Tarea 2 Búsqueda aleatoria

Asignatura: Optimización
Profesor: D. Sc. Gerardo García Gil
Alumno: 2023-A Miguel Angel Luis Espinoza 20110393
Ingeniería en Desarrollo de Software
Centro de Enseñanza Técnica Industrial (CETI)

Búsqueda Aleatoria

Presentación

Realizar un programa con el algoritmo de búsqueda aleatoria para resolver el siguiente problema de maximización:

3*(1-x) .^ 2. *exp(-(x.^2)-(y+1).^2)-10*(x/5-x.^3-y.^5).*exp(-x.^2-y.^2)-3*exp(-(x+1).^2-y.^2)

Sujeto a: -3 <= x1 <= 3 y -3 <= x2 <= 3

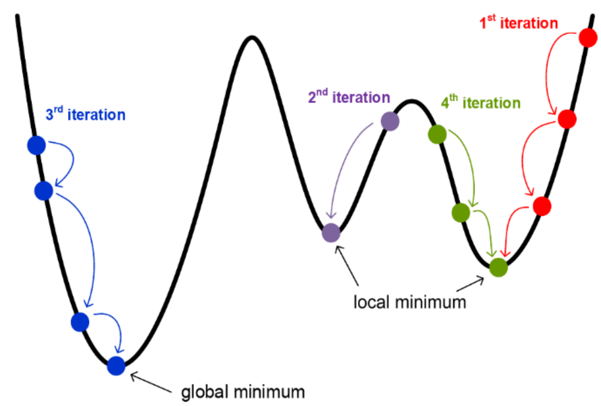
Y su representación gráfica utilizando la plataforma de MATLAB.

Introducción

La búsqueda aleatoria (RS) es una familia de métodos de optimización numérica que no requieren que se optimice el gradiente del problema y, por lo tanto, RS puede usarse en funciones que no son continuas o diferenciables. Dichos métodos de optimización también se conocen como métodos de búsqueda directa, sin derivados o de caja negra.

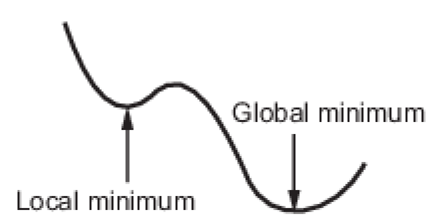
Anderson en 1953 revisó el progreso de los métodos para encontrar el máximo o el mínimo de problemas usando una serie de conjeturas distribuidas con cierto orden o patrón en el espacio de búsqueda de parámetros, por ejemplo, un diseño confundido con espaciamientos/pasos distribuidos exponencialmente. Esta búsqueda continúa secuencialmente en cada parámetro y refina iterativamente las mejores conjeturas de la última secuencia.

El nombre de "búsqueda aleatoria" se atribuye a Rastrigin, quien hizo una presentación temprana de RS junto con un análisis matemático básico. RS funciona moviéndose iterativamente a mejores posiciones en el espacio de búsqueda, que se muestrean desde una hiperesfera que rodea la posición actual.



En general, los solvers devuelven un mínimo local (u óptimo). El resultado puede ser un mínimo global (u óptimo), pero este resultado no está garantizado.

- Un mínimo local de una función es un punto en el que el valor de la función es menor que en puntos cercanos, pero posiblemente mayor que en un punto alejado.
- Un mínimo global es un punto en el que el valor de la función es menor que en otros puntos factibles.



Algoritmo

Sea f : ℝ n → ℝ la función de coste o de aptitud que debe minimizarse. Sea x ∈ ℝ n designe una posición o solución candidata en el espacio de búsqueda. El algoritmo RS básico se puede describir como:

Inicialice x con una posición aleatoria en el espacio de búsqueda.

Hasta que se cumpla un criterio de terminación (p. ej., número de iteraciones realizadas o aptitud adecuada alcanzada), repita lo siguiente:

Muestree una nueva posición y de la hiperesfera de un radio dado que rodea la posición actual x (consulte, por ejemplo, la técnica de Marsaglia para muestrear una hiperesfera).

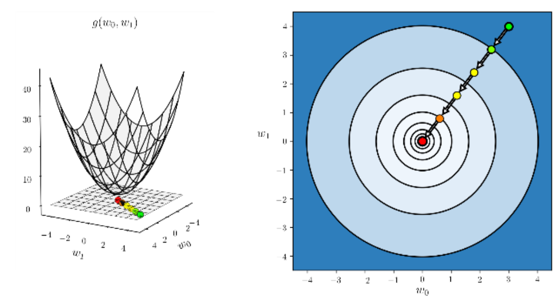
Si f (y) < f (x) , muévase a la nueva posición configurando x = y

Algorithm 1.2 Random Search Method

1. $k \leftarrow 0$
2. $x_1^k \leftarrow$ Random $[-3,3]$, $x_2^k \leftarrow$ Random $[-3,3]$
3. **while** (k< *Niter*) {
4. $\Delta x_1 = N(0,1), \Delta x_2 = N(0,1)$
5. $x_1^{k+1} = x_1^k + \Delta x_1, x_2^{k+1} = x_2^k + \Delta x_2$
6. $\mathbf{x}^k = \left(x_1^k, x_2^k\right), \mathbf{x}^{k+1} = \left(x_1^{k+1}, x_2^{k+1}\right)$
7. If $\left(\mathbf{x}^{k+1} \notin \mathbf{X}\right)\left\{f\left(\mathbf{x}^{k+1}\right) = -\infty\right\}$
8. If $\left(f\left(\mathbf{x}^{k+1}\right) < f\left(\mathbf{x}^k\right)\right)\left\{\mathbf{x}^{k+1} = \mathbf{x}^k\right\}$
9. $k \leftarrow k+1$}

Con la búsqueda aleatoria hacemos lo más perezoso posible que uno pudiera pensar en hacer para encontrar una dirección de descenso: muestreamos un número determinado de direcciones aleatorias derivadas de wk − 1, evalúe cada punto de actualización candidato y elija el que nos dé la evaluación más pequeña (siempre que esté más abajo en la función que nuestro punto actual). En otras palabras, buscamos localmente alrededor del punto actual en un número fijo de direcciones aleatorias en busca de un punto que tenga una evaluación más baja y, si encontramos uno, nos movemos hacia él.

1. entrada: punto inicial w0, número máximo de pasos k, número de muestras aleatorias por paso PAG, un paso α o regla
2. de longitud de paso decreciente: for k = 1... k
3. calcular PAG unidades de longitud direcciones aleatorias {dpag}PAGp = 1 (p. ej., muestreando y normalizando un norte Gaussiana dimensional)
4. encontrar s =argmínpag = 1... paggramo(wk − 1+ adpag)
5. conjunto dk=ds
6. formar un nuevo punto wk=wk − 1+ adk
7. if gramo(wk) <gramo(wk − 1)
8. wk − 1⟵wk
9. salida: historial de pesos {wk}kk = 0 y evaluaciones de funciones correspondientes {gramo (wk)} kk = 0



Desarrollo

Con este algoritmo se desarrollará un programa con el IDE de MATLAB, donde la función será: 10-(exp(-1*(x^2+3*y^2))). Se grafica en 3D.

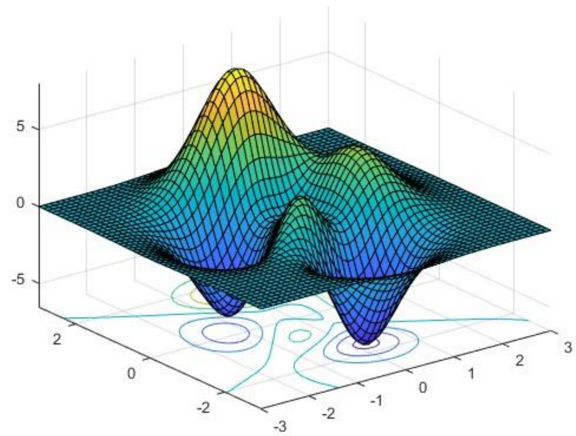
Código

```
%Random Search Algorithm
%Miguel Luis
clear all; clc;
%Definition of objective function
funstr = '3*(1-x).^2.*exp(-(x.^2)-(y+1).^2)-10*(x/5-x.^3-y.^5).*exp(-x.^2-y.^2)-3*exp(-(x+1).^2-y.^2)'; %Los puntos
f = vectorize(inline(funstr));
range=[-3 3 -3 3]; %El primer rango es para x: -3 a 3. Lo segundo para y: -3 a 3, search space
```

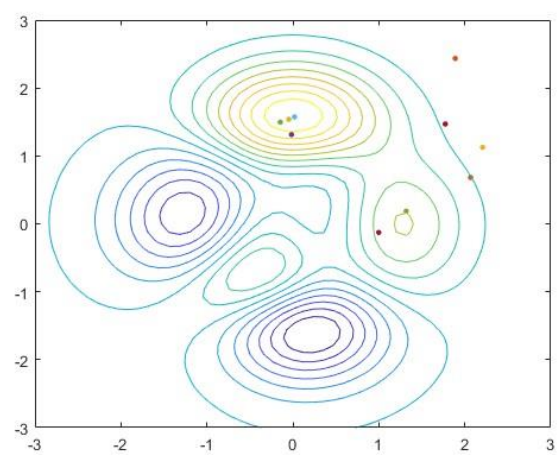
```
%Draw the objective function
Ndiv=50;
dx=(range(2)-range(1))/Ndiv;
dy=(range(4)-range(3))/Ndiv;
[x,y] = meshgrid(range(1):dx:range(2), range(3):dy:range(4)); %Meshgrid, líneas dentro de la figura
z = f(x,y);
figure(1);
surfc(x,y,z);
NITER = 300; k=0;
%Initialization of the candidate solution
xrange = range(2) - range(1);
yrange = range(4) - range(3);
xn = rand * xrange + range(1);
yn = rand * yrange + range(3);
%x^k+1 = x^k+triangulo x
%Starting point of the optimization process
while (k<NITER)
   %It is tested if the solution falls inside the search space
   if((xn>=range(1)) && (xn<=range(2)) && (yn>=range(3)) && (yn<=range(4))) % If yes, it is evaluated
      zn1=f(xn,yn);
   else
   % if not, it is assigned a low quality
      zn1= -100;
   end
   %The produced soluction is draw
   figure(2);
   contour(x,y,z,15); hold on; %Imagen plasmada en el suelo y hold on deja la imagen anterior
   plot(xn, yn, '.', 'markersize',10, 'markerfacecolor', 'g');
   drawnow;hold on;
   %A new solution is produced
   xnc = xn + randn*1;
   ync = yn + randn*1;
   if((xnc >= range(1)) && (xnc<=range(2)) && (ync>=range(3)) && (ync<=range(4)))
      %if yes, it is evaluated
      zn2=f(xnc,ync);
   else
      %If not, it is assigned a low quality
      zn2 = -1000;
   end
   % It is analyzed if the new solution is accepted
   if (zn2 > zn1)
      xn = xnc;
      yn = ync;
   end
   k = k+1;
end
```

Resultados

Figura a analizar



Representación gráfica del punto máximo



Conclusión

Con la elaboración de esta práctica se familiarizo con la lógica del algoritmo de búsqueda aleatoria y con el sistema de cómputo de MATLAB; Se realizaron gráficas tanto en 3D y 2D y se usaron nuevos comandos como vectorize, contour y rand.

Referencias

1. Ã“ ptimos locales frente a globales - MATLAB & Simulink - MathWorks América Latina. (s. f.). https://la.mathworks.com/help/optim/ug/local-vs-global-optima.html
2. Wikipedia contributors. (2022, 5 octubre). Random search. Wikipedia. https://en.wikipedia.org/wiki/Random_search
3. Random search. (s. f.). https://kenndanielso.github.io/mlrefined/blog_posts/5_Zero_order_methods/5_4_Random_search.html