A genetikai algoritmus opcióinak hatásai

Ez a példa a ga genetikai algoritmusfüggvény néhány beállításának hatását mutatja be. A beállításokat az optimalizálás funkció használatával hozhatja létre és módosíthatja.

Probléma beállítása a ga számára

A ga egy függvény minimumát keresi a genetikai algoritmus segítségével. Ebben a példában használja a ga-t a shufcn fitneszfüggvény minimalizálására, amely két változó valós értékű függvénye.

Ábrázolja a shufcn függvényt a = [-2 2;-2 2] tartományban a plotobjective meghívásával:

```
>> plotobjective(@shufcn,[-2 2; -2 2]);
```

A ga megoldó használatához adjon meg legalább két **bemeneti** argumentumot: egy alkalmassági függvényt és a probléma változóinak számát.

A ga által **visszaadott** első két kimeneti argumentum az x, a legjobb talált pont, és az Fval, a függvényérték a legjobb pontban. A harmadik kimeneti argumentum, az exitFlag, jelzi, hogy a ga miért állt le. A ga egy negyedik argumentumot is visszaadhat, a kimenetet, amely a megoldó teljesítményére vonatkozó információkat tartalmaz.

```
>> FitnessFunction = @shufcn;
>> numberof variables = 2;
```

Futtassa a ga megoldót:

```
>> [x,Fval,exitFlag,Output] = ga(FitnessFunction,numberOfVariables);

>> fprintf('A generációk száma: %d\n', Output.generations);
>> fprintf('A függvényértékelések száma: %d\n', Output.funccount);
>> fprintf('A legjobb talált függvényérték: %g\n', Fval);
```

Hogyan működik a genetikai algoritmus

A genetikai algoritmus egy populáción dolgozik a populációra alkalmazott operátorkészlet segítségével. A populáció a tér pontjainak halmaza. A kezdeti populáció alapértelmezés szerint véletlenszerűen jön létre. Az algoritmus a populáció következő generációját számítja ki, felhasználva a jelenlegi generáció egyedeinek alkalmasságát.

Vizualizáció

A megoldó teljesítményének megjelenítéséhez futás közben állítson be egy „PlotFcn” opciót az optimum opciók használatával. Ebben az esetben válasszon két diagramfüggvényt egy cellatömbben. Állítsa be a **gaplotbestf** értéket, amely minden generációnál a populáció legjobb és átlagos értékét ábrázolja.

Állítsa be a **gaplotstopping** kritériumot is, amely a teljes leállási feltétel százalékos arányát ábrázolja.

```
>>opts = optimoptions(@ga,'PlotFcn',{@gaplotbestf,@gaplotstopping});
```

Futtassa a ga megoldót, beleértve az opts argumentumot.

```
>> [x,Fval,exitFlag,Output] =
ga(FitnessFunction,numberOfVariables,[],[],[],[],[],[],[],opts);
```

Adja meg a populáció beállításokat

A beállítások nagy hatással lehetnek a megoldó teljesítményére. Az egyes iterációk sebessége a populáció méretétől függ: a nagyobb populáció lassabb iterációkat eredményez. Ezzel szemben a nagyobb populáció alaposabb feltáráshoz vezet, így jobb megoldást adhat.

A ga létrehoz egy alapértelmezett kezdeti populációt egy egységes véletlenszám-generátor használatával. A ga által használt *alapértelmezett populációméret 50, ha a döntési változók száma kevesebb, mint 5*, egyébként pedig 200. Előfordulhat, hogy az alapértelmezett méret bizonyos problémák esetén nem működik megfelelően; például kisebb problémákra elegendő lehet egy kisebb populáció. Mivel a jelenlegi problémának csak két változója van, adjon meg egy 10-es populációméretet. Állítsa a PopulationSize beállítás értékét 10-re:

```
>> opts.PopulationSize = 10;
```

A kezdeti populáció elemeinek megadása

A kezdeti populáció létrehozásának alapértelmezett módszere egységes véletlenszám-generátort használ. Az egész számra vonatkozó megkötések nélküli problémák esetén a ga létrehoz egy kezdeti sokaságot, amelyben az összes pont a –10 és 10 közötti tartományban van. Például ezzel a paranccsal létrehozhat egy 3-as méretű populációt az alapértelmezett tartományban:

```
>> Population = [-10,-10] + 20*rand(3,2);
```

A kezdeti tartományt az InitialPopulationRange beállítás módosításával állíthatja be. A tartománynak kétsoros mátrixnak kell lennie. Ha a tartománynak csak egy oszlopa van, azaz 2-szer 1, akkor minden változó tartománya az adott tartomány. Például, ha a tartományt [-1; 1], akkor mindkét változó kezdeti tartománya –1 és 1 között van. Ha minden változóhoz más kezdeti tartományt szeretne megadni, a tartományt két sorból és numberOfVariables oszlopból álló mátrixként kell megadnia. Például, ha a tartomány [-1 0; 1 2], akkor az első változó –1 és 1, a második változó pedig 0 és 2 közötti (minden oszlop egy változónak felel meg).

Módosítsa az InitialPopulationRange opció értékét:

```
>> opts.InitialPopulationRange = [-1 0; 1 2];
```

Futtassa a ga megoldót:

```
>> [x,Fval,exitFlag,Output] = ga(FitnessFunction,numberOfVariables,[],[],[], ...
    [],[],[],[],opts);

>> fprintf('A generációk száma: %d\n', Output.generations);
>> fprintf('A függvényértékelések száma: %d\n', Output.funccount);
>> fprintf('A legjobb talált függvényérték: %g\n', Fval);
```

Futtassa a ga megoldót:

```
>> [x,Fval,exitFlag,Output] = ga(FitnessFunction,numberOfVariables,[],[],[], ...
    [],[],[],[],opts);

>> fprintf('A generációk száma: %d\n', Output.generations);
>> fprintf('A függvényértékelések száma: %d\n', Output.funccount);
>> fprintf('A legjobb talált függvényérték: %g\n', Fval);
```

Eredmények reprodukálása

Alapértelmezés szerint a ga egy véletlenszerű kezdeti populációval kezdődik, amelyet MATLAB® véletlenszám-generátorokkal hoztak létre. A megoldó a következő generációt ga operátorokkal állítja elő, amelyek szintén ugyanazokat a véletlenszám-generátorokat használják. Valahányszor véletlen számot generálnak, a véletlenszám-generátor állapota megváltozik. Tehát még ha nem is változtat semmilyen beállításon, a megoldó újbóli futtatásakor eltérő eredményeket kaphat.

Futtassa kétszer a megoldót a jelenség megjelenítéséhez.

Futtassa a ga megoldót:

```
>> [x,Fval,exitFlag,Output] = ga(FitnessFunction,numberOfVariables);
fprintf('A legjobb talált függvényérték: %g\n', Fval);
```

Futtassa újra a ga-t:

```
>> [x,Fval,exitFlag,Output] = ga(FitnessFunction,numberOfVariables);
fprintf('A legjobb talált függvényérték: %g\n', Fval);
```

A ga különböző eredményeket ad a két futásban, mert a véletlenszám-generátor állapota egyik futásról a másikra változik.

Ha reprodukálni szeretné az eredményeket a ga futtatása előtt, mentheti a véletlenszám-folyam állapotát.

```
>> thestate = rng;

>> [x,Fval,exitFlag,Output] = ga(FitnessFunction,numberOfVariables);
fprintf('A legjobb talált függvényérték: %g\n', Fval);
```

Állítsa vissza az adatfolyamot, és futtassa újra a ga-t. Az eredmények megegyeznek az előző futással.

```
>> rng(thestate);
>> [x,Fval,exitFlag,Output] = ga(FitnessFunction,numberOfVariables);
fprintf('A legjobb talált függvényérték: %g\n', Fval);
```

Módosítsa a leállási feltételeket

A ga négy különböző kritériumot használ annak meghatározására, hogy mikor kell leállítani a megoldót. A ga leáll, amikor eléri a generációk maximális számát; alapértelmezés szerint ez a szám 100-szorosa a változók számának. A ga azt is érzékeli, hogy a legjobb fitness érték nem változik-e egy ideig másodpercben (leállási időkorlát), vagy néhány generáción keresztül (maximális leállási generáció). Egy másik kritérium a maximális időkorlát másodpercben. Módosítsa a leállítási feltételeket, a generációk maximális számát 300-ra, a maximális leállási generációkat pedig 100-ra növelje.

```
>> opts = optimoptions(opts,'MaxGenerations',300,'MaxStallGenerations',
100);
```

Futtassa újra a ga megoldót:

```
>> [x,Fval,exitFlag,Output] = ga(FitnessFunction,numberOfVariables,[],[],[], ...
    [],[],[],[],opts);

>> fprintf('A generációk száma: %d\n', Output.generations);
>> fprintf('A függvényértékelések száma: %d\n', Output.funccount);
>> fprintf('A legjobb talált függvényérték: %g\n', Fval);
```

A ga operátorok megadása

A ga a sokaság véletlenszerű pontkészletével kezdődik, és operátorokat használ a sokaság következő generációjának létrehozásához. A különböző operátorok a skálázás, a kiválasztás, a keresztezés és a mutáció. Az eszköztár számos funkciót kínál, amelyeket minden egyes operátorhoz megadhatunk.

Adja meg a FitnessScalingFcn-hez a fitscalingprop-ot, a SelectionFcn-hez pedig a selectiontornament-et:

```
>> opts = optimoptions(@ga,'SelectionFcn',@selectiontournament,
                    'FitnessScalingFcn',@fitscalingprop);
```

Futtassa újra ga-t:

```
>> [x,Fval,exitFlag,Output] = ga(FitnessFunction,numberOfVariables,[],[],[],
    [],[],[],[],opts);

>> fprintf('A generációk száma: %d\n', Output.generations);
>> fprintf('A függvényértékelések száma: %d\n', Output.funccount);
>> fprintf('A legjobb talált függvényérték: %g\n', Fval);
```

A legjobb függvényérték javulhat vagy romolhat a megadott operátorok alapján. ***A különböző operátorokkal végzett kísérletezés gyakran a legjobb módja annak, hogy meghatározza, melyik operátorkészlet felel meg a legjobban a problémának!***