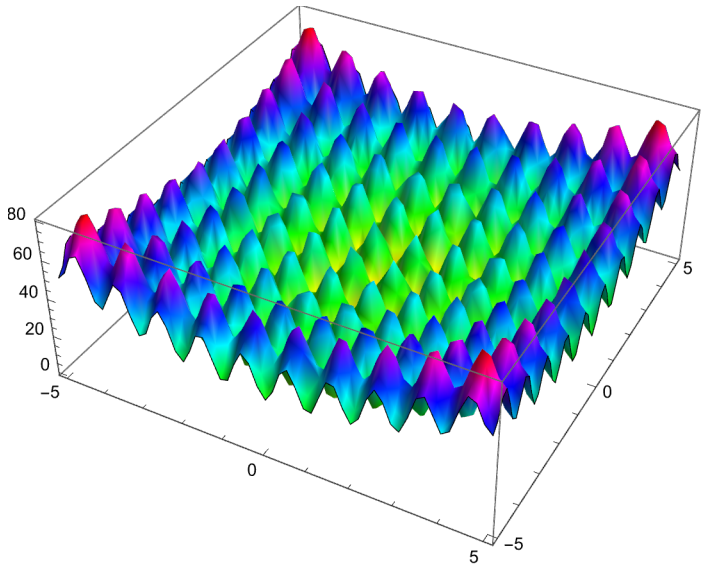
蜻蜓算法

In[ ]:=
```
fitness[xn_] := 20 + Sum[x^2 - 10 Cos[2 π x], {x, xn}]
                      求和          余弦

Plot3D[fitness[{x, y}], {x, -5, 5}, {y, -5, 5},
绘制三维图形
 ColorFunction → Hue, Mesh → None, RotationAction → "Clip"]
 颜色函数          色相 网格   无    旋转操作          剪切
```

Out[ ]=



In[ ]:=
```
fitness[{0, 0}]
```

Out[ ]=
```
0
```

DA 实现

In[ ]:=
```
levy[d_] := Block[{β = 3 / 2, σ},
            块
  σ = (Gamma[1 + β] Sin[π β/2] / (Gamma[(1+β)/2] β 2^((β-1)/2)))^(1/β);

  0.01 RandomVariate[NormalDistribution[], d] σ / Abs[RandomVariate[NormalDistribution[], d]]^(1/β)]
```

In[ ]:=
```
levy[2]
```

Out[ ]=
```
{-0.00374963, 0.00218879}
```

In[ ]:=
```
0.01 RandomVariate[LevyDistribution[-1, 1.5], {1, 2}]
     伪随机变数        利维分布
```

Out[ ]=
```
{{1.98068, 0.304878}}
```

(调试) In[ ]:=
```
ub = 5;
lb = -5;
dimension = 2;
num = 50;

x = RandomReal[{ub, lb}, {num, dimension}];
    伪随机实数
deltaX = RandomReal[{ub, lb}, {num, dimension}];
         伪随机实数
xFood = RandomReal[{ub, lb}, dimension];
        伪随机实数
fitFood = ∞;
xEnemy = RandomReal[{ub, lb}, dimension];
         伪随机实数
fitEnemy = -∞;

iterNum = 100;
fitCurve = Table[0, iterNum];
           表格
(*迭代*)
Monitor[Do[
监控    Do循环
  r = (ub - lb)/4 + (ub - lb) 2 iter/iterNum;
  w = 0.9 - iter (0.9 - 0.4)/iterNum;
  e = 0.1 - iter (0.1 × 2)/iterNum;
  If[e < 0, e = 0];
  如果
  s = 2 RandomReal[] e;
      伪随机实数
  a = 2 RandomReal[] e;
      伪随机实数
  c = 2 RandomReal[] e;
      伪随机实数
  f = 2 RandomReal[];
      伪随机实数

  (*计算目标函数*)
  Do[fit = fitness[x[[i]]];
  Do循环
    If[fit < fitFood, fitFood = fit; xFood = x[[i]]];
    如果
    If[fit > fitEnemy, fitEnemy = fit; xEnemy = x[[i]], {i, num}];
    如果
```

```
fitCurve[[iter]] = fitFood;

(*遍历所有个体*)
Do[
 Do循环
  neighboursNo = 0;
  neighboursX = {};
  neighboursDeltaX = {};
  (*寻找邻居*)
  Do[
   Do循环
    If[EuclideanDistance[x[[u]], x[[v]]] ≤ r, neighboursNo++;
     … 欧几里得距离
      AppendTo[neighboursX, x[[v]]];
       附加
      AppendTo[neighboursDeltaX, deltaX[[v]]];
       附加
    ], {v, num}];

    If[neighboursNo > 0,
     如果
      Si = -Sum[x[[u]] - j, {j, neighboursX}];
            求和
      Ai = Mean[neighboursDeltaX];
           平均值
      Ci = Mean[neighboursX] - x[[u]];
           平均值
      Fi = xFood - x[[u]];
      Ei = xEnemy + x[[u]];

      deltaX[[u]] = (s Si + a Ai + c Ci + f Fi + e Ei) + w deltaX[[u]];
      x[[u]] = x[[u]] + deltaX[[u]],

      x[[u]] = x[[u]] (1 + levy[dimension])];

     x, {u, num}],

   {iter, iterNum}], {iter, fitFood}]
(*输出最优结果*)
{fitFood, xFood}
```

(调试) Out[ ]=

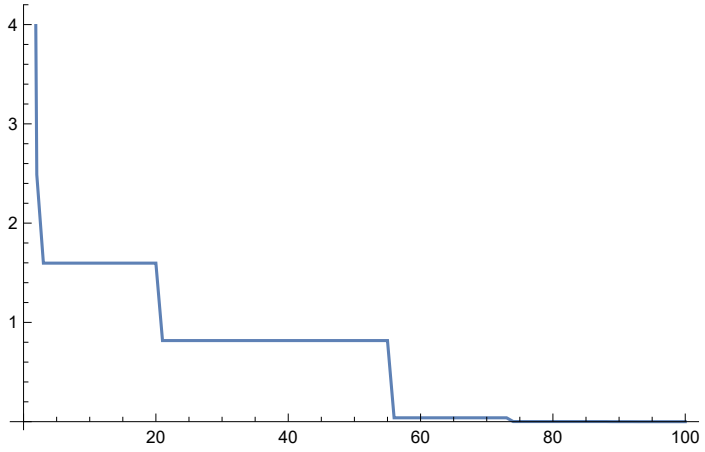
$$\left\{1.63645 \times 10^{-9}, \left\{-7.29073 \times 10^{-7}, 2.77795 \times 10^{-6}\right\}\right\}$$

(调试) In[ ]:=

```
ListLinePlot[fitCurve]
```

绘制点集的线条

(调试) Out[ ]=



DSolve

求解微分方程