Tarea 10 - Optimización estocástica

Erick Salvador Alvarez Valencia

CIMAT A.C.,
erick.alvarez@cimat.mx

Resumen En el presente reporte se hablará sobre los resultados obtenidos en la tarea 10 de optimización estocástica, en donde se analiza el comportamiento del algoritmo CMA-ES mediante una implementación conseguida en internet, además se hablará sobre los resultados obtenidos. Posteriormente se hará una comparación estadística con una técnica basada en remuestreo a los algoritmos AG y ED.
Para finalizar se darán algunas conclusiones sobre los ejercicios trabajados.

Keywords: CMA-ES, Evolución diferencial.

1. Ejercicio 1 - Implementación del CMA-ES y realización de pruebas con varias funciones

Para esta primera parte de la tarea se consiguió la implementación del algoritmo CMA-ES para el lenguaje C++. Dicha implementación fue descargada del siguiente sitio web *https://github.com/AlexanderFabisch/CMA-ESpp* posteriormente fue adaptada al código de la tarea y los parámetros fueron configurados para que trabajara con las funciones que se han venido probando en las tareas anteriores.
Una vez configurado el algoritmo se hicieron pruebas con dichas funciones y usando la siguiente configuración:

1. **Dimensiones** : 10 y 30.
2. **Número de evaluaciones de la función** : 150.
3. **Sigma** : 0.5.
4. **Número total de ejecuciones del algoritmo** : 30
5. **Tamaño de la población** : 20.

Las funciones a optimizar son las siguientes:

1. **Sphere** : $\sum_{i=1}^{d} x_i^2$
2. **Ellipsoid** : $\sum_{i=1}^{d} 10^{6\frac{i-1}{d-1}} x_i^2$
3. **Zakharov** : $\sum_{i=1}^{d} x_i^2 + (\sum_{i=1}^{d} 0{,}5ix_i^2)^2 + (\sum_{i=1}^{d} 0{,}5ix_i^2)^4$
4. **Rosenbrock** : $\sum_{i=1}^{d-1}[(1-x_i)^2 + 100(x_{i+1}-x_i^2)^2]$
5. **Ackley** : $-20\exp{(-0{,}2\sqrt{\frac{1}{d}\sum_{i=1}^{d} x_i^2})} - \exp{(\frac{1}{d}\sum_{i=1}^{d} cos(2\pi x_i))} + 20 + e$

6. **Griewangk** : $\sum_{i=1}^{d} \frac{x_i^2}{4000} - \prod_{i=1}^{d} cos(\frac{x_i}{\sqrt{i}}) + 1$
7. **Rastrigin** : $10d + \sum_{i=1}^{d}[x_i^2 - 10cos(2\pi x_i)]$

A continuación se muestran las tablas con los resultados obtenidos de ambas series de ejecuciones.

Cuadro 1: Resultados del CMA-ES con dimensión 10

Función	Promedio del error	Desviación estándar del error	Mediana del error	Mejor aptitud	Peor aptitud
Sphere	1.26791e-315	0	4.64916e-320	0	2.22507e-308
Ellipsoid	3.96834e-293	0	6.95111e-303	3.79573e-310	1.19049e-291
Zakharov	4.72624e-318	1.1189e-15	4.72624e-317	4.94066e-324	2.22507e-308
Rosenbrock	0.66443	1.48571	4.39488e-28	7.14782e-29	3.98658
Ackley	7.43109e-15	1.94229e-15	7.54952e-15	3.9968e-15	1.46549e-14
Griewangk	0.00870346	0.0088951	0.00862666	0	0.0320134
Rastrigin	13.6973	4.43237	13.4319	5.96975	21.8891

Cuadro 2: Resultados del CMA-ES con dimensión 30

Función	Promedio del error	Desviación estándar del error	Mediana del error	Mejor aptitud	Peor aptitud
Sphere	0	0	0	0	2.22507e-308
Ellipsoid	1.38038e-168	0	1.70977e-170	7.02555e-174	3.30409e-167
Zakharov	1.023e-216	0	2.1548e-224	6.78616e-229	2.18348e-215
Rosenbrock	0.132887	0.715621	4.21269e-27	1.78115e-27	3.98662
Ackley	1.51286e-14	2.71084e-15	1.46549e-14	7.54952e-15	2.53131e-14
Griewangk	0.00180713	0.00411995	0	0	0.012321
Rastrigin	52.3016	15.2603	49.2504	24.874	106.46

En los dos cuadros anteriores se aprecian los resultados obtenidos con el algoritmo CMA-ES para las funciones anteriormente mencionadas. Se puede notar que en general se obtuvieron buenos resultados aunque en las funciones de Rosenbrock y Rastrigin no tanto, ya que el error promedio es del orden de 10^{-1} en el caso de Rosenbrock y de 10^1 en el caso de Rastrigin, en esta última no se pudo llegar al óptimo global ninguna vez, tanto en dimensión 10 como en 30.
Cabe mencionar que se probó este algoritmo con diferentes parámetros tanto del tamaño de la población como de valor de sigma inicial para la matriz de covarianza y en ninguno de ellos se pudo llegar al óptimo de la función Rastrigin, esto puede ser debido a la implementación hecha por la persona o porque esa función es dificil para dicho algoritmo.

2. Ejercicio 2

Para este ejercicio se hizo una comparación estadística del algoritmo genético y de la evolución diferencial para determinar si ambos algoritmos eran ïguales”, es decir, producian los mismos resultados para las mismas funciones o no.
El test estadístico se basó en una técnica que realiza una remuestreo de un

conjunto finito de datos en donde la cardinalidad de dicho conjunto es muy pequeña. Esto para al final poder generar un intervalo de confianza y dar un resultado para la prueba de hipótesis en donde H_0 asume que ambos algoritmos son iguales y H_1 lo contrario. El algoritmo de dicha técnica de remuestreo es el siguiente:

Algorithm 1 Comparación estadística.

procedure BOOTSTRAPSTATISTICALCOMP(X_0, X_1, n, α)
    $\theta \leftarrow$ abs(mean(X_0) - mean(X_1)).
    C $\leftarrow \{X_0, X_1\}$.
    M $\leftarrow$ 5000.
    $\phi \leftarrow \{\}$.
    for i $\leftarrow$ 1 to M **do**
        d $\leftarrow$ Muestreo con reemplazo de tamaño t de C.
        $K_1 \leftarrow [d_1, d_2, ..., d_{|X_0|}]^T$.
        $K_2 \leftarrow [d_{|X_0+1|}, d_{|X_0+2|}, ..., d_{|C|}]^T$.
        $\phi \leftarrow \phi + \{abs(mean(K_1) - mean(K_2))\}$.
    cnt $\leftarrow$ 0.
    for i $\leftarrow$ 1 to M **do**
        if $\phi_i > \theta$ **then**
            cnt $\leftarrow$ cnt + 1.
    $p_v al \leftarrow \frac{cnt}{M}$
    if $p_v al < \alpha$ **then**
        Aceptar H_0.
    else
        Rechazar H_0.

Como vemos el algoritmo anterior genera una distribución en base a un muestreo con reemplazo de las mejores medias obtenidas en ambos algoritmos, posteriormente se calcula un $p\ val$ y se hace la pregunta si dicho valor es menor que el θ entonces se acepta la hipótesis nula, en caso contrario se rechaza y se hace una comparativa de las medias muestrales de ambos algoritmos para determinar cuál de ellos es mejor. Este experimento se realizó con los datos de la ED y del AG obtenidos con 30 corridas en dimensiones 10 y 30. A continuación se muestran los resultados obtenidos.

Cuadro 3: Resultados de la comparación estadística entre ED y AG usando bootstrap y un α de 0.05.

Dimensión	**Función**	**P value**	**Theta**	**Resultado**
10	Sphere	0	8.59152e-05	Se rechaza H_0 ED mejor que AG
10	Ellipsoid	0	1.95307e-09	Se rechaza H_0 ED mejor que AG
10	Zakharov	0	56.0879	Se rechaza H_0 ED mejor que AG
10	Rosenbrock	0.0022	71.9193	Se rechaza H_0 ED mejor que AG
10	Ackley	0	0.000361627	Se rechaza H_0 ED mejor que AG
10	Griewangk	0	0.0452957	Se rechaza H_0 ED mejor que AG
10	Rastrigin	0	3.95621	Se rechaza H_0 ED mejor que AG
30	Sphere	0	5.23873e-05	Se rechaza H_0 ED mejor que AG
30	Ellipsoid	0	1.95259e-09	Se rechaza H_0 ED mejor que AG
30	Zakharov	0	82.7303	Se rechaza H_0 AG mejor que ED
30	Rosenbrock	0.1314	21.1772	Se acepta H_0
30	Ackley	0	0.000305487	Se rechaza H_0 ED mejor que AG
30	Griewangk	0	0.0421548	Se rechaza H_0 ED mejor que AG
30	Rastrigin	0	68.2738	Se rechaza H_0 AG mejor que ED

En la Tabla anterior podemos ver que para la mayoría de los casos mostrados, con un 95 % de confianza la evolución diferencial superó al algoritmo genético en el caso de la media de los valores de aptitud, y es que si analizamos el *p val* en casi todos los casos vale cero, lo cual indica que en ningún momento la diferencia de las medias de la distribución muestreada superó a la diferencia de las medias de las aptitudes de los conjuntos originales. Para los resultados de las corridas en dimensión 30 fue cuando el algoritmo genético igualó y superó a la evolución diferencial en las funciones de Zakharov, Rosenbrock y Rastrigin.
Por otro lado se puede apreciar que para algunas funciones como Zakharov y Rosenbrock el valor de θ resultó ser bastante algo, lo que indica que uno de los algoritmos generó en promedio mejores resultados que el otro, y por lo que se ha visto en reportes anteriores, la evolución diferencial ha trabajado mejor que el algoritmo genético, al menos con este conjunto de funciones.

Además de realizar este test estadístico también se generó un intervalo de confianza con el mismo conjunto de datos el cual nos indica entre qué valores el algoritmo nos generará soluciones. Dicho intervalo de confianza se generó en base al siguiente algoritmo:

Algorithm 2 Intervalo de confianza.

```
procedure ConfidenceInterval(A, n, α)
    m ← 1000.
    c ← Muestreo con reemplazo de tamaño m de A.
    ĉ ← minSort(c).
    k ← floor(α m).
    z_low ← ĉ_k.
    z_up ← ĉ_{m−k}.
    Return [z_low, z_up].
```

El algoritmo anterior genera de igual manera un conjunto de datos basados en un muestreo con reemplazo y posteriormente los ordena para determinar con un cierto nivel de significancia los valores del intervalo que acotan la generación de buenos valores de aptitud del algoritmo de optimización estudiado.
Este método se usó para generar varios intervalos de confianza con respecto a las mismas funciones a optimizar en dimensiones 10 y 30. A continuación se muestran los datos obtenidos.

Cuadro 4: Intervalos de confianza para el mejor valor de aptitud en ambos algoritmos con una confianza de 95 %.

		ED		**GA**	
Dimensión	**Función**	Z_{low}	Z_{up}	Z_{low}	Z_{up}
10	Sphere	9.80931e-61	4.60006e-59	5.23873e-05	0.000220027
10	Ellipsoid	1.83754e-67	5.04406e-65	1.95259e-09	1.96154e-09
10	Zakharov	4.1307e-14	6.50443e-12	0.00123352	206.516
10	Rosenbrock	5.27211	7.58107	6.338	406.455
10	Ackley	4.44089e-16	3.9968e-15	0.000305487	0.000563772
10	Griewangk	0	0	1.00239e-05	0.116612
10	Rastrigin	0	1.6207e-06	0.000852242	18.8328
30	Sphere	2.22268e-79	5.44584e-78	5.23873e-05	5.23873e-05
30	Ellipsoid	2.91201e-85	9.42009e-84	1.95259e-09	1.95259e-09
30	Zakharov	84.9074	222.101	6.65028e-07	158.237
30	Rosenbrock	25.5095	38.8869	7.13598	308.88
30	Ackley	3.9968e-15	7.54952e-15	0.000305487	0.000305487
30	Griewangk	0	0	7.68511e-06	0.151715
30	Rastrigin	43.6927	83.7642	1.1548e-05	8.0891

Podemos ver en la Tabla anterior que para cada función en dimension 10 y 30 el intervalo de confianza expresa que con un 95 % de certeza se generará el mejor valor de actitud dentro de ese rango. Y lo que se puede notar es que coincide con los resultados anteriores generados por el bootstrap, como ejemplo para la función de Rastrigin en dimensión 30 la prueba de hipótesis indició que el algoritmo genético es mejor que la evolución diferencial, y ahora para los intervalos de confianza se puede ver que el genético generará mejores valores de aptitud en comparación a la evolución diferencial.

3. Comentarios finales

En la presente tarea se usó la implementación del algoritmo CMA-ES en el lenguaje C++ y se utilizó para encontrar los puntos óptimos del conjunto de funciones previamente usado en otras tareas, los resultados obtenidos fueron muy buenos excepto para la última función, en donde no se logró llegar al óptimo global. Esto podría ser por la implementación del mismo algoritmo o que en dado caso no funcione bien para dicha función.
Así mismo se hicieron los análisis estadísticos del algoritmo genético y la evolución diferencial para ver quién generaba los mejores resultados en promedio y con ello se implementó el algoritmo de bootstrap para generar la prueba de hipótesis en donde se mostró que en su mayoría la evolución diferencial ofrece mejores resultados en comparación al algoritmo genético, aunque para la dimensión 30 se pudo ver que la evolución diferencial presenta resultados peores en ciertas funciones que el algoritmo genético, aunque hay que tener en cuenta que ambos algoritmos siempre se corrieron con la misma configuración en sus parámetros, de lo contrario se pudo encontrar un valor de CR para la ED en la que se obtuvieran mejores resultados.