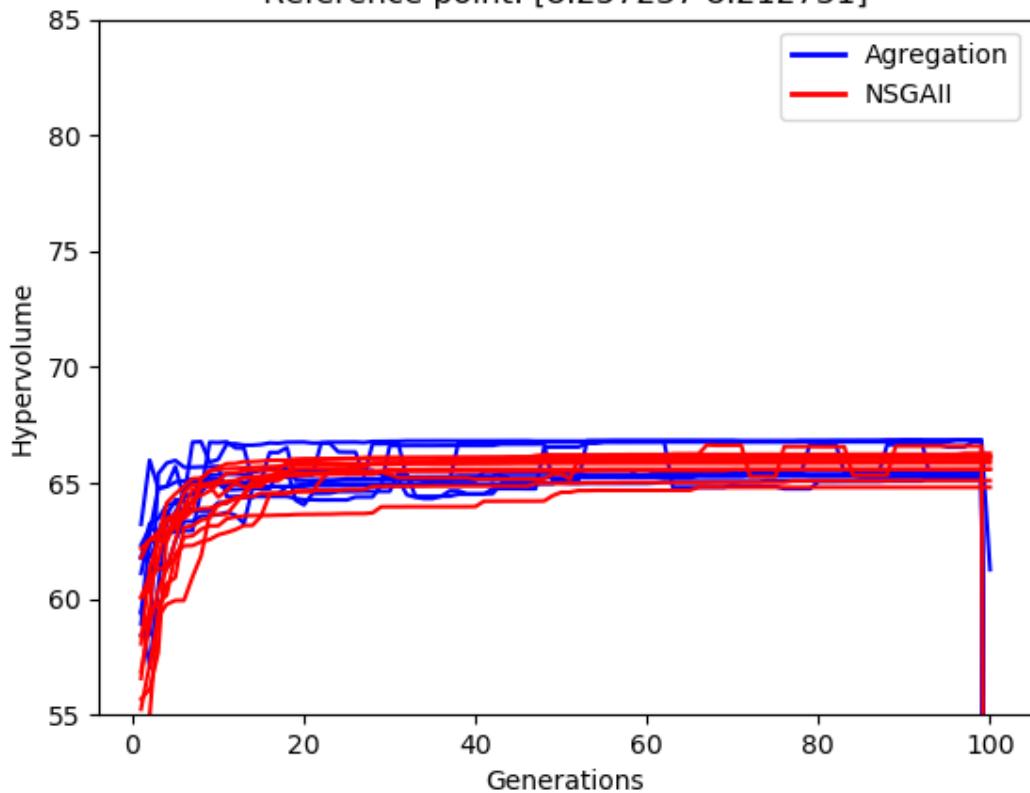
Para terminar, podemos decir que NSGAII resuelve de mejor forma el problema ZDT3, obteniendo poblaciones de soluciones más aproximadas y mejor equiespaciadas por el frente.

Problema CF6 en 4 dimensiones

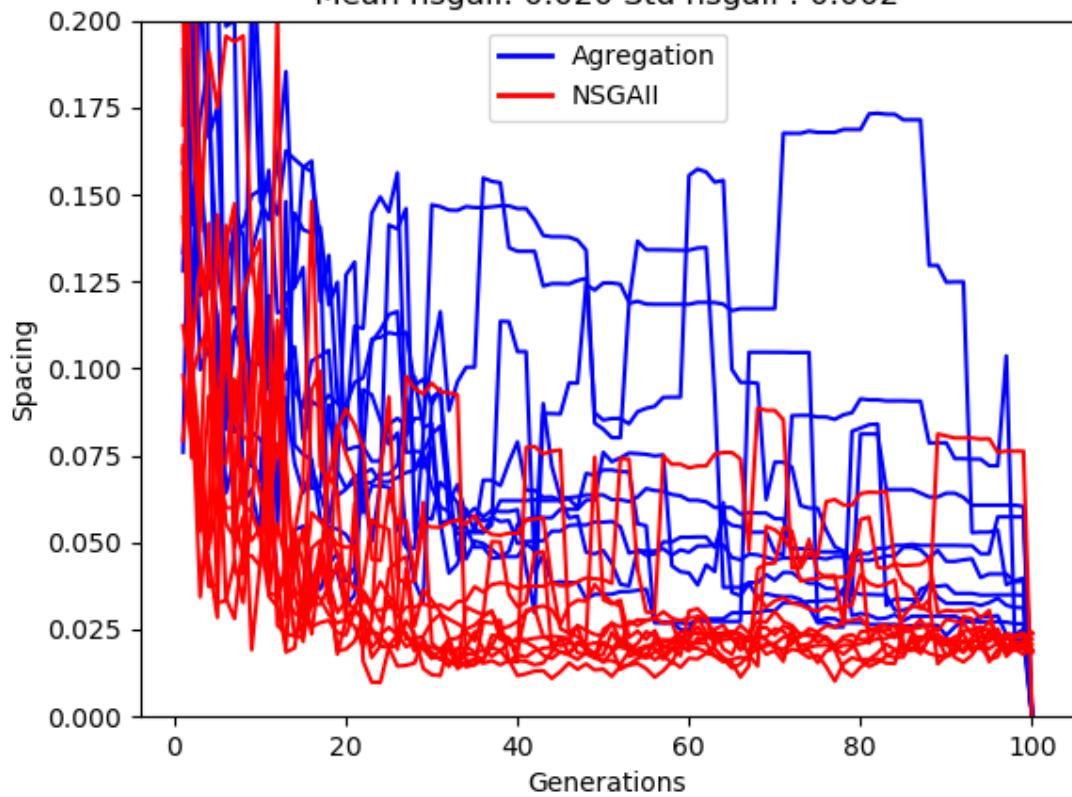
40 individuos y 100 generaciones



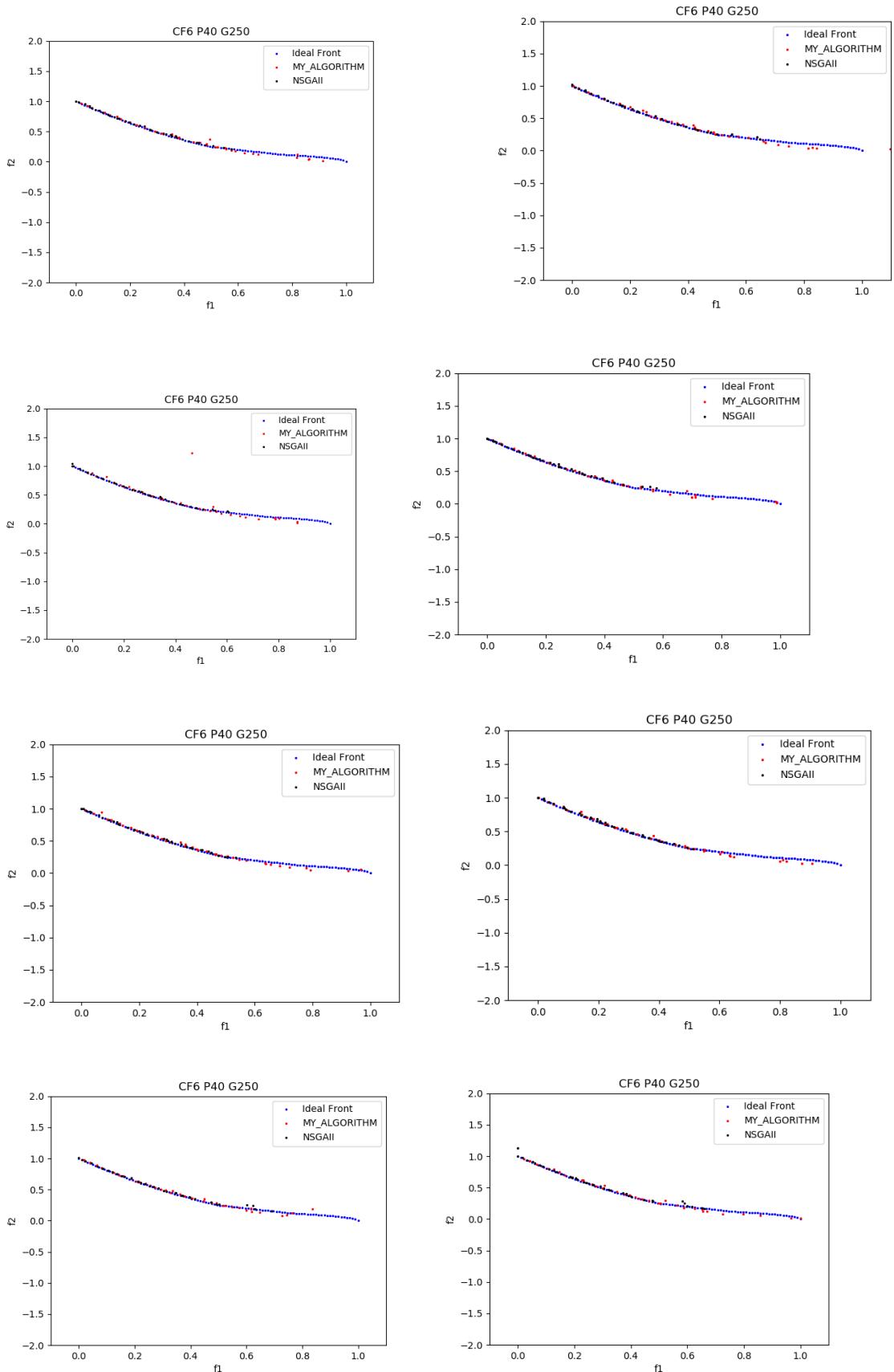
P40G100-> Mean my_alg: 63.33 Std my_alg: 19.79
Mean nsgaii: 65.75 Std nsgaii : 0.460
Reference point: [8.257237 8.212731]

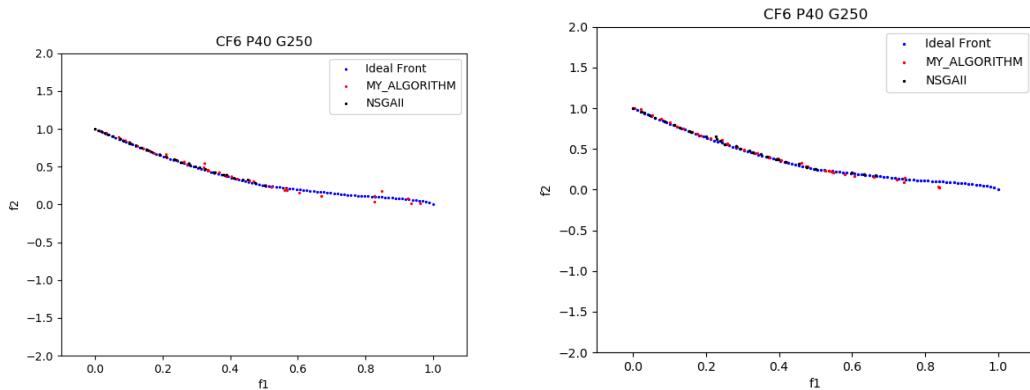


P40G100-> Mean my_alg: 0.033 Std my_alg: 0.016
Mean nsgaii: 0.020 Std nsgaii : 0.002

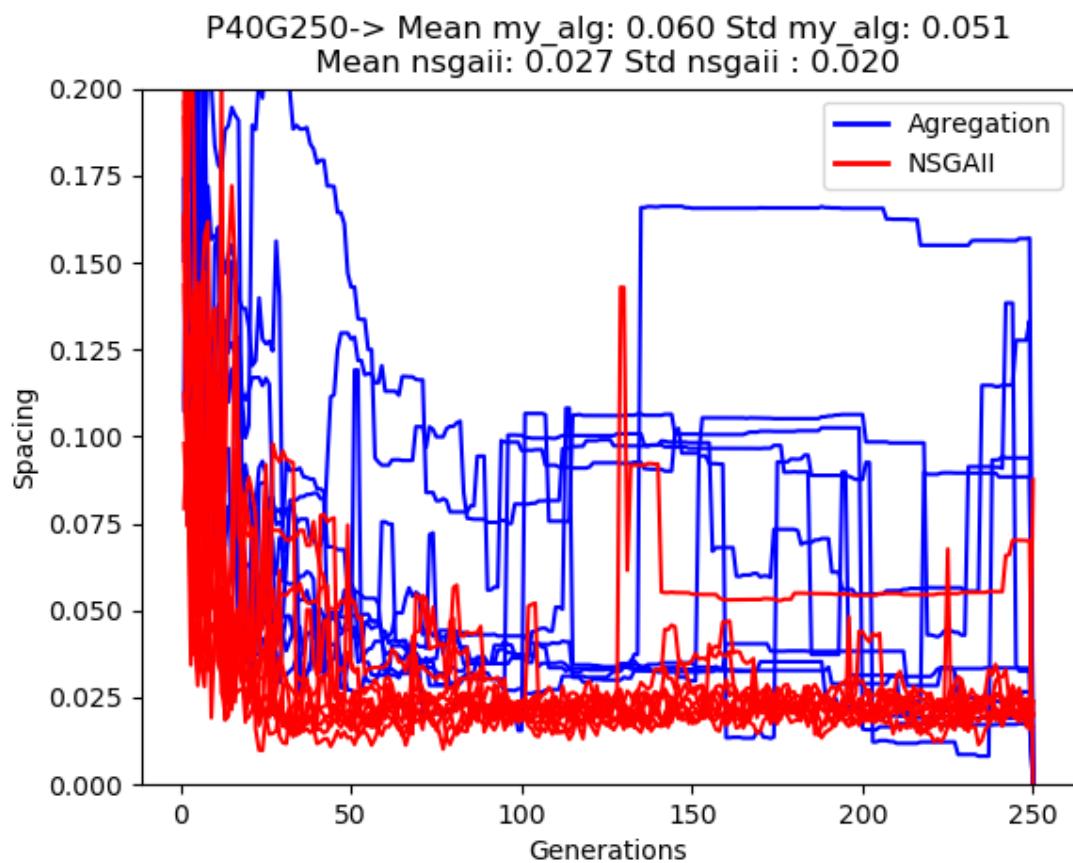
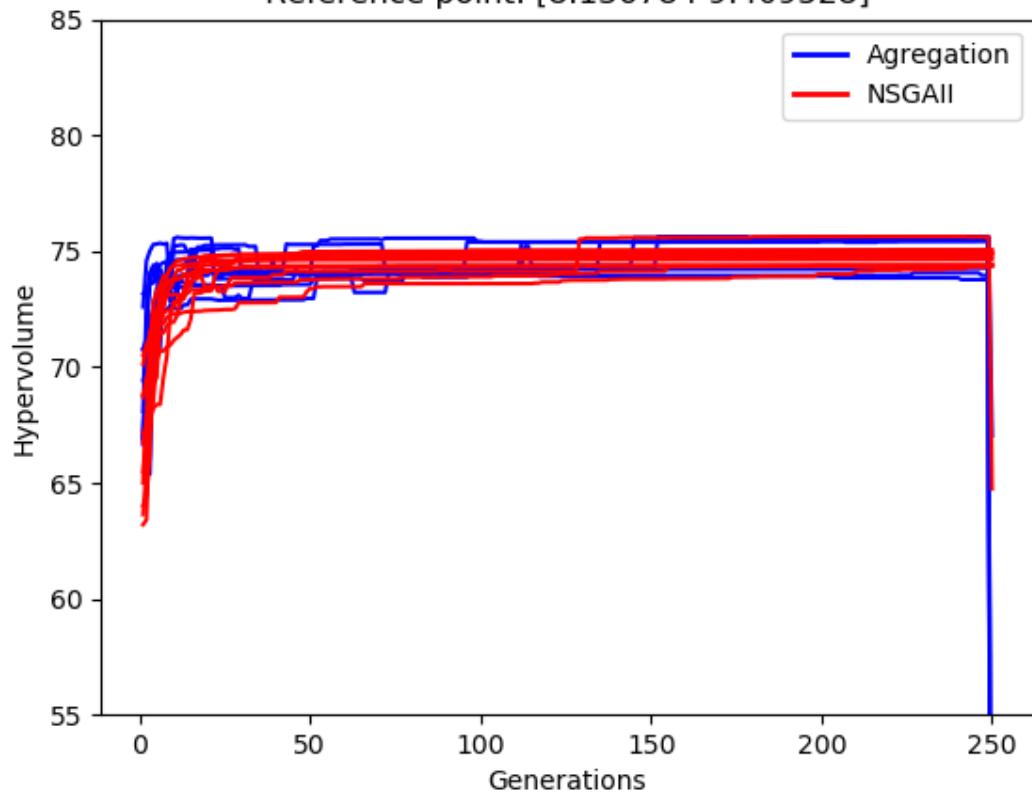


40 individuos y 250 generaciones

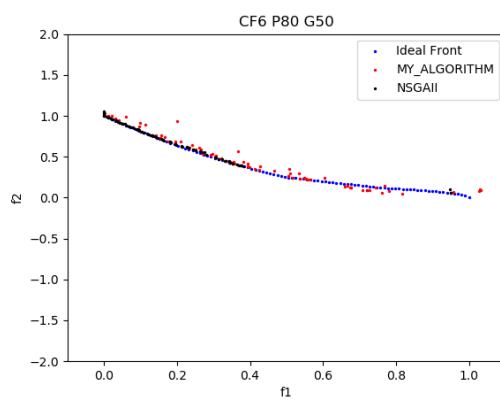
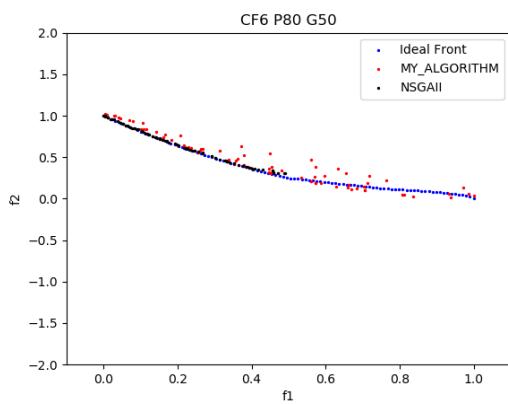
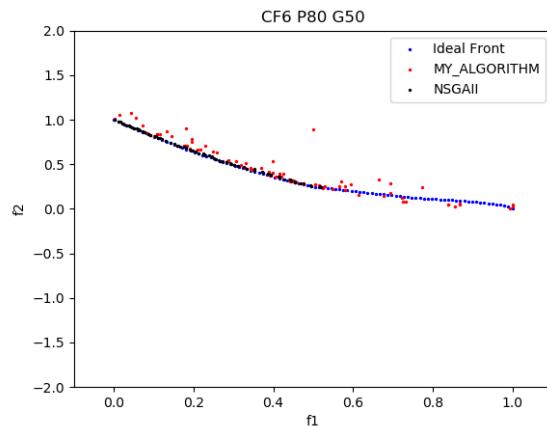
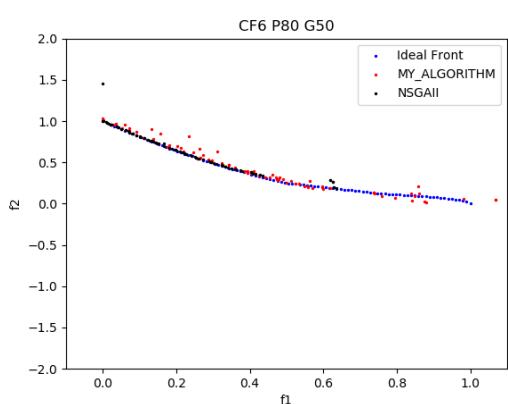
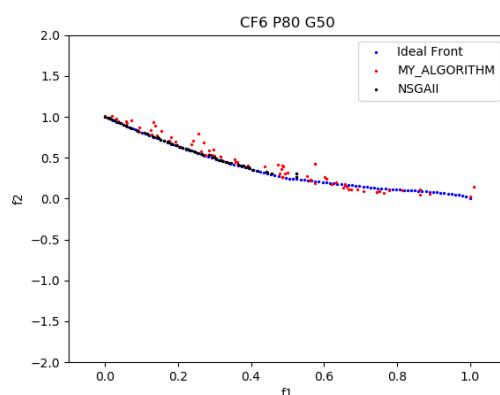
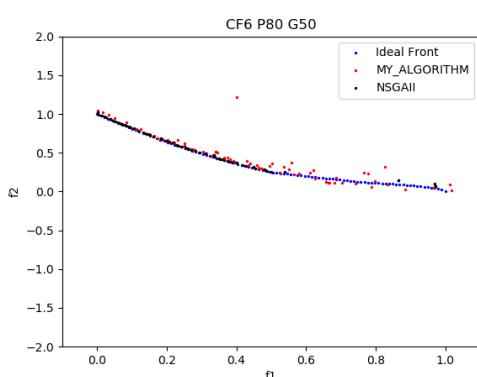
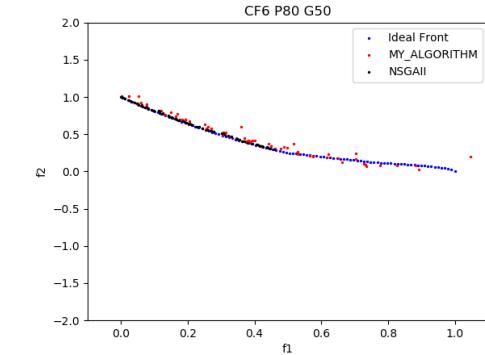
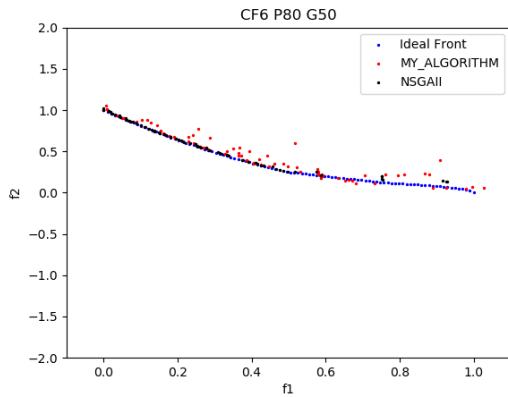


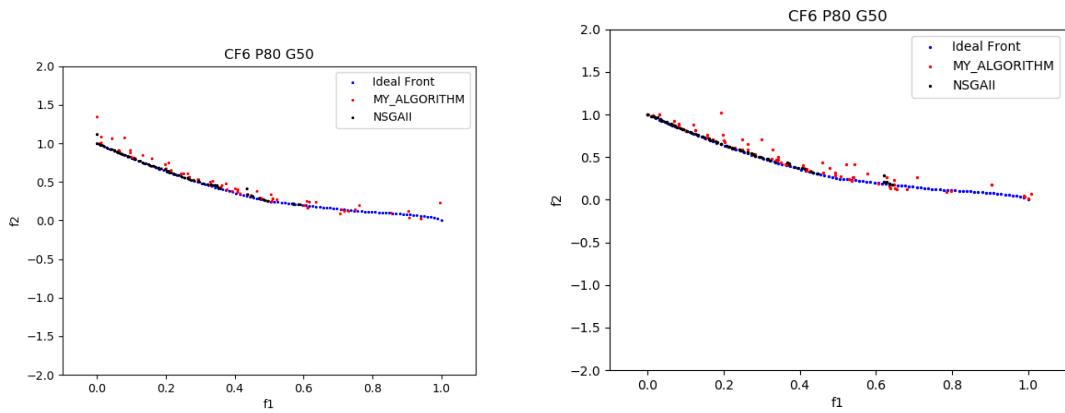


P40G250-> Mean my_alg: 71.51 Std my_alg: 22.51
Mean nsgaii: 74.67 Std nsgaii : 0.255
Reference point: [8.136784 9.409528]

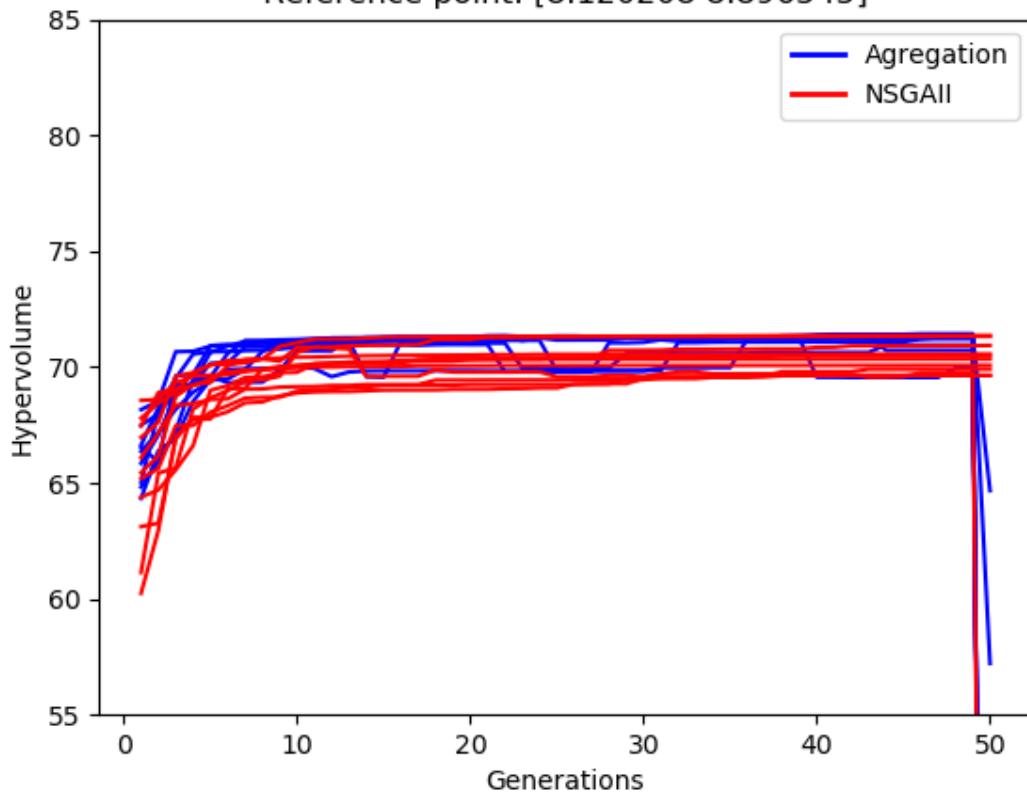


80 individuos y 50 generaciones

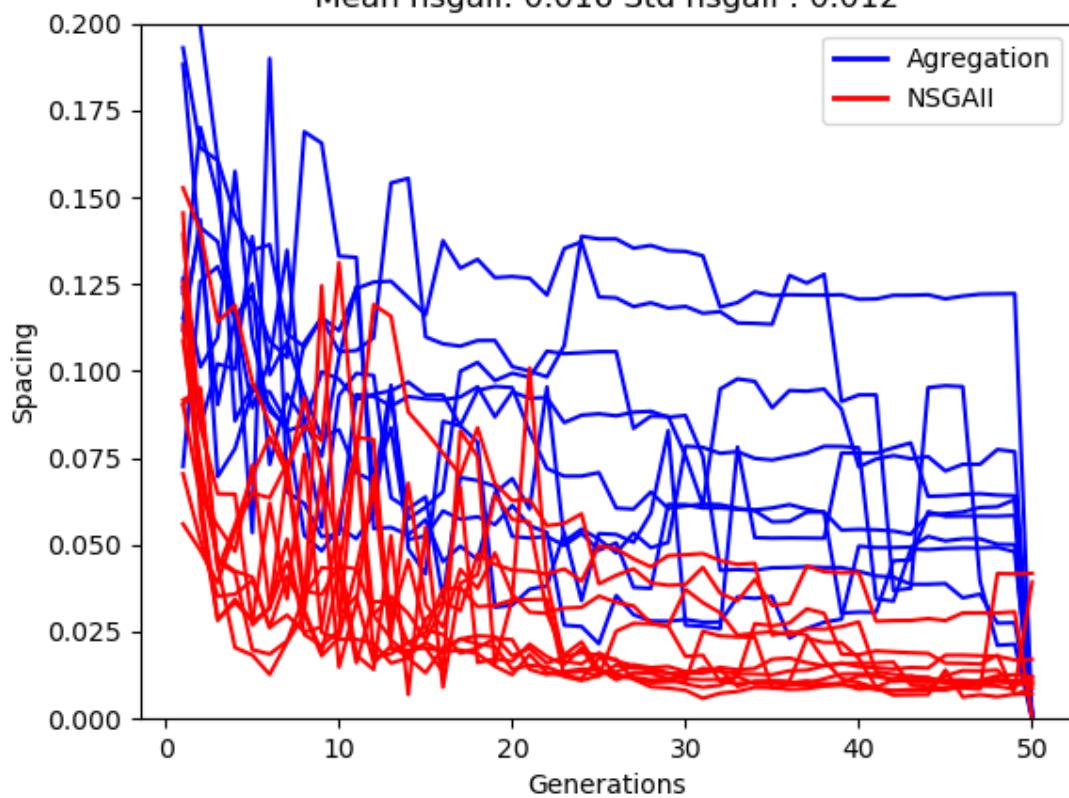




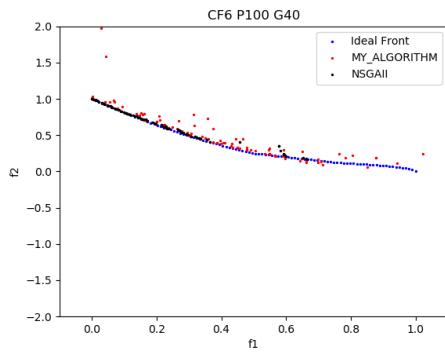
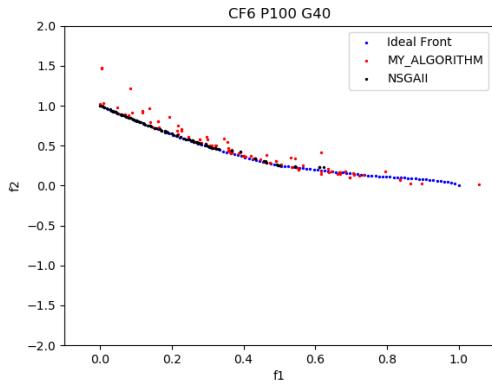
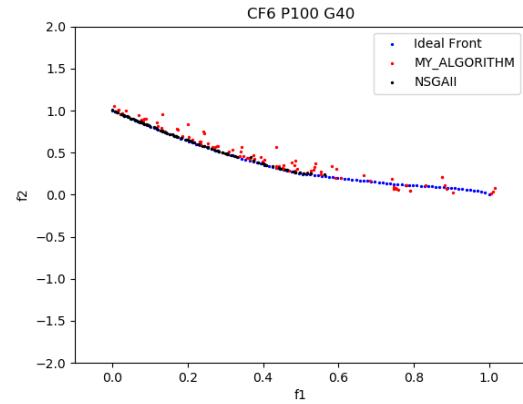
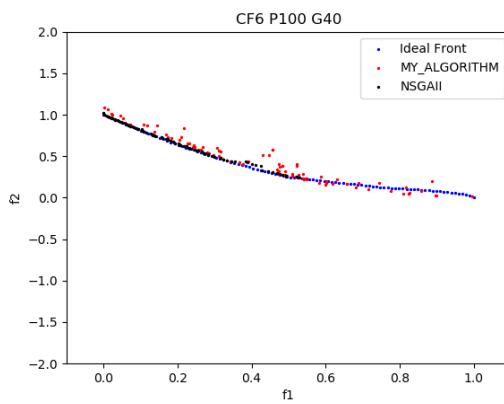
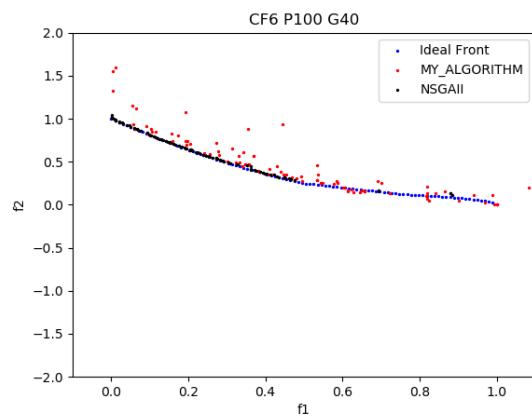
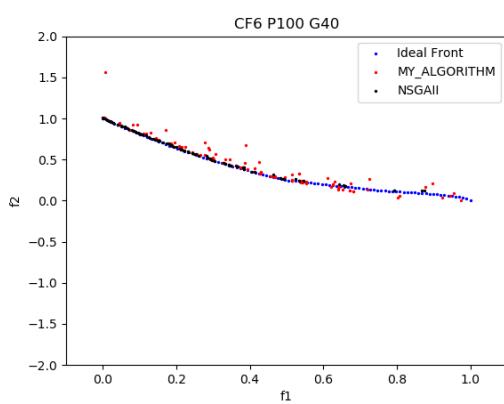
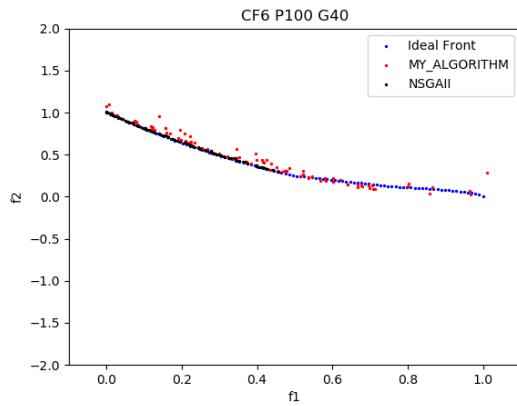
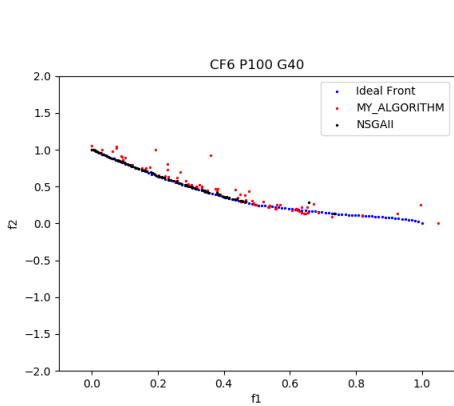
P80G50-> Mean my_alg: 67.91 Std my_alg: 21.31
Mean nsgaii: 70.42 Std nsgaii : 0.599
Reference point: [8.120208 8.896545]

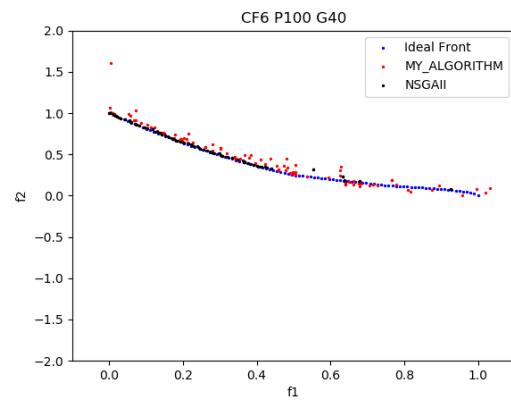
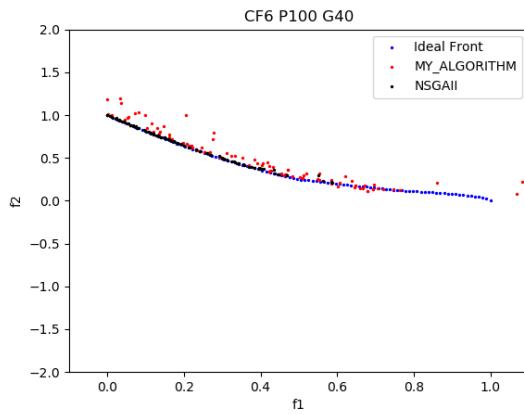


P80G50-> Mean my_alg: 0.053 Std my_alg: 0.031
Mean nsgaii: 0.016 Std nsgaii : 0.012

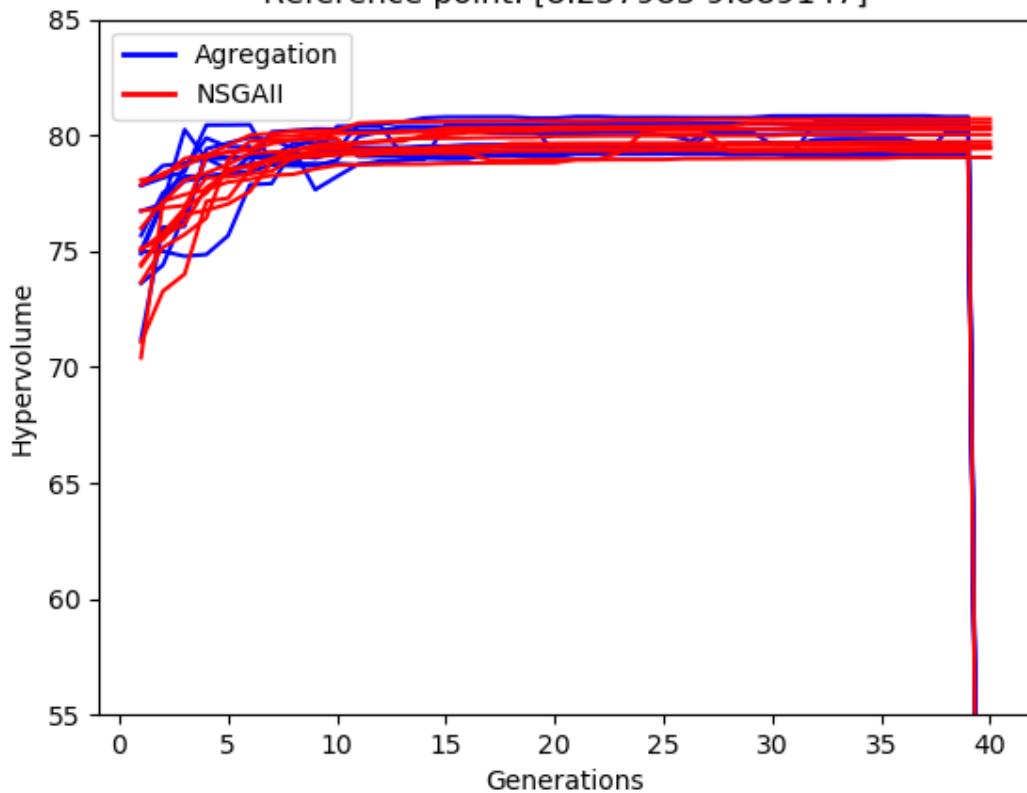


100 individuos y 40 generaciones

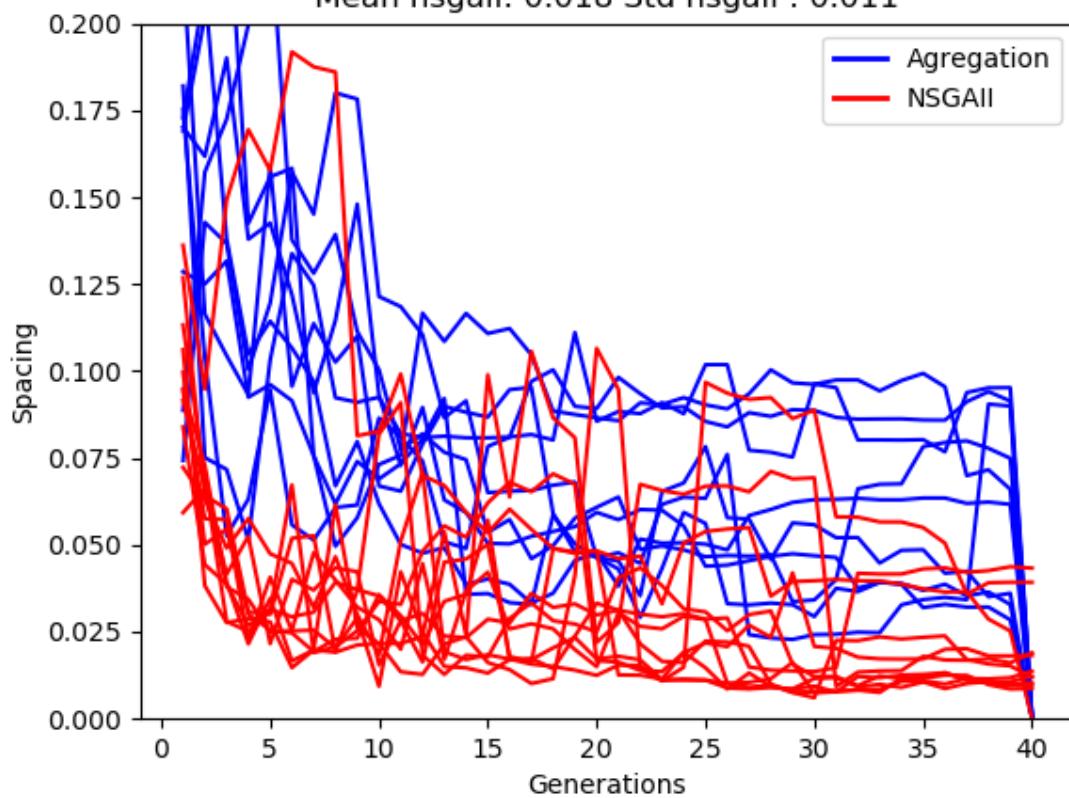




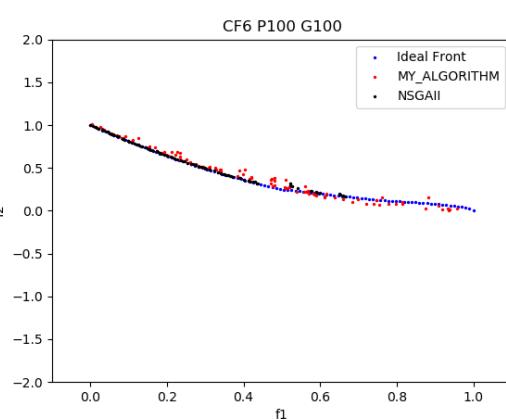
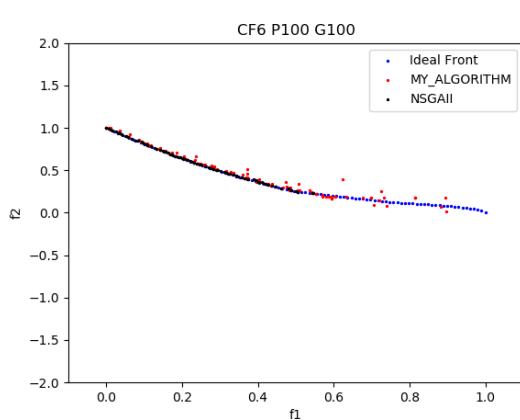
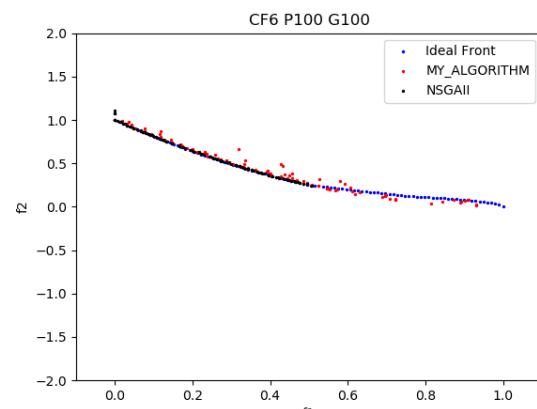
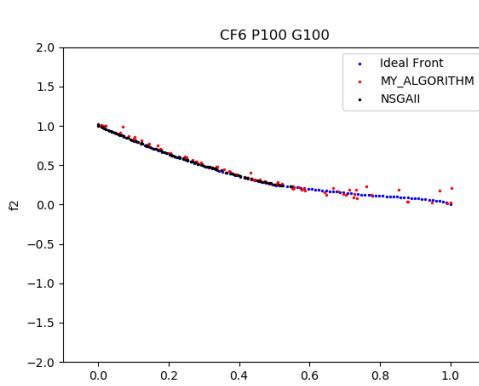
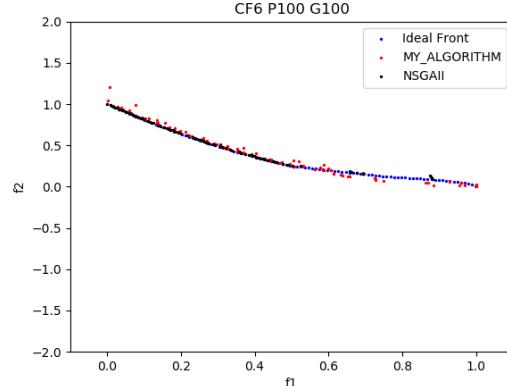
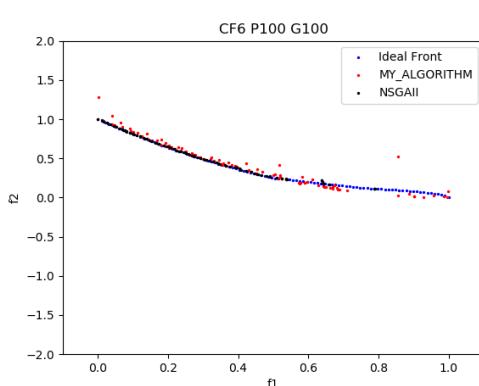
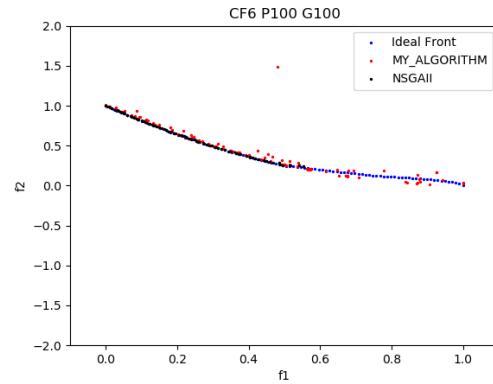
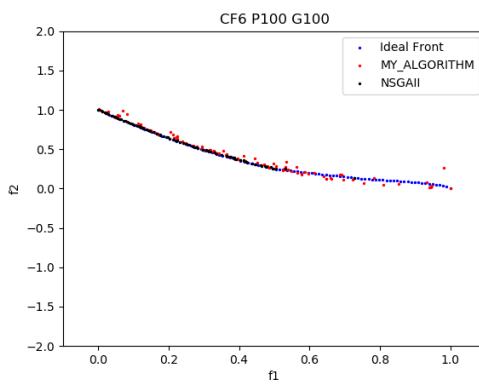
P100G40-> Mean my_alg: 75.73 Std my_alg: 23.91
Mean nsgaii: 79.91 Std nsgaii : 0.519
Reference point: [8.257983 9.889147]

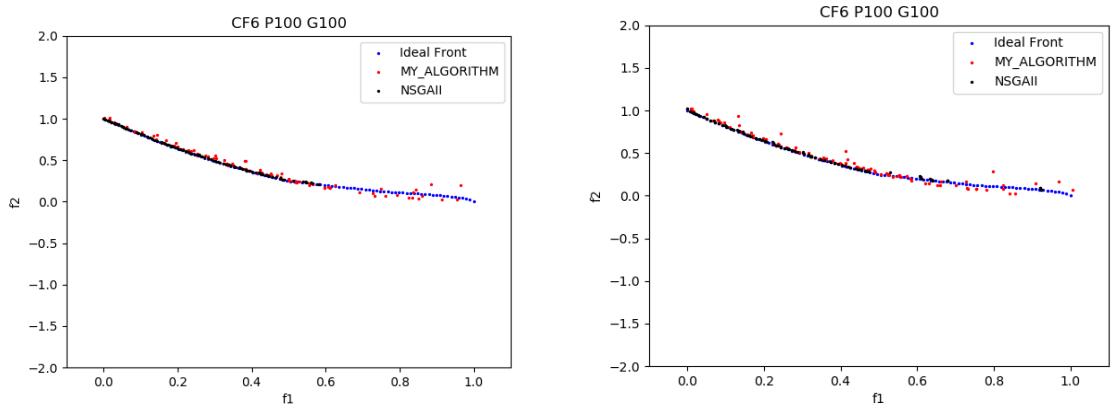


P100G40-> Mean my_alg: 0.057 Std my_alg: 0.030
Mean nsgaii: 0.018 Std nsgaii : 0.011

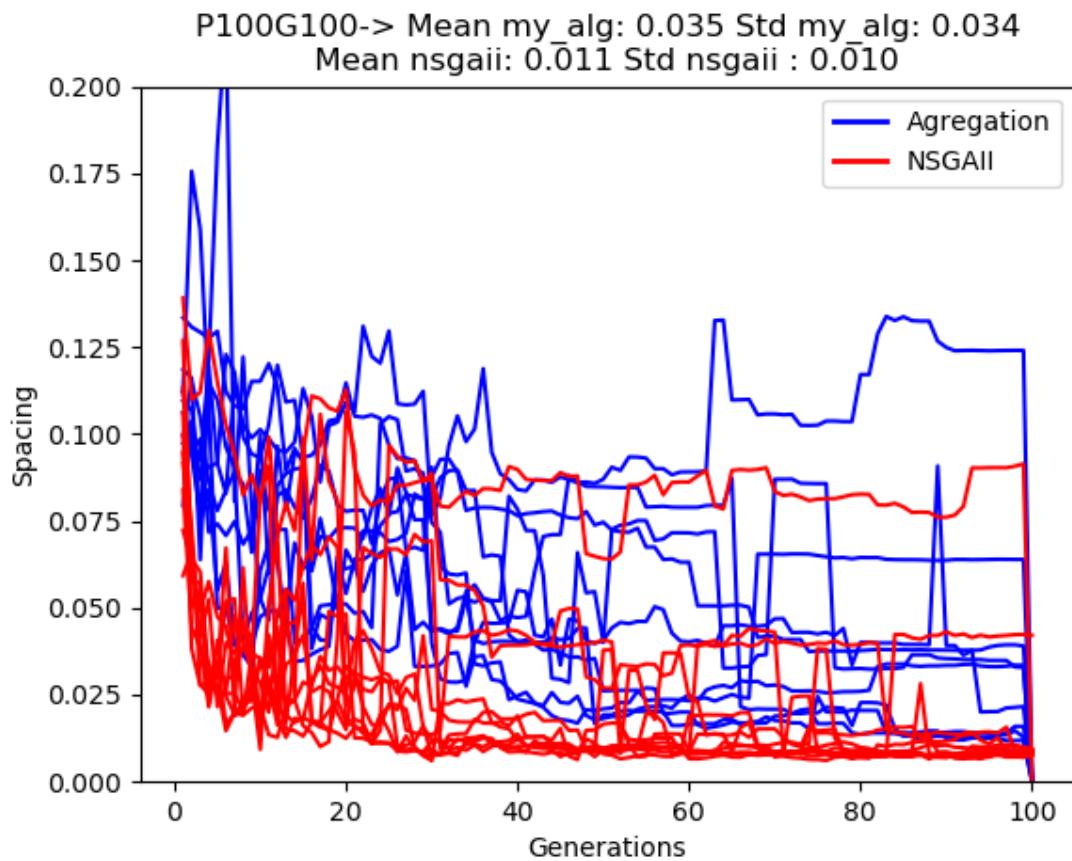
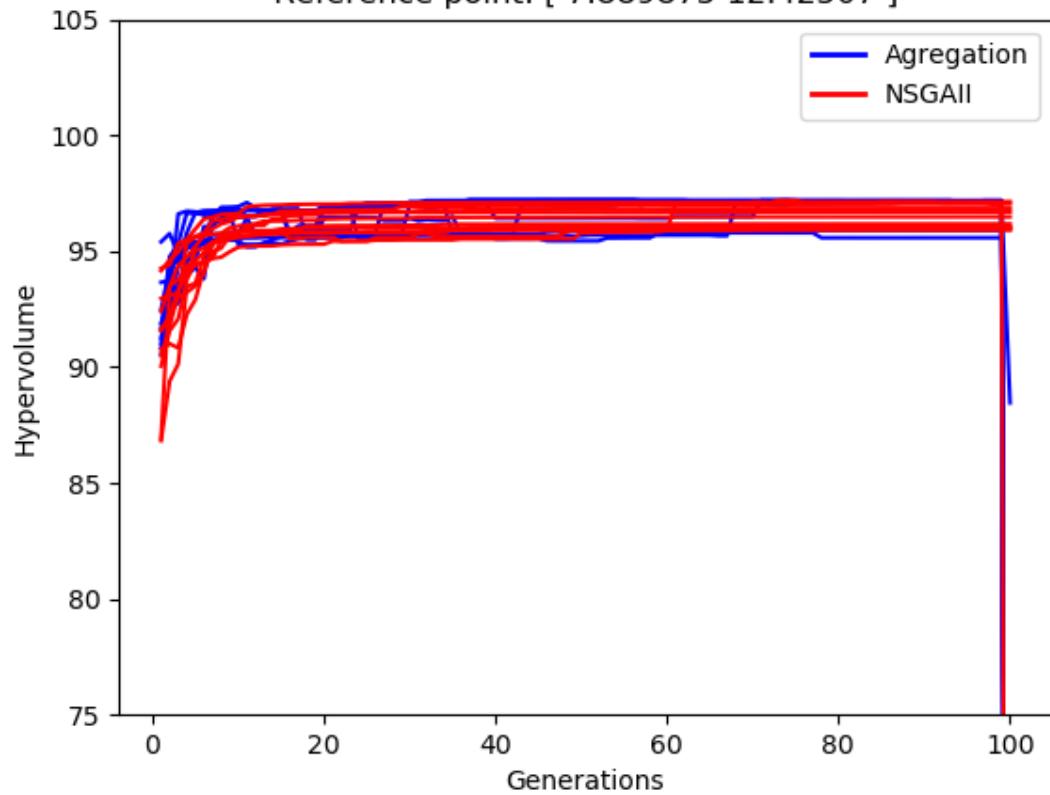


100 individuos y 100 generaciones

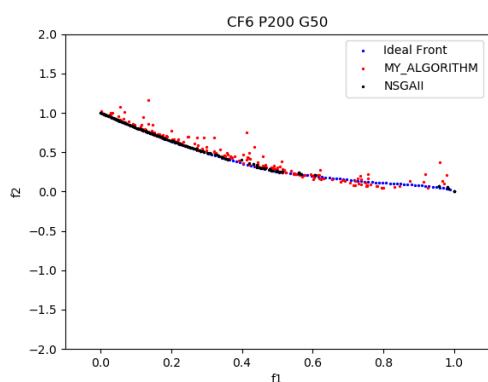
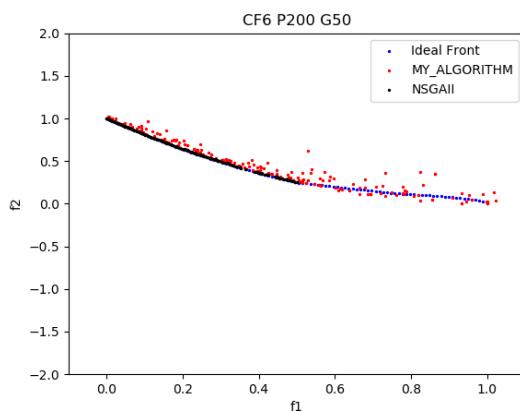
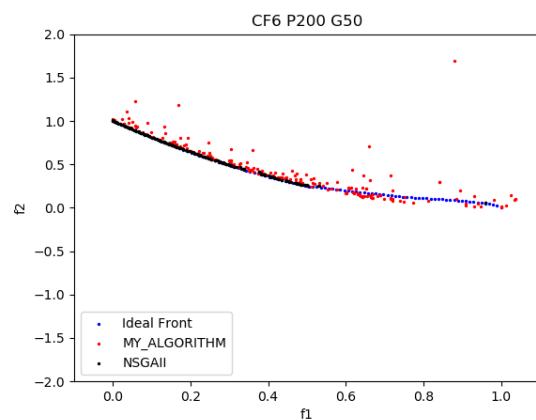
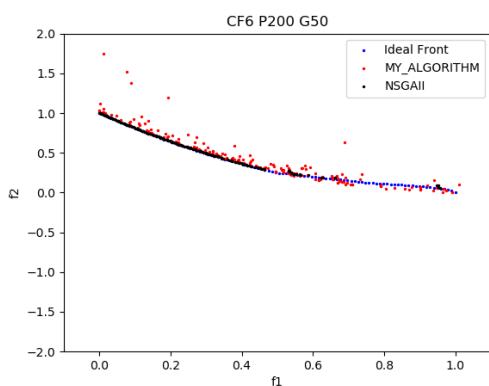
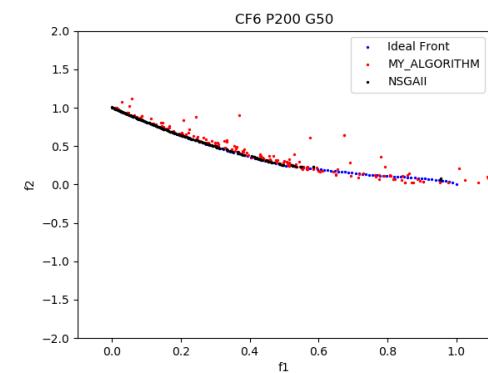
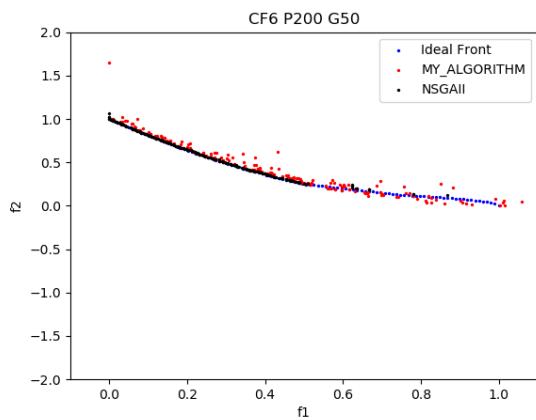
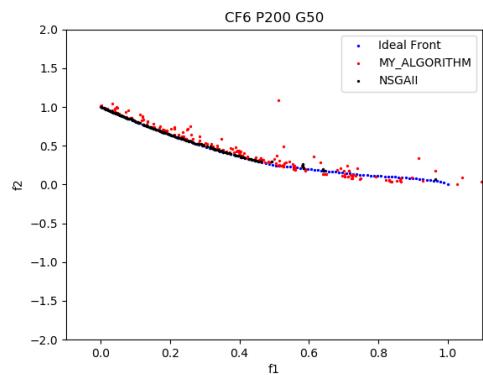
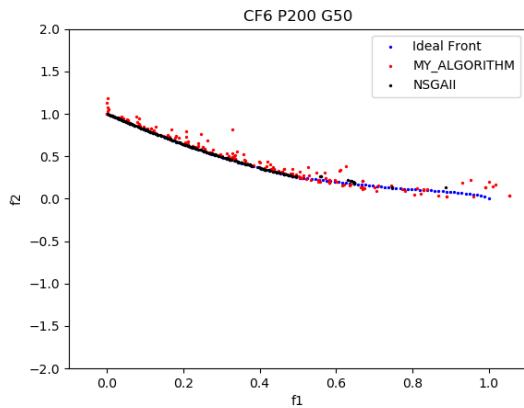


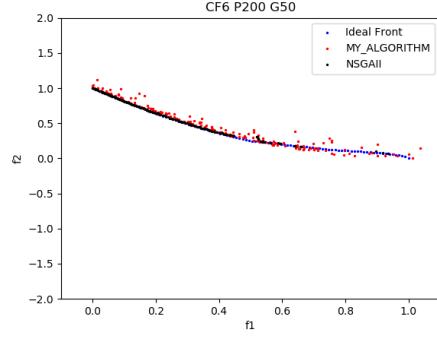
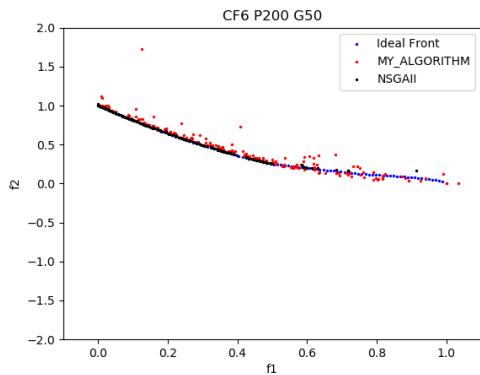


P100G100-> Mean my_alg: 91.15 Std my_alg: 29.05
Mean nsgaii: 96.42 Std nsgaii : 0.446
Reference point: [7.889875 12.42507]

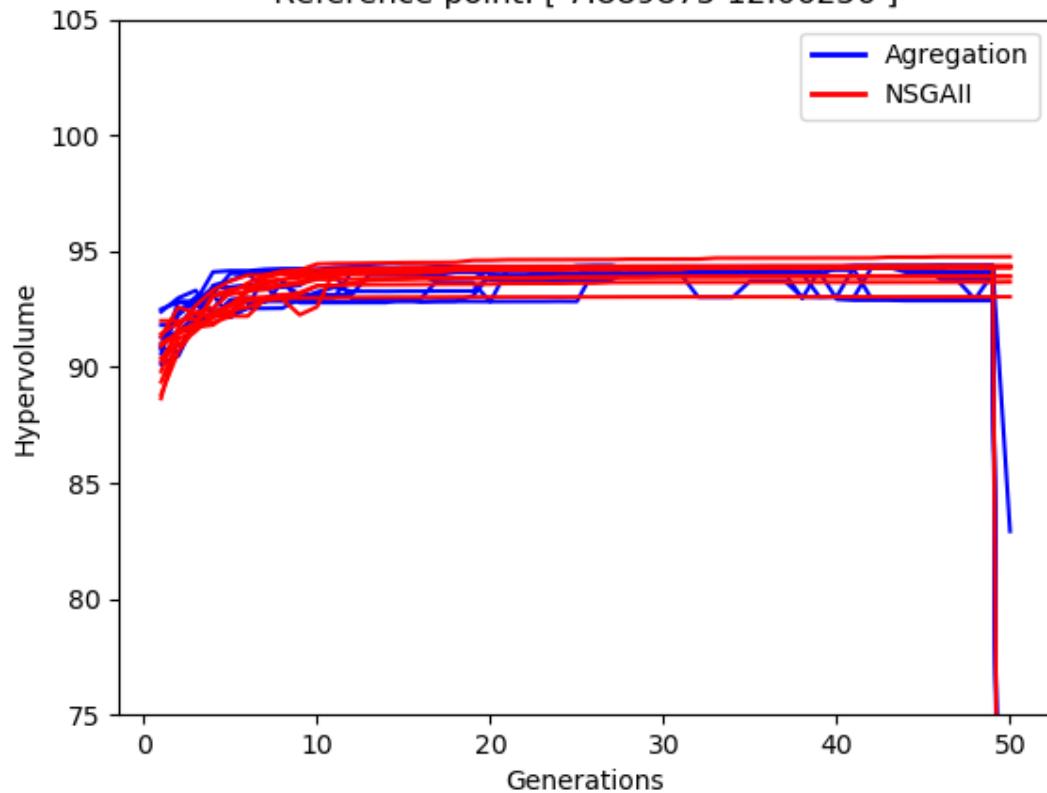


200 individuos y 50 generaciones

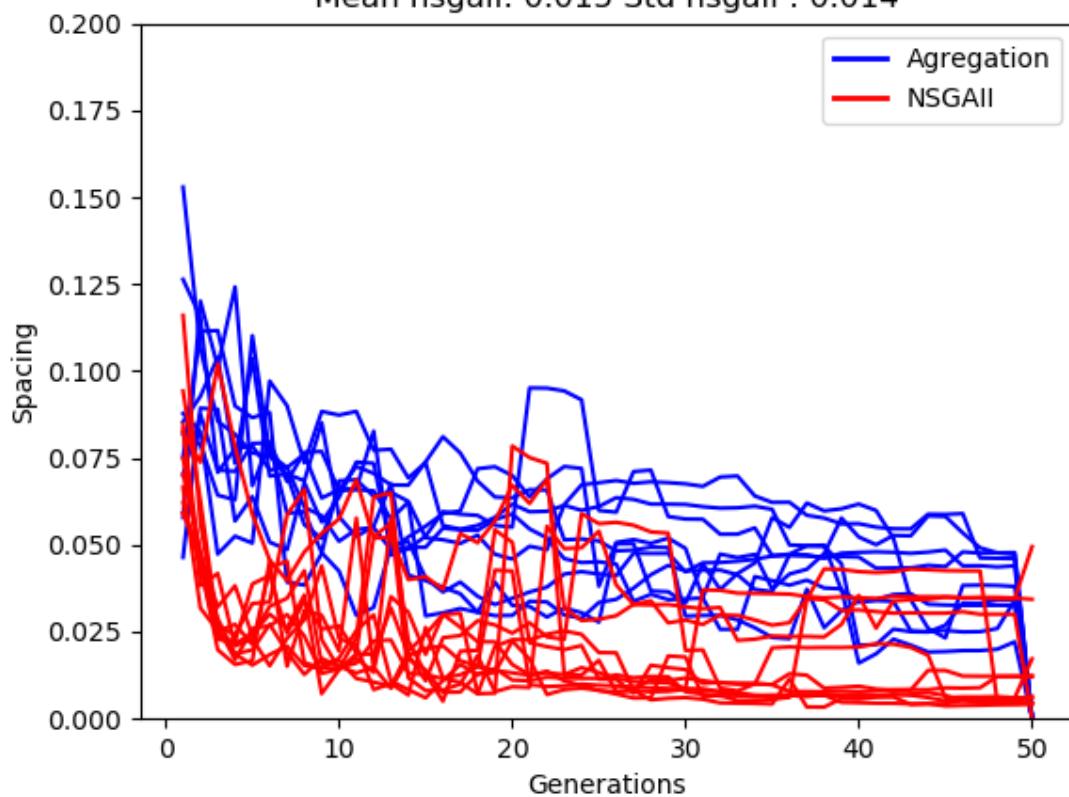




P200G50-> Mean my_alg: 88.60 Std my_alg: 28.20
Mean nsgaii: 94.09 Std nsgaii : 0.458
Reference point: [7.889875 12.06256]



P200G50-> Mean my_alg: 0.034 Std my_alg: 0.013
Mean nsgaii: 0.015 Std nsgaii : 0.014



Conclusiones

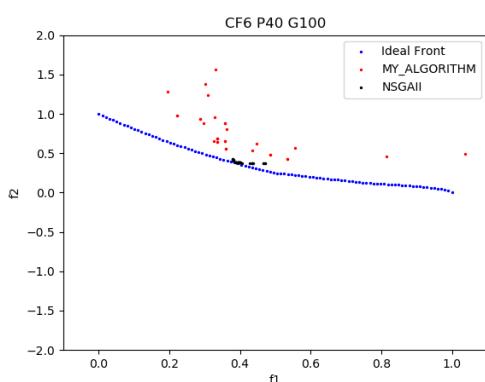
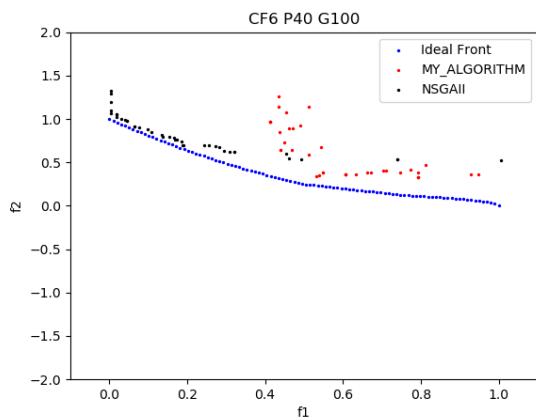
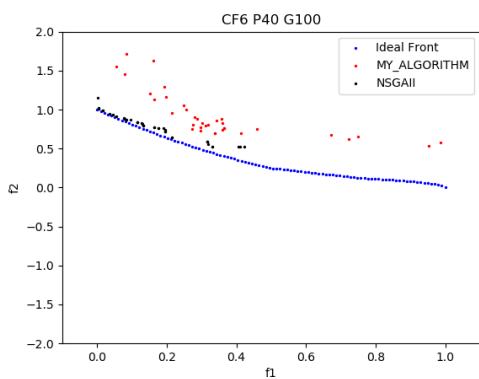
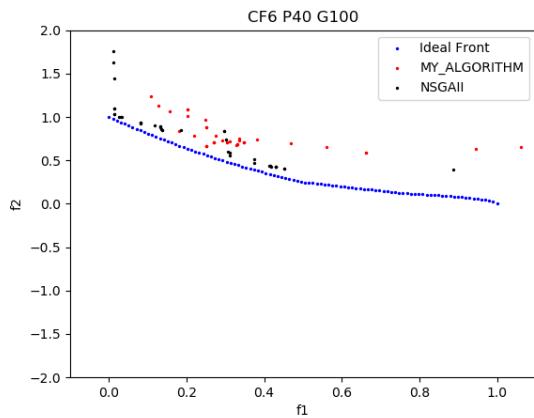
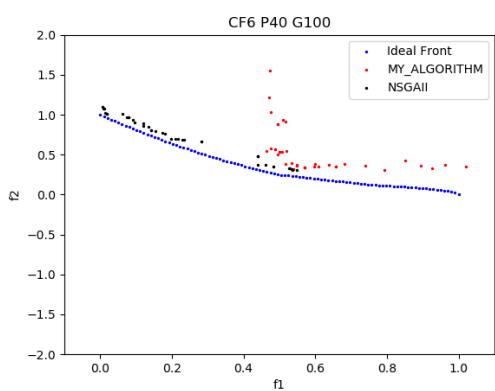
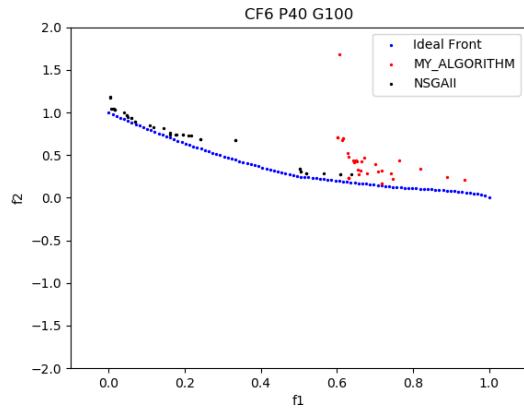
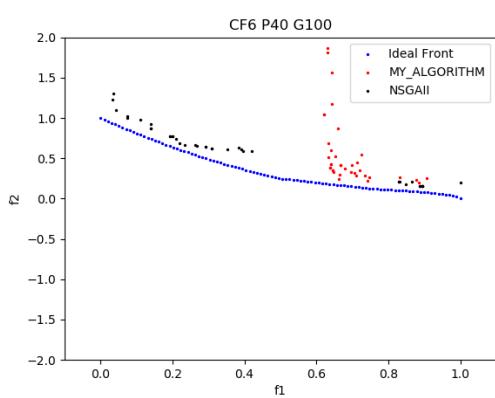
Igual que en el problema anterior, observamos a priori que tanto el algoritmo basado en agregación como el algoritmo NSGAII encuentran buenas soluciones que se acercan a una pequeña distancia al frente ideal, cumpliendo las restricciones, aunque vemos que hay individuos de agregación de la población final que no cumplen las restricciones (por debajo del frente ideal) por lo que no podemos contarlas como soluciones válidas. Aunque a diferencia de anteriormente, al ser solo 4 dimensiones, vemos que con tan solo 4000 evaluaciones nos aproximamos al frente de manera muy aproximada.

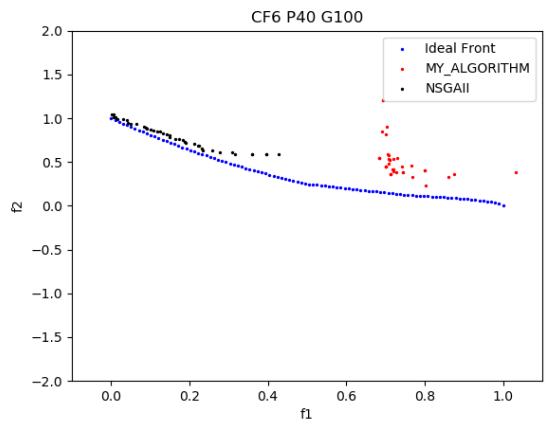
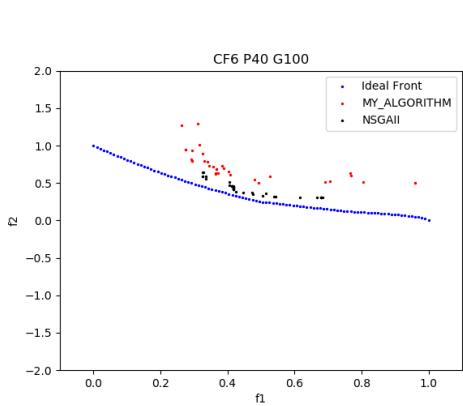
En cuanto a comparación, observamos que ambos algoritmos están muy parejos, lo cual lo podemos ver reflejado en las gráficas de hipervolumen. Aunque podemos afirmar basándonos en las medias de hipervolúmenes y en el frente que NSGAII encuentra soluciones en conjunto mejores y más aproximadas (podemos ver que está justo encima del frente mientras que agregación no).

Para terminar, podemos decir que NSGAII resuelve de mejor forma el problema CF6 para 4 dimensiones, aunque ambos algoritmos encuentran muy buenas soluciones en general.

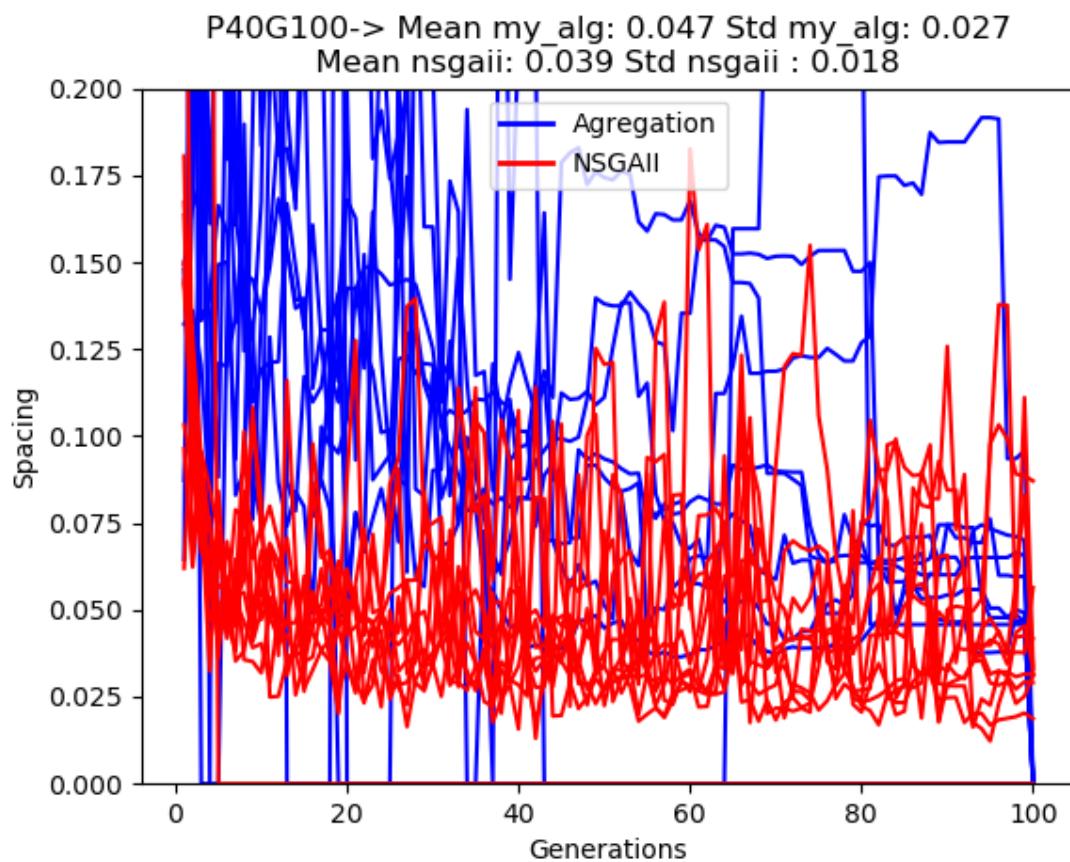
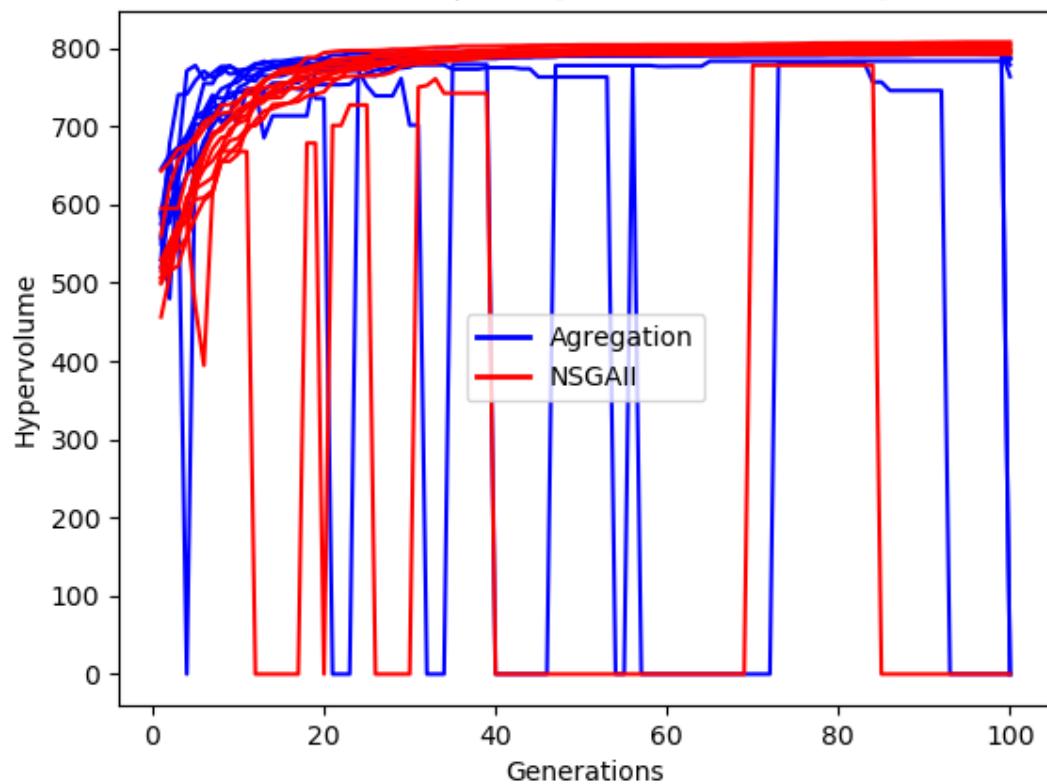
Problema CF6 en 16 dimensiones

40 individuos y 100 generaciones

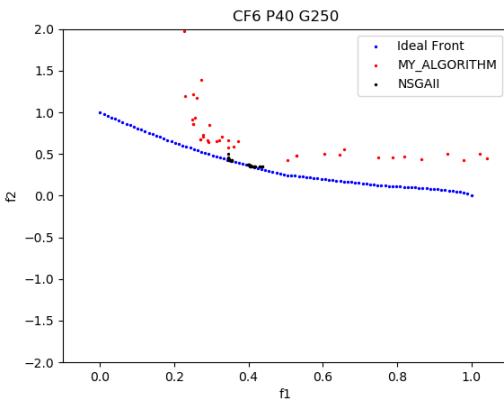
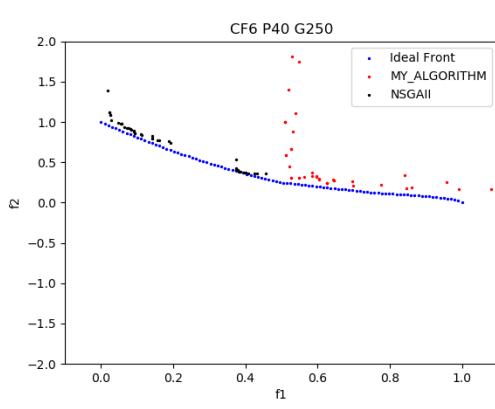
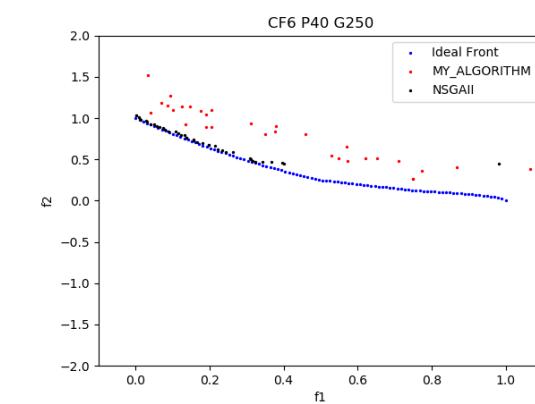
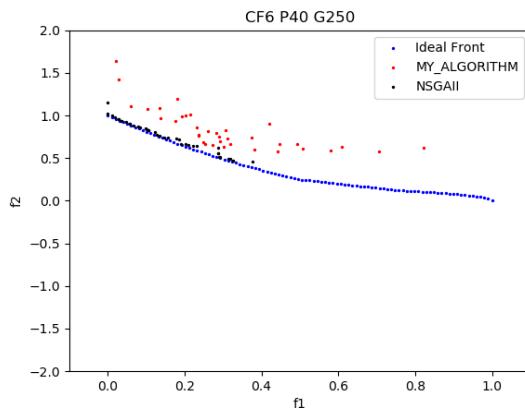
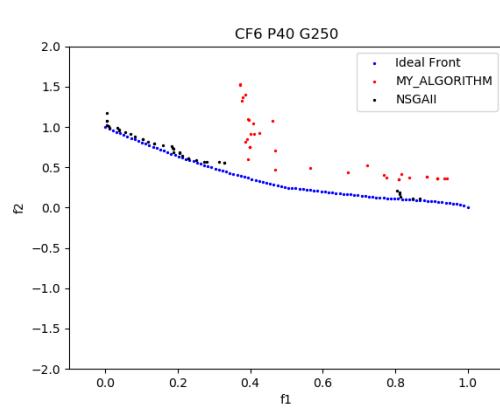
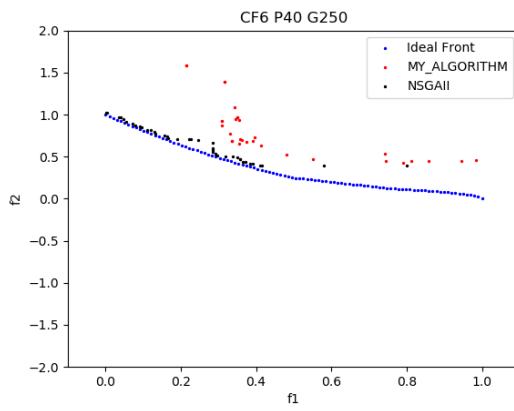
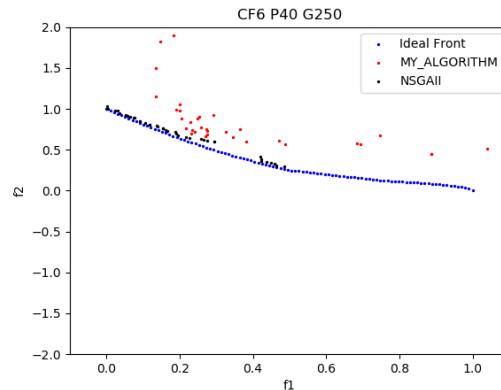
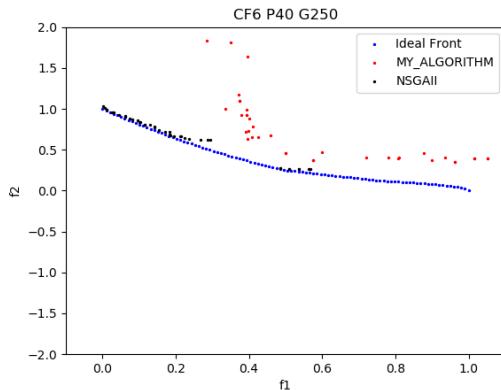


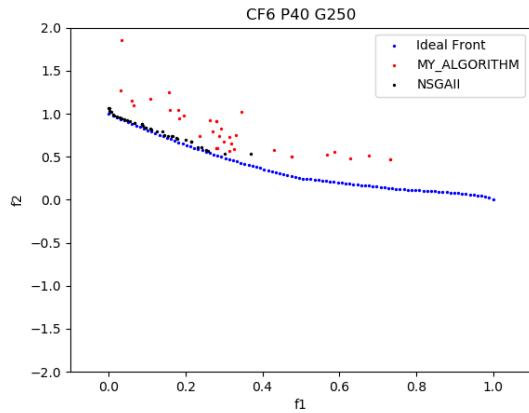
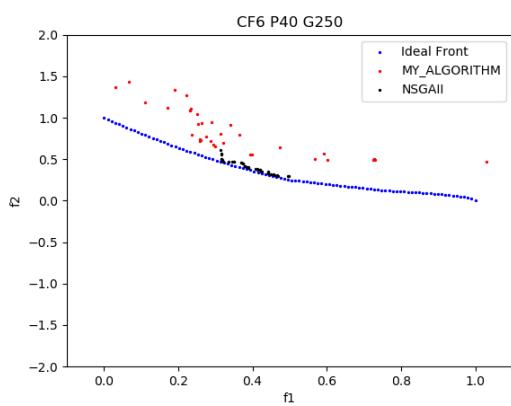


P40G100-> Mean my_alg: 712.3 Std my_alg: 236.5
Mean nsgaii: 799.3 Std nsgaii : 4.938
Reference point: [29.77245 27.33495]

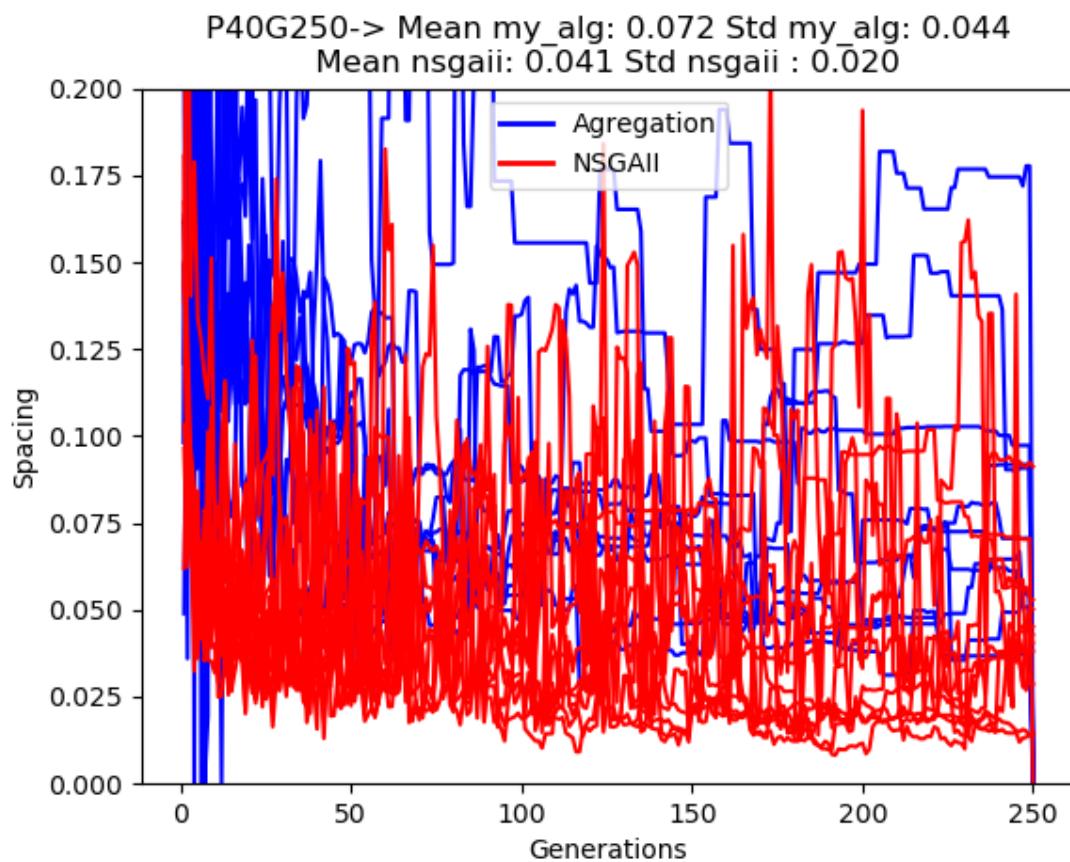
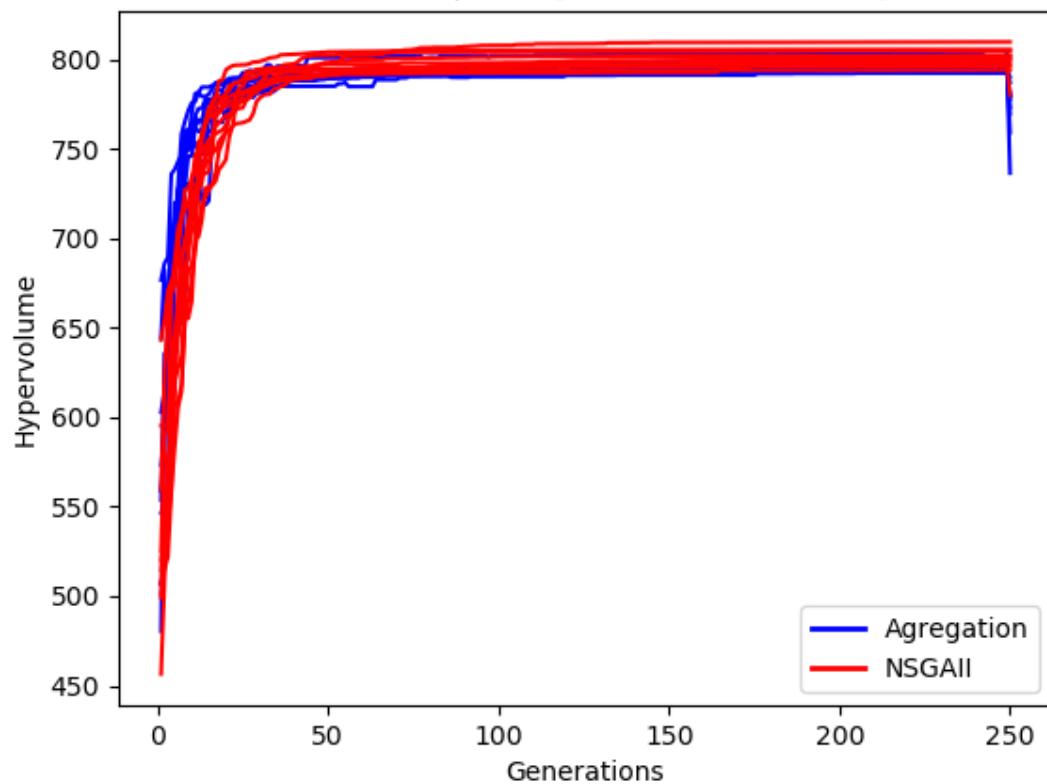


40 individuos y 250 generaciones

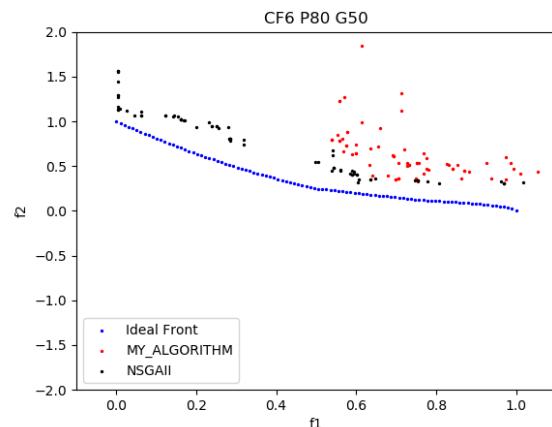
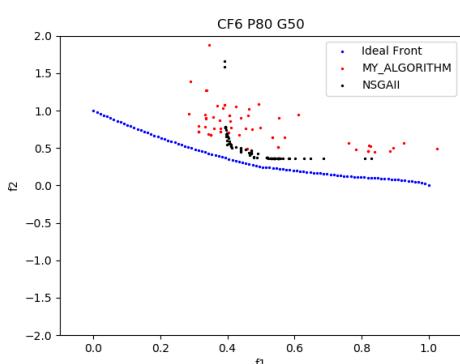
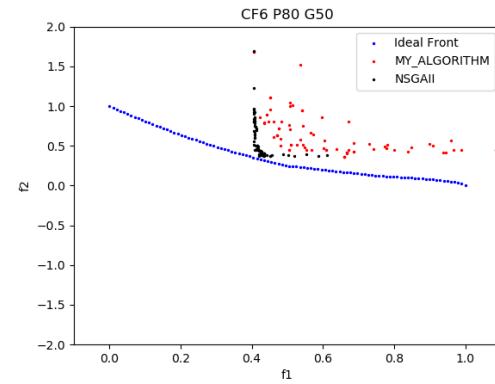
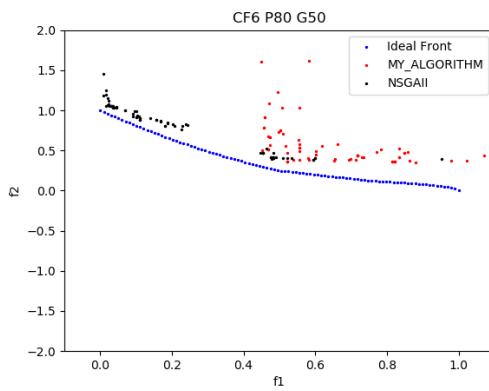
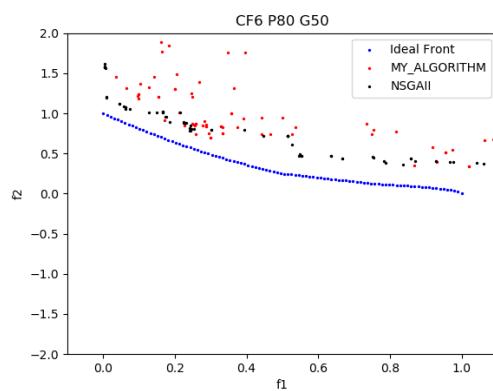
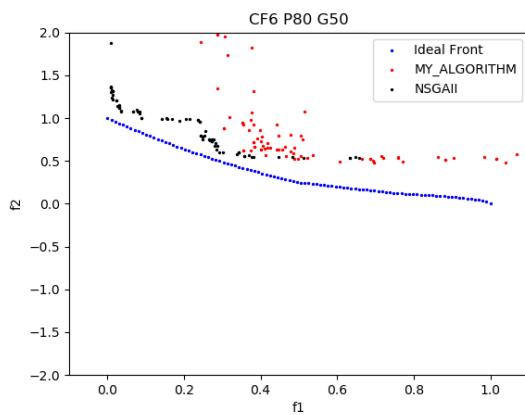
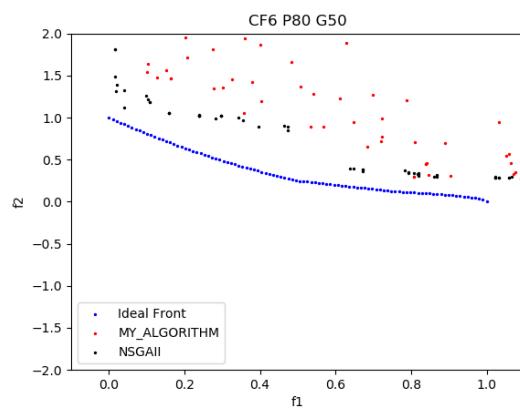
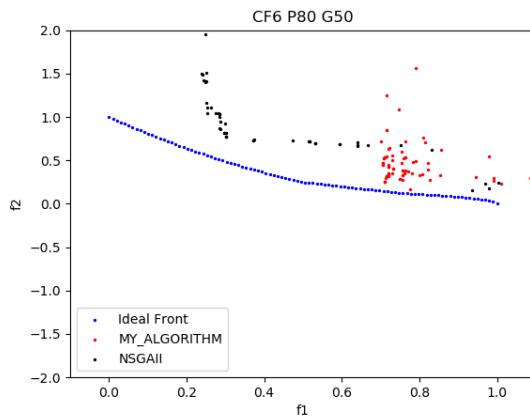


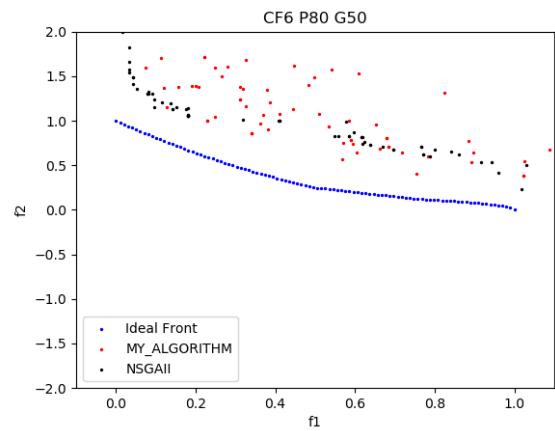
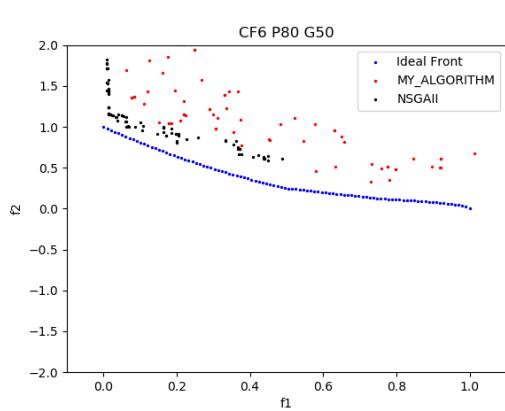


P40G250-> Mean my_alg: 721.2 Std my_alg: 239.1
Mean nsgaii: 801.3 Std nsgaii : 4.444
Reference point: [29.77245 27.33495]

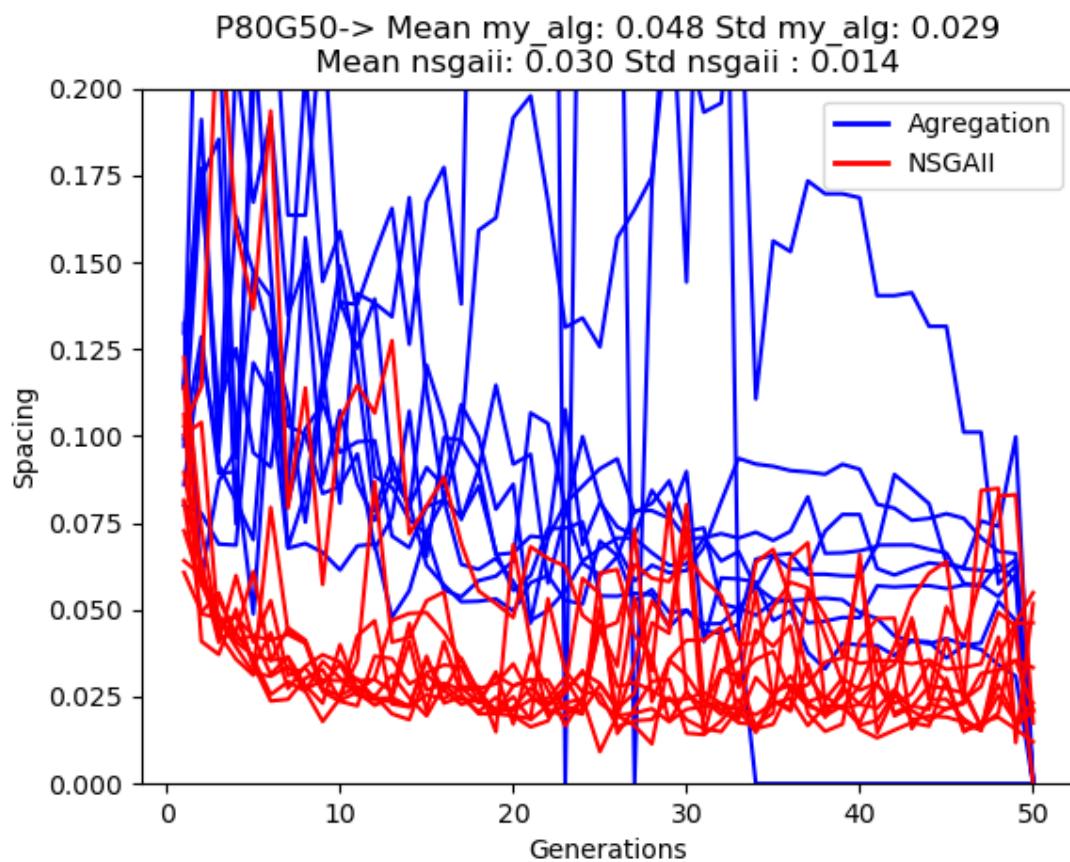
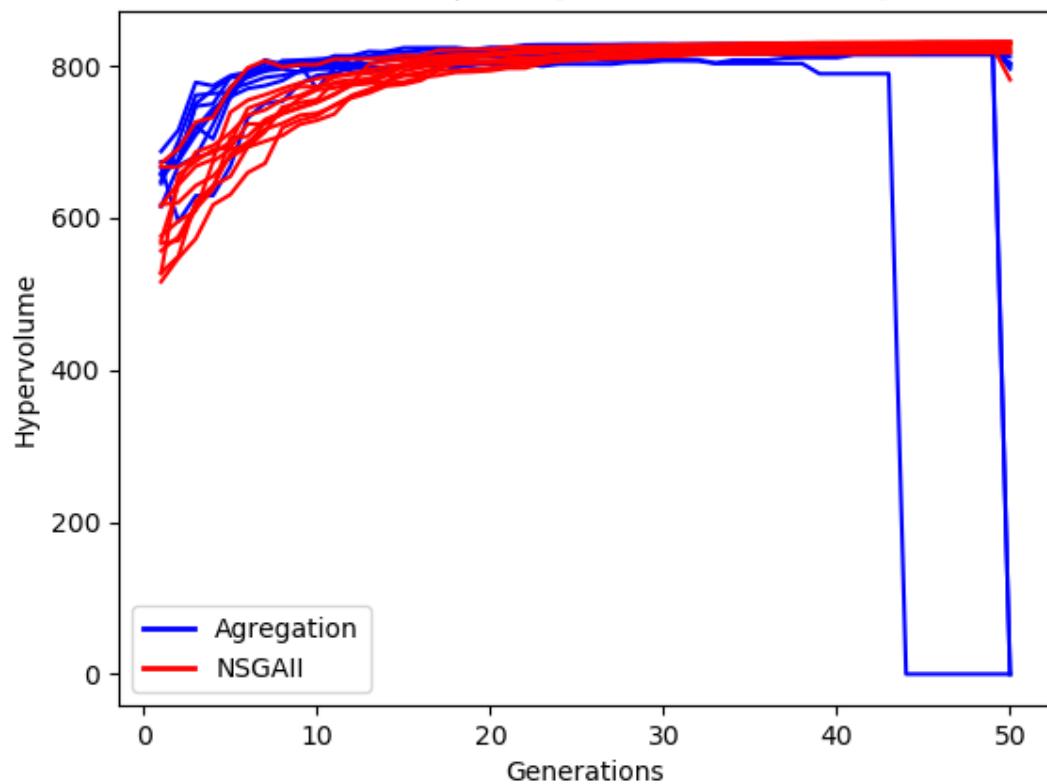


80 individuos y 50 generaciones

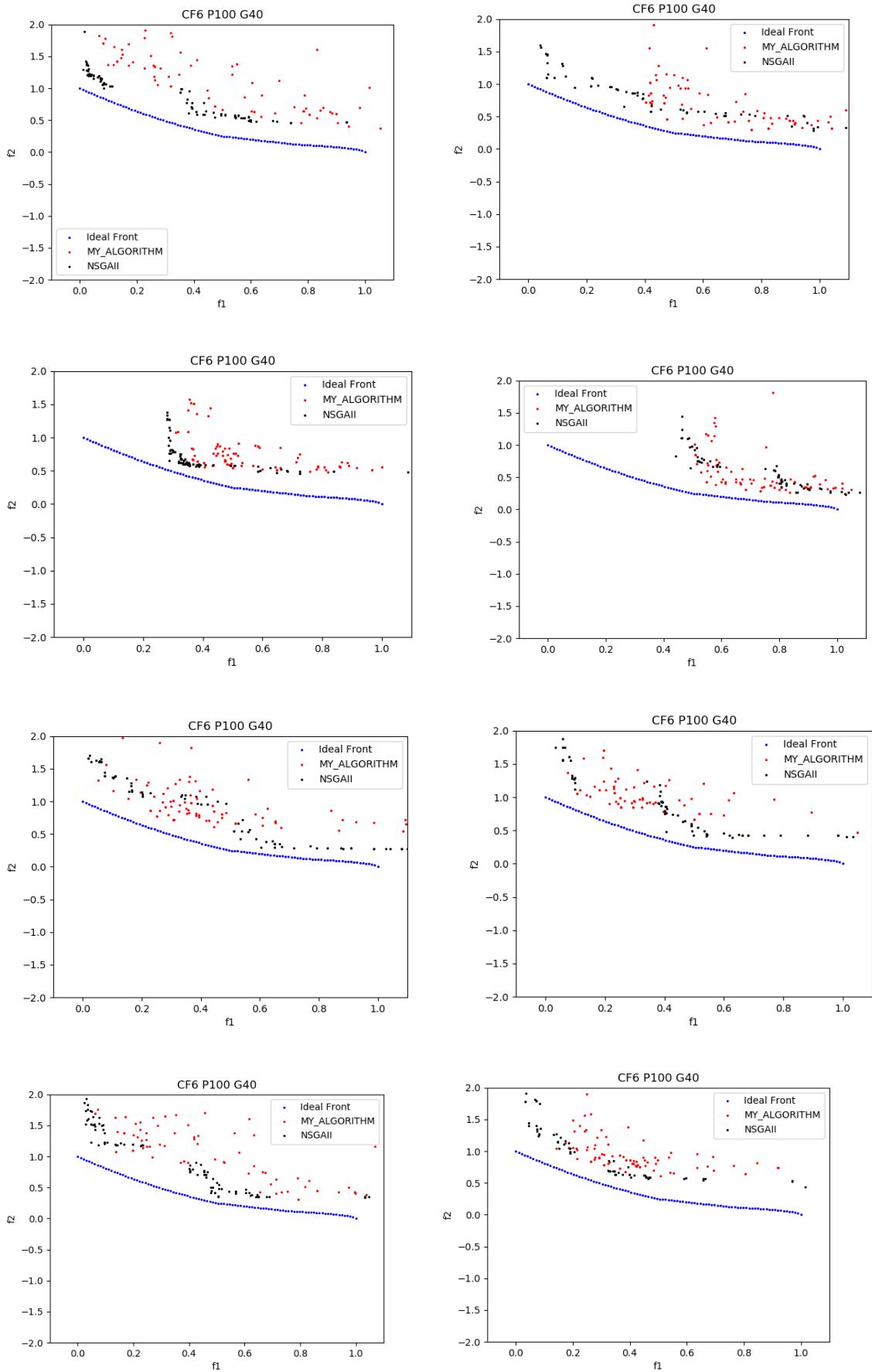


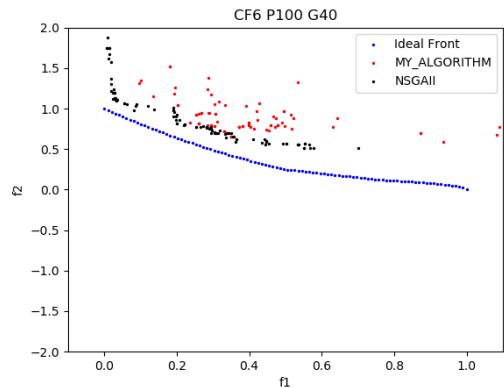
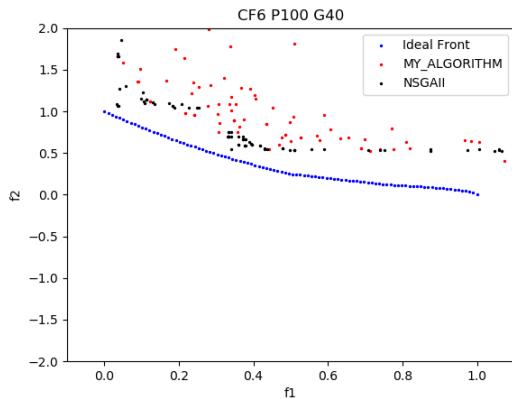


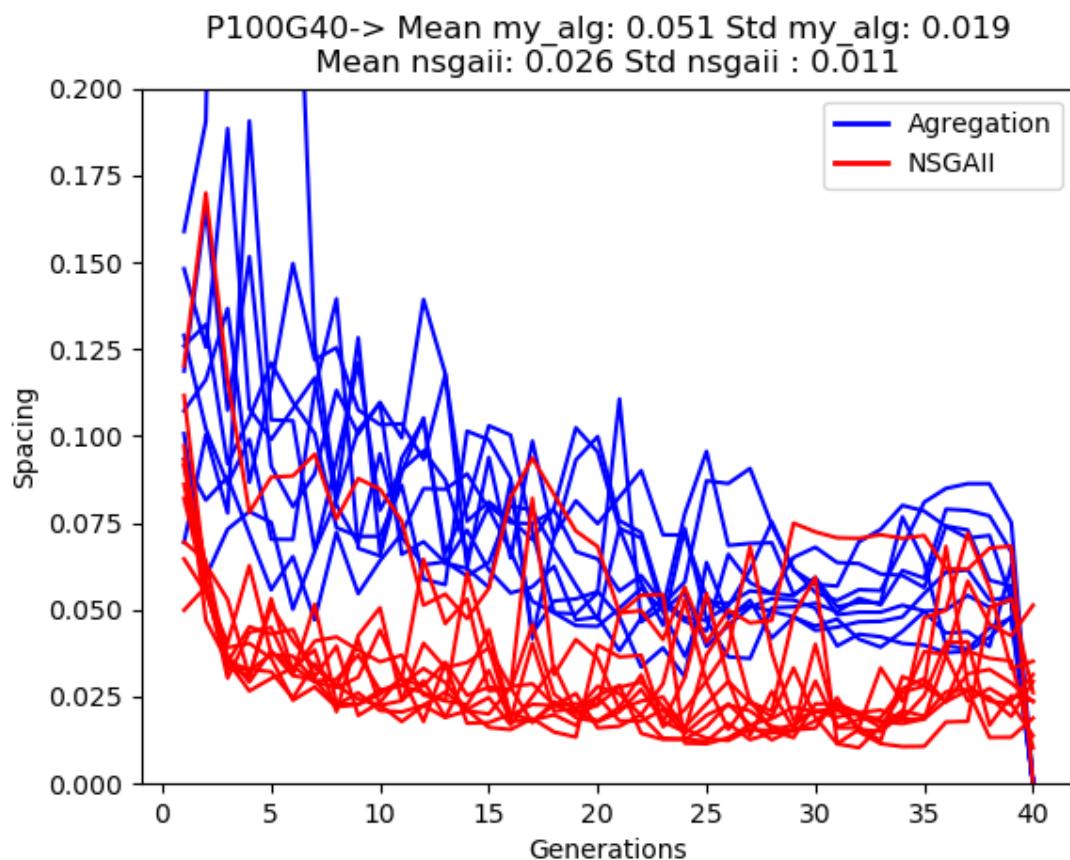
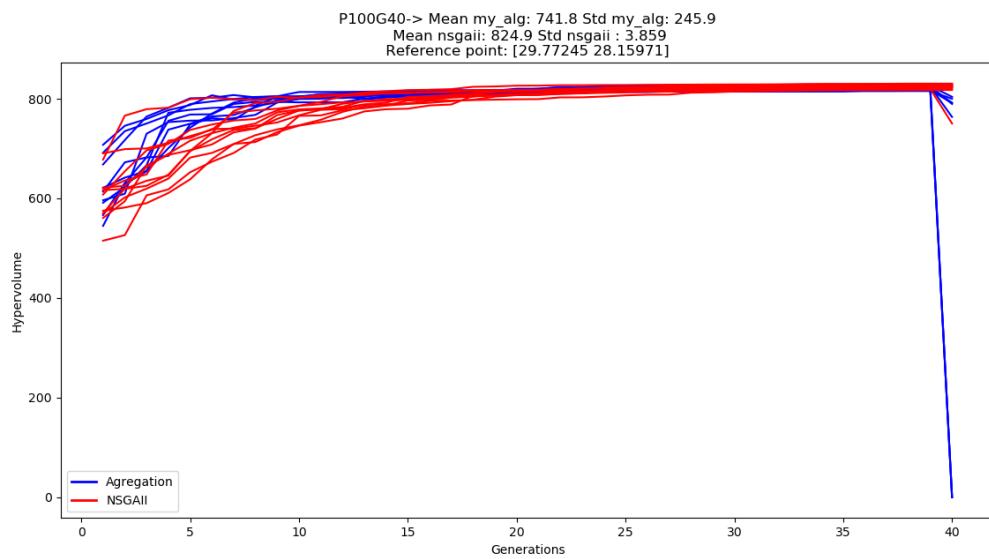
P80G50-> Mean my_alg: 739.5 Std my_alg: 245.3
Mean nsgaii: 826.1 Std nsgaii : 5.238
Reference point: [29.77245 28.20557]



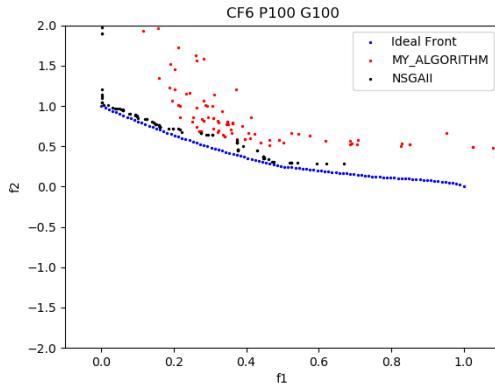
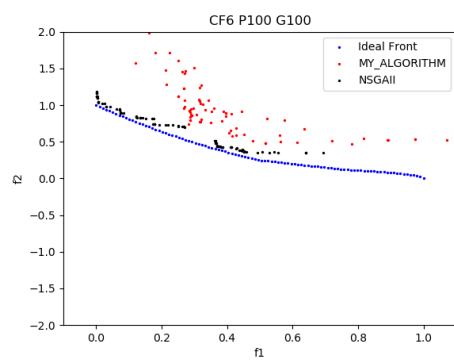
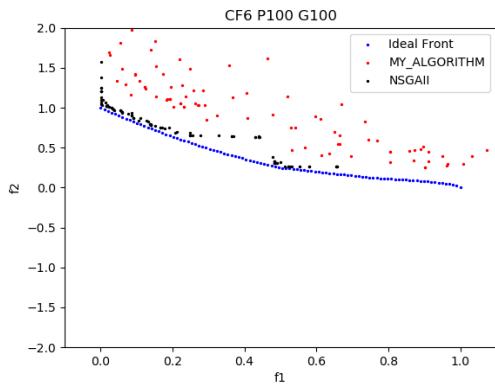
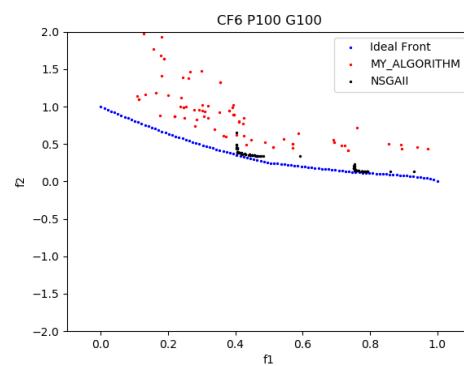
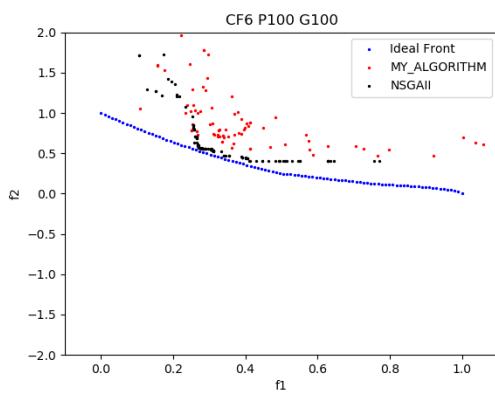
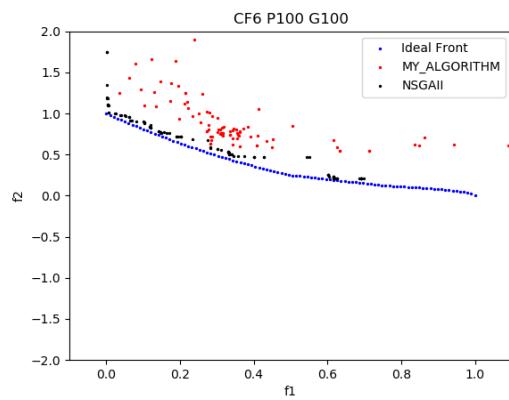
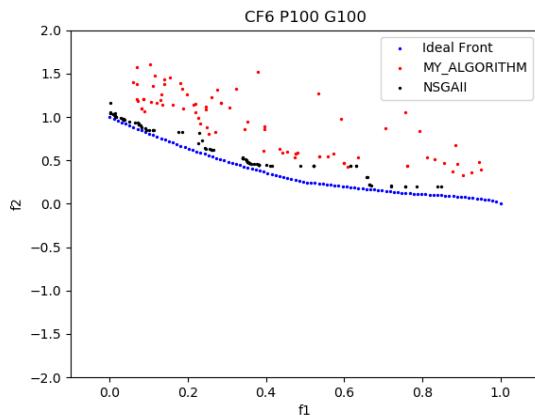
100 individuos y 40 generaciones

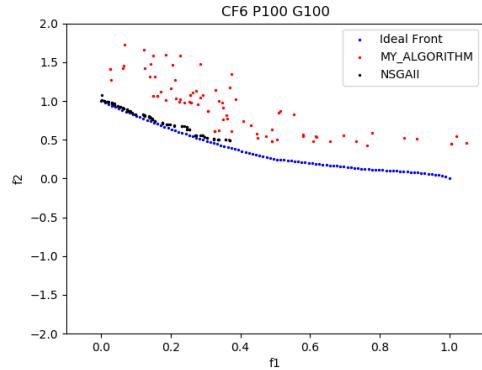
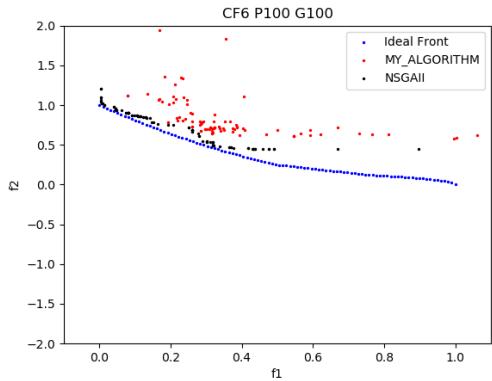




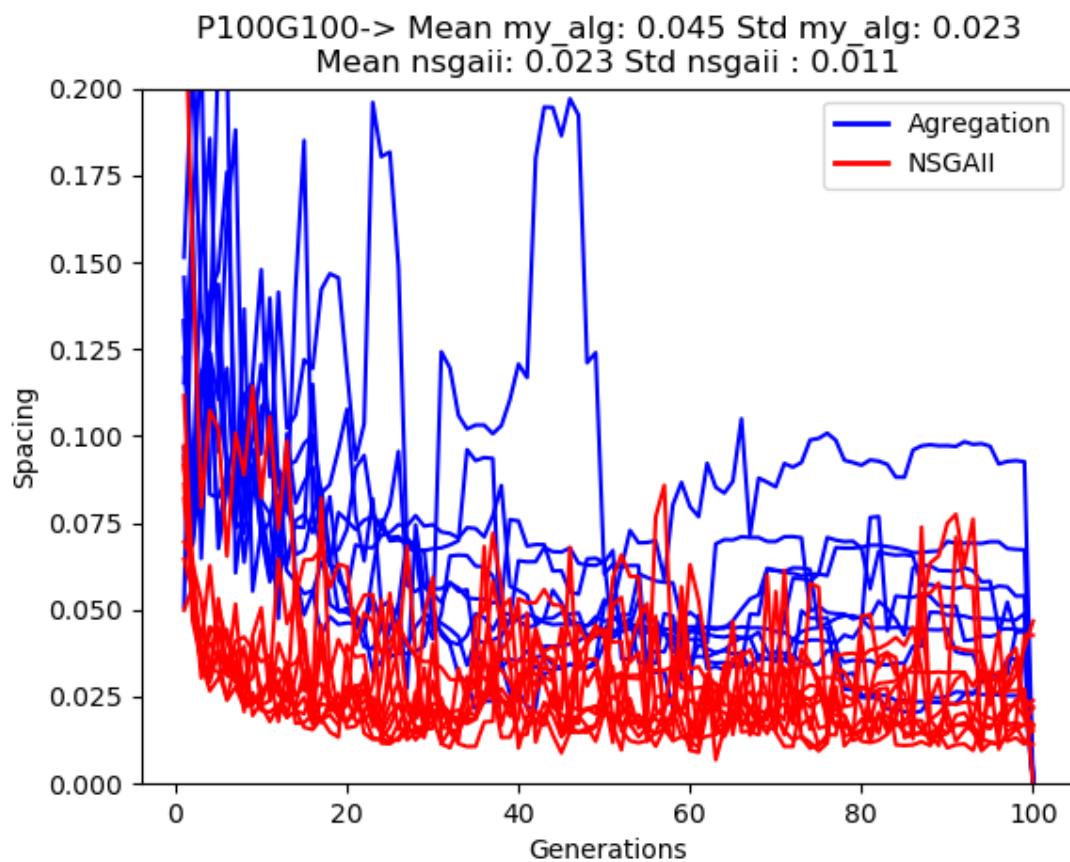
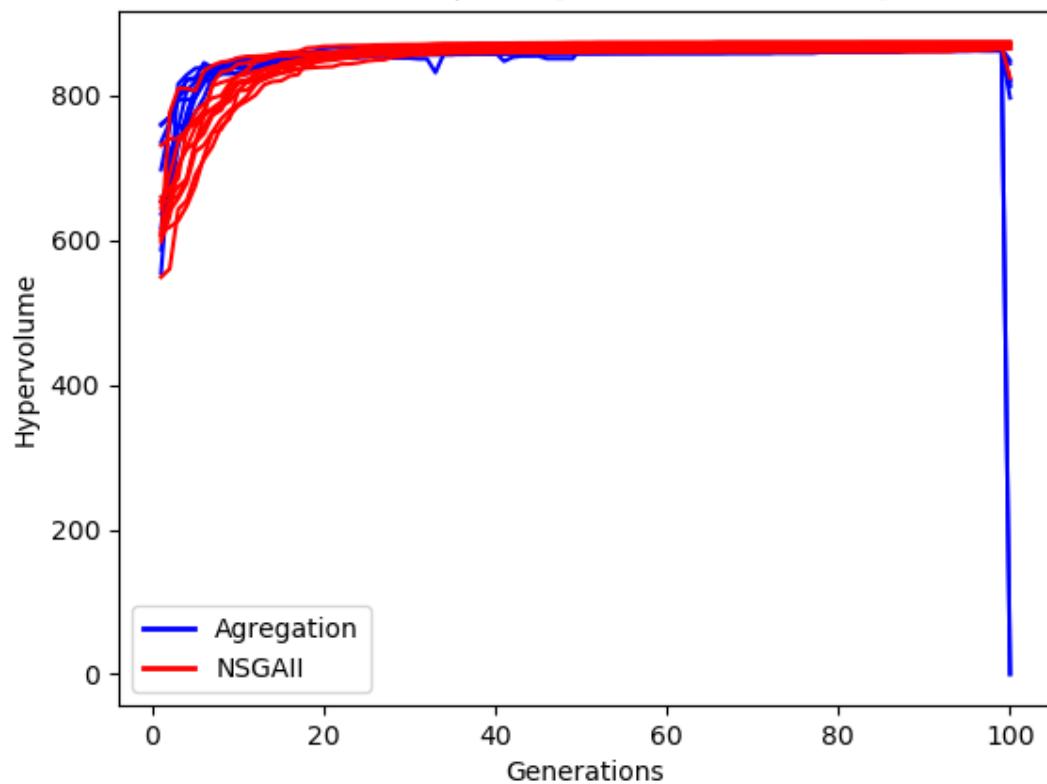


100 individuos y 100 generaciones

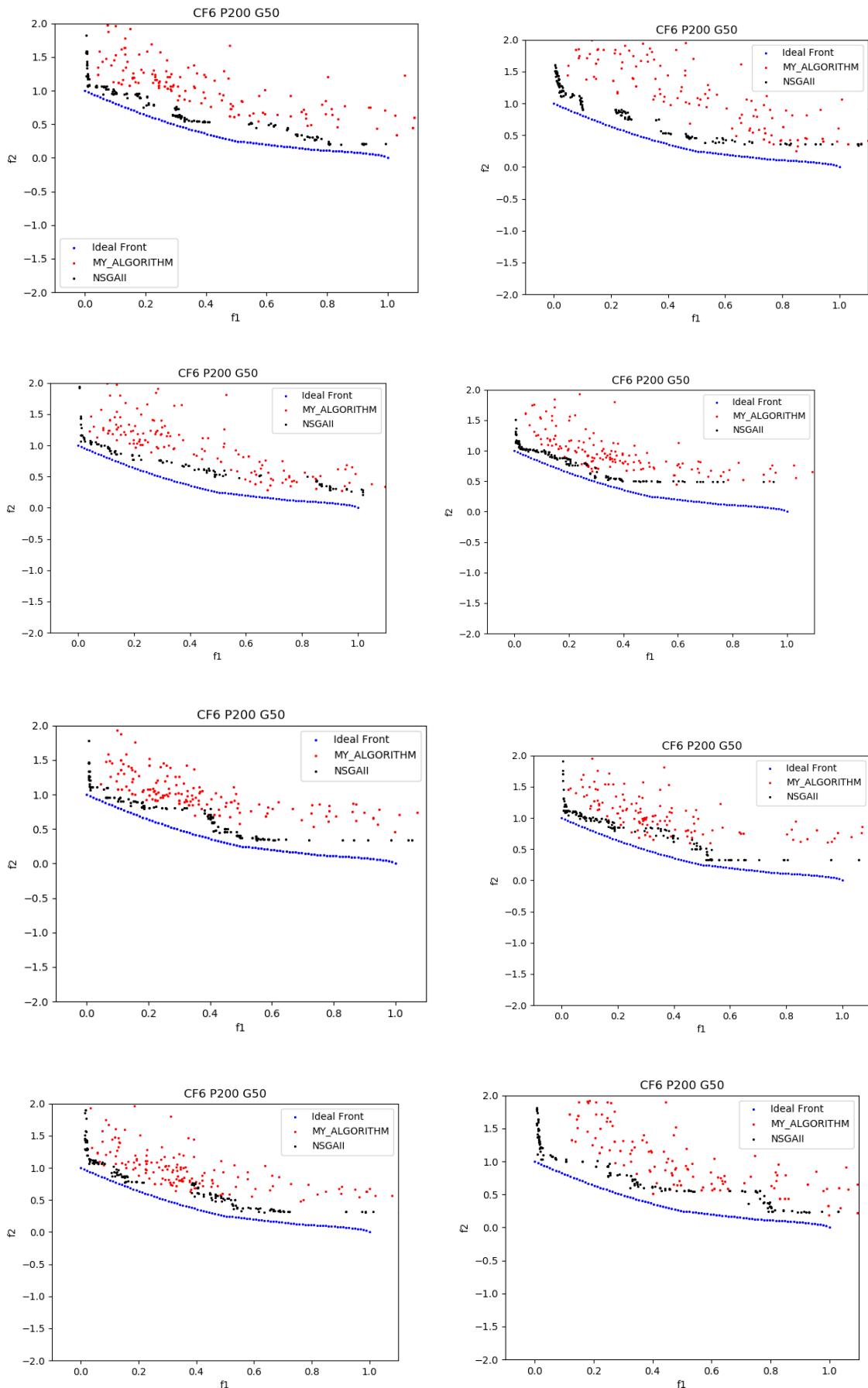


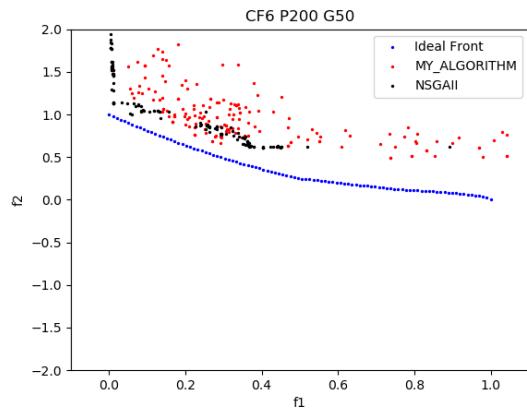
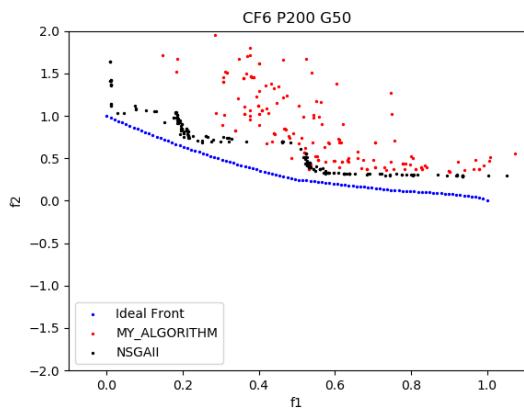


P100G100-> Mean my_alg: 782.8 Std my_alg: 259.6
Mean nsgaii: 869.1 Std nsgaii : 3.456
Reference point: [29.77245 29.56026]

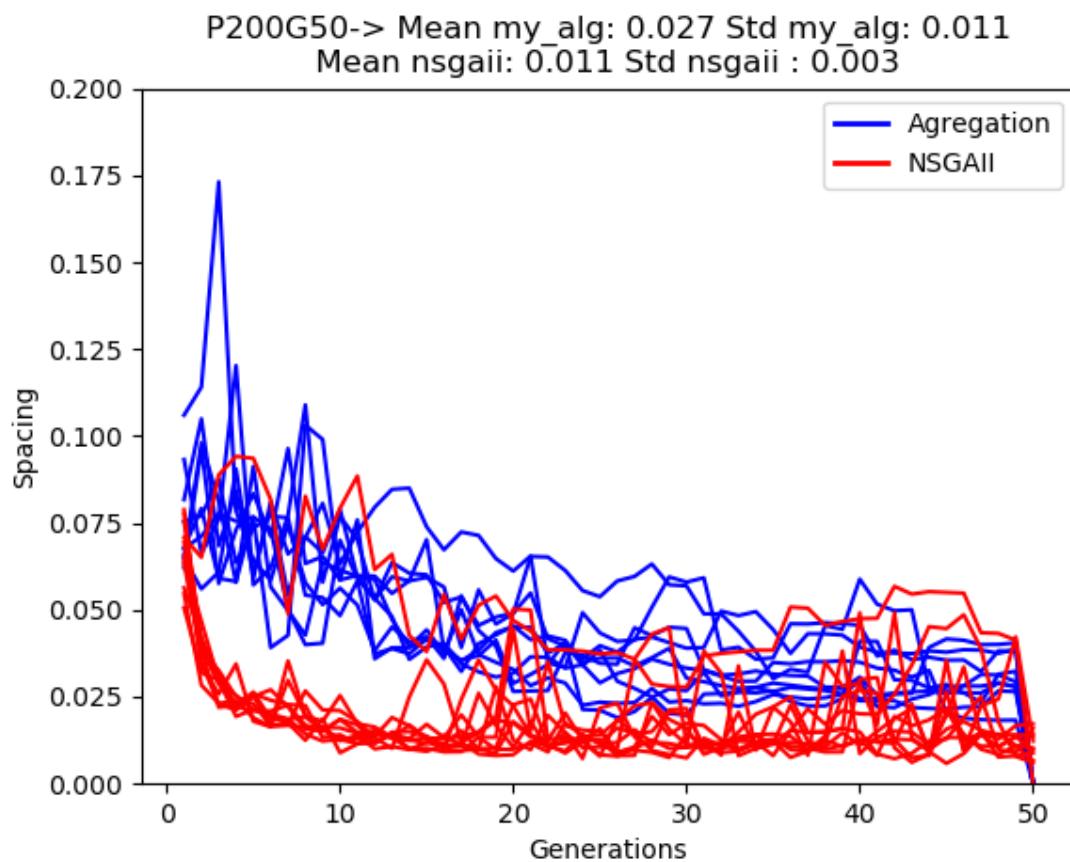
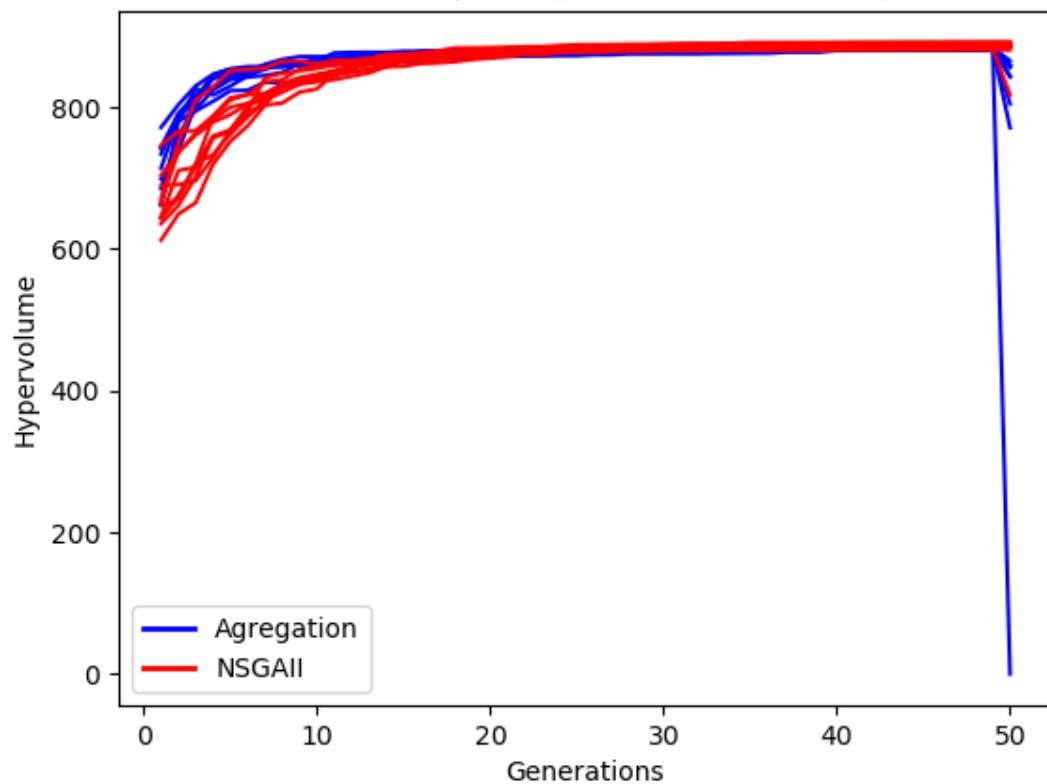


200 individuos y 50 generaciones





P200G50-> Mean my_alg: 797.7 Std my_alg: 264.5
Mean nsgaii: 887.2 Std nsgaii : 3.172
Reference point: [29.77245 30.08797]



Conclusiones

En esta variación del problema para 16 dimensiones ya notamos que ambos algoritmos les resulta complicado encontrar un buen conjunto de soluciones, ya no se llegan a aproximar tanto como lo hacían para 4 dimensiones (sobre todo para 4000 evaluaciones), algo evidente ya que añadir 12 dimensiones aumenta de manera muy grande el espacio de búsqueda de soluciones.

En cuanto a comparación, observamos que para 4000 evaluaciones aunque ninguno se aproxime totalmente al frente, NSGAII encuentra un conjunto de soluciones mejores tal y como demuestran las gráficas de hipervolúmenes (media de hipervolumen es más alta en general). Para 10000 evaluaciones ya observamos que sobre todo con poblaciones grandes NSGAII consigue aproximarse bastante bien al frente ideal, mientras que agregación no nota mucho el aumento de población o generaciones. En ambos casos vemos que un mejor spacing de NSGAII lo cual se refleja en los frentes en que están mejor repartidas las soluciones por el frente.

Para terminar, podemos decir que NSGAII resuelve de mejor forma el problema CF6 para 16 dimensiones, aunque ambos algoritmos les cuesta encontrar buenas soluciones.

3. Conclusión final

Tras visualizar y analizar todas las ejecuciones para distintas combinaciones de poblaciones y generaciones y también distintos parámetros, podemos sacar las siguientes conclusiones.

Un punto a favor para mi algoritmo es que parece comenzar mejor la búsqueda de soluciones (en la mayoría de gráficas de hipervolúmenes vemos que en generaciones tempranas el algoritmo de agregación supera a NSGAII).

Por el contrario, en general NSGAII encuentra mejores conjuntos de soluciones que están más aproximadas al frente ideal y mejor equiespaciadas.

Tras ello, podemos entonces concluir que NSGAII resuelve en general mejor los problemas que mi algoritmo de agregación implementado.