Projekt zaliczeniowy

Projekt zaliczeniowy

Projekt polega na prostym opracowaniu statystycznym wyników porównania działania wybranych algorytmów minimalizacji stochastycznej. Do porównania należy wybrać **dwa** spośród zaprezentowanych poniżej.

Poszukiwanie przypadkowe (*Pure Random Search, PRS*)

Losujemy po kolei zadaną z góry liczbę punktów z rozkładem jednostajnym w zadanej dziedzinie. Jeżeli dziedzina jest kostką wielowymiarową (tu tak ma być), to można (i trzeba) losować kolejno współrzędne poszczególnych punktów według odpowiedniego jednowymiarowego rozkładu jednostajnego, np. jeśli dziedzina poszukiwania jest kostką trójwymiarową

$$[0, 1] \times [-2, 2] \times [100, 1000],$$

to pierwszą współrzędną każdego punktu losujemy z rozkładu $\mathcal{U}(0, 1)$, drugą z rozkładu $\mathcal{U}(-2, 2)$, a trzecią z rozkładu $\mathcal{U}(100, 1000)$. Każdy wylosowany punkt porównujemy z aktualnie zapamiętanym minimum i jeśli wartość minimalizowanej funkcji w tym punkcie jest mniejsza, to ten punkt zapamiętujemy jako aktualny punkt minimalny. Wartość funkcji w ostatnim zapamiętanym punkcie stanowi wynik algorytmu.

Metoda wielokrotnego startu (*multi-start, MS*)

Losujemy (podobnie jak poprzednio) zadaną liczbę punktów z rozkładem jednostajnym w dziedzinie przeszukiwania, a następnie z każdego z wylosowanych punktów startujemy numeryczną metodę optymalizacji lokalnej: w tym wypadku należy użyć metody **L-BFGS-B** dostępnej w standardowej fukcji `optim()`. Wynikiem algorytmu jest wartość optymalizowanej funkcji w tym z punktów zwróconych przez uruchomienia metody lokalnej, w którym ta wartość jest najmniejsza.

Algorytm genetyczny (*GA*)

Implementacje można znaleźć np. w bibliotekach `GA` i `ecr` dostępnych w CRAN. Wynikiem jest najmniejsza znaleziona wartość funkcji.

Uwagi

- Do porównania należy wybrać **dwie** z funkcji dostępnych w pakiecie `smoof`, które są skalarne (*single-objective*) i wielomodalne (*multimodal*), i mają wersje dla różnej liczby wymiarów (akceptują parametr `dimensions`). Własności te mają np. funkcje Ackley'a, Rastrigina, Schwefela, Rosenbrocka, Michalewicza, *Eggholder*, *Alpine01*, *Alpine02*, ale **nie** funkcja kwadratowa (*Sphere*).
- Dziedziny poszukiwań są przypisane poszczególnym funkcjom z pakietu `smoof` (atrybut `par.set`) i należy je respektować (w szczególności w wywołaniu funkcji `optim()`).
- Porównanie należy wykonać dla każdej funkcji osobno i osobno dla liczby wymiarów 2, 10 i 20 (co daje łącznie 6 porównań).

- Porównać należy **średni** wynik algorytmu, czyli średnią *znalezionych* minimów. Średnią liczymy z nie mniej niż 50 uruchomień każdego algorytmu osobno dla każdej funkcji i każdej liczby wymiarów (co daje łącznie 600 uruchomień). W tego rodzaju powtarzanych obliczeniach użyteczna jest funkcja `replicate()`.
- Należy zadbać o wyrównany budżet obliczeniowy porównywanych algorytmów - budżet określamy jako liczbę wywołań minimalizowanej funkcji. Dla algorytmów genetycznych ten budżet należy określić wprost jako warunek zatrzymania. Konkretnie:
    - w porównaniu PRS-GA należy budżet określić na 1000 wywołań;
    - jeśli jedną z porównywanych metod jest MS, to należy tę metodę uruchomić jako pierwszą dla 100 punktów startowych, a następnie **średnią** liczbę wywołań z uruchomień MS określić jako budżet porównywanej metody (GA lub PRS).
- W ramach porównania należy wykonać obrazki, tzn. histogramy i wykresy pudełkowe (lub skrzypcowe), i dokonać analizy ich zawartości pod kątem położenia i rozproszenia rozkładu wyników.
- Należy też dokonać analizy istotności statystycznej różnicy między średnim wynikiem obu algorytmów (dla każdej funkcji i liczby wymiarów osobno). Analizę należy oprzeć na odpowiednich przedziałach ufności (95-procentowych) i testach hipotez zerowych. Ze względu na liczbę powtórzeń zakładamy, że rozkład średniej może być sensownie przybliżony przez rozkład normalny.
- Projekt może być realizowany w zespołach dwuosobowych.
- Projekt ma być wykonany w środowisku `R`. Raport wyników może mieć formę dokumentu R Markdown.