南京智能优化与调度学习班

差分进化算法（Differential Evolution）

龚文引 (教授、博士生导师)

中国地质大学（武汉）计算机学院

August 30, 2019

1. 大纲

- 算法简介
- 算法基本流程
- 简单示例
- 代表性改进方法
- 小结

2. 算法简介

2. 算法简介

差分进化算法（差分进化算法）

- 进化算法是一种自适应, 并行的全局优化算法;
- 差分进化算法是一种进化算法;
- 进化算法还包括遗传算法, 进化策略, 进化规划和遗传编程等;
- 差分进化算法与其他进化算法的最大区别在于差分变异算子的应用.

R. Storn, K. Price, “Differential evolution – A simple and efficient heuristic for global optimization over continuous spaces," *J. of Global Optim.* 1997, 11 (4): 341 - 359.

差分进化算法

- Storn & Price 于 1995 年首次提出;
- 采用实数编码, 主要用于求解实数优化问题;
- 所求解优化问题可描述为:
  对于目标函数 $f: \mathbb{X} \subseteq \mathbb{R}^n \rightarrow \mathbb{R}$, 求最小解 $\mathbf{x}^*$

$$\mathbf{x}^* \in \mathbb{X} : f(\mathbf{x}^*) \leq f(\mathbf{x}), \forall \mathbf{x} \in \mathbb{X}$$

2. 算法简介

差分进化算法

- DE grew out of Ken Price's attempts to solve the Chebychev Polynomial fitting Problem that had been posed to him by Rainer Storn;
- A breakthrough happened, when Ken came up with the idea of using vector differences for perturbing the vector population;
- Since this seminal idea a lively discussion between Ken and Rainer and endless ruminations and computer simulations on both parts yielded many substantial improvements which make DE the versatile and robust tool it is today.
- It is the strong wish of Ken and Rainer that DE will be developed further by scientists around the world and that DE may improve to help more users in their daily work. This wish is the reason why DE has not been patented in any way.

2. 算法简介

算法优点

- 结构简单，其核心代码只需约 30 行 C 语言代码;
- 容易使用，算法参数少，只有 3 个（群体大小 NP，杂交概率 Cr，缩放因子 F）；
- 收敛速度快;
- 鲁棒性好.

3. 算法基本流程

3. 算法基本流程

符号说明

- 设要求解一个 n 维实数优化问题;
- 设置群体大小为 NP ($NP \geq 4$);
- 个体编码为:

$$\mathbf{x}_i = \{x_{i,1}, x_{i,2}, \cdots, x_{i,n}\}$$

其中: $i = 1, 2, \cdots, NP$; $x_{i,j}$ 为一实数, 且 $L_j \leq x_{i,j} \leq U_j$, $j = 1, 2, \cdots, n$.

3. 算法基本流程

算术杂交 (Arithmetic Crossover)

$$\begin{aligned}\mathbf{y} &= F \times \mathbf{x}_1 + (1 - F) \times \mathbf{x}_2 \\ &= \mathbf{x}_2 + F \times (\mathbf{x}_1 - \mathbf{x}_2)\end{aligned}$$

where $F \in [0, 1]$ is a real-valued parameter.

3. 算法基本流程

算术杂交 (Arithmetic Crossover)

$$\begin{aligned}\mathbf{y} &= F \times \mathbf{x}_1 + (1-F) \times \mathbf{x}_2 \\ &= \mathbf{x}_2 + F \times (\mathbf{x}_1 - \mathbf{x}_2)\end{aligned}$$

where $F \in [0, 1]$ is a real-valued parameter.

均匀杂交 (Uniform Crossover)

$$y_i = \begin{cases} x_{1,i}, & \text{if } \mathrm{rndreal}(0,1) < Cr \\ x_{2,i}, & \text{otherwise} \end{cases}$$

where $Cr \in [0, 1]$ is a crossover rate.

3. 算法基本流程

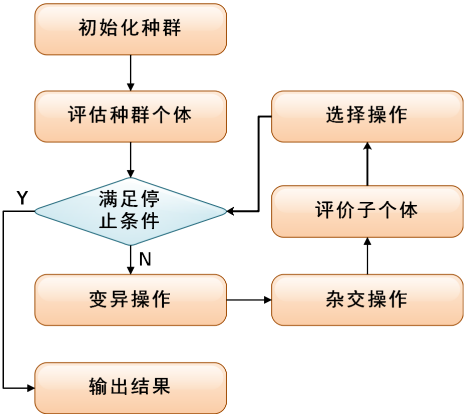
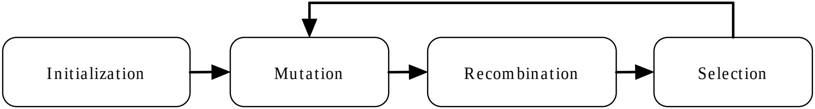


Figure: 差分进化算法基本流程图

3. 算法基本流程



Algorithm 1: 经典差分演化算法流程

产生初始群体

计算群体中个体适应值

while *终止条件未达到* **do**

  for $i = 1$ *to* NP **do** /* 产生子个体 */

    选择三个随机父个体$r_1 \neq r_2 \neq r_3 \neq i$

    利用DE变异和杂交算子产生实验向量$\mathbf{u}_i$

    计算$\mathbf{u}_i$适应值

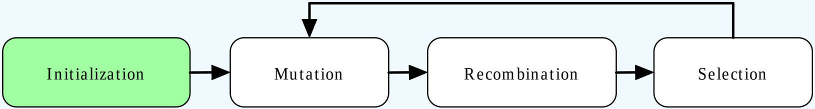
  for $i = 1$ *to* NP **do** /* 选择算子 */

    if $\mathbf{u}_i$ *优于*$\mathbf{x}_i$ **then**

      $\mathbf{x}_i = \mathbf{u}_i$

3. 算法基本流程

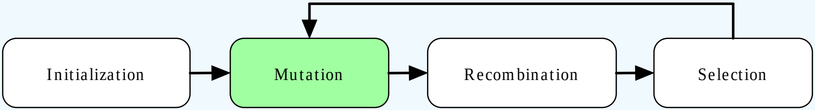
群体初始化



群体中每个个体每一维 $x_{i,j}$ 在其自变量范围 $[L_j, U_j]$ 内采用均匀随机初始化:

$$x_{i,j} = \text{rndreal}(L_j, U_j)$$

差分变异算子



差分进化算法的核心算子是差分变异(Differential mutation)算子，其中，经典算子“DE/rand/1”为:

$$\mathbf{v}_i = \mathbf{x}_{r_1} + F \cdot \left(\mathbf{x}_{r_2} - \mathbf{x}_{r_3}\right)$$

- F为缩放因子 (scaling factor), $F \in (0, 1+)$;
- $\mathbf{v}_i$为变异向量 (mutant vector);
- $\mathbf{x}_{r_1}$为基向量 (base vector);
- $\mathbf{x}_{r_2} - \mathbf{x}_{r_3}$为差分向量 (differential vector);
- $r_1, r_2, r_3 \in \{1, NP\}$, 且 $r_1 \neq r_2 \neq r_3 \neq i$.

3. 算法基本流程

变异算子

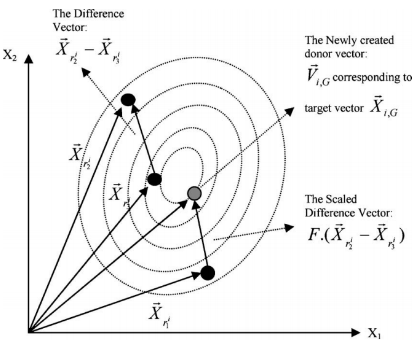
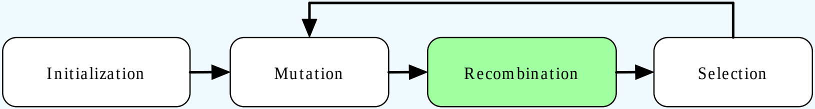


Figure: DE 变异算子示意图 (Das & Suganthan, TEVC, 2011.)

3. 算法基本流程

杂交算子



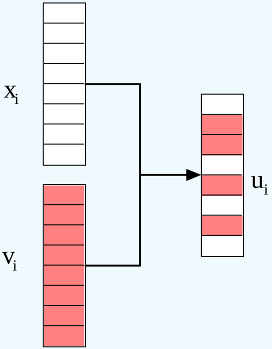
差分进化算法采用离散重组 (discrete recombination), 常用的二项式杂交算子为:

$$u_{i,j} = \begin{cases} v_{i,j}, & \text{if } \mathrm{rndreal}(0,1) < Cr \mid\mid j == j_{\mathrm{rand}} \\ x_{i,j}, & \text{otherwise} \end{cases}$$

- $j_{\mathrm{rand}} = \mathrm{rndint}(0, n-1)$;
- $Cr \in [0,1]$ 为杂交概率;
- $\mathbf{x}_i$ 为目标向量 (target vector);
- $\mathbf{u}_i$ 为实验向量 (trial vector).

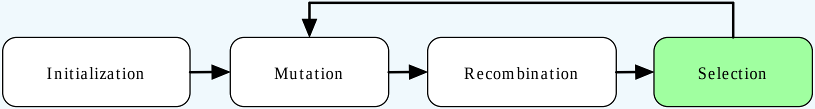
3. 算法基本流程

杂交算子



3. 算法基本流程

选择算子

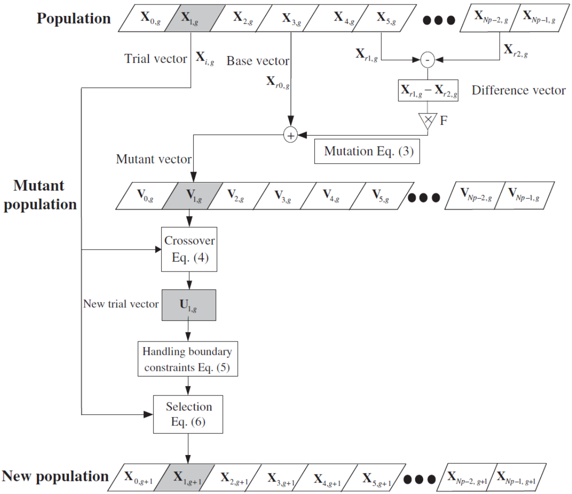


差分进化算法采用一对一锦标赛选择 (one-to-one tournament selection) 算子, 即目标向量 $\mathbf{x}_i$ 与实验向量 $\mathbf{u}_i$ 相比较, 较好个体保存到一下代:

$$\mathbf{x}_i' = \begin{cases} \mathbf{u}_i, & \text{if } f(\mathbf{u}_i) \leq f(\mathbf{x}_i) \\ \mathbf{x}_i, & \text{otherwise} \end{cases}$$

变异, 重组 (杂交) , 选择算子重复执行, 直到终止条件达到.

3. 算法基本流程


Population
$X_{0,g}$ $X_{1,g}$ $X_{2,g}$ $X_{3,g}$ $X_{4,g}$ $X_{5,g}$ ●●● $X_{Np-2,g}$ $X_{Np-1,g}$
Trial vector $X_{i,g}$
Base vector $X_{r0,g}$
$X_{r1,g}$
$X_{r2,g}$
-
$X_{r1,g} - X_{r2,g}$
Difference vector
F
+
Mutation Eq. (3)
Mutant vector
Mutant population
$V_{0,g}$ $V_{1,g}$ $V_{2,g}$ $V_{3,g}$ $V_{4,g}$ $V_{5,g}$ ●●● $V_{Np-2,g}$ $V_{Np-1,g}$
Crossover Eq. (4)
New trial vector
$U_{1,g}$
Handling boundary constraints Eq. (5)
Selection Eq. (6)
New population
$X_{0,g+1}$ $X_{1,g+1}$ $X_{2,g+1}$ $X_{3,g+1}$ $X_{4,g+1}$ $X_{5,g+1}$ ●●● $X_{Np-2,g+1}$ $X_{Np-1,g+1}$

DE 算子的两大特点

1. 自适应性 (Self-adaptation)：搜索步长（step size）和搜索方向（orientation）随着进化会随目标函数自适应改变
2. 旋转不变性 (Rotationally invariant)：当$Cr = 1$时，DE 的搜索不随坐标系统旋转而改变。因此Cr取值接近1时，适合于求解变量耦合（变量相互依赖）问题

4. 简单示例

- 算法简介
- 算法基本流程
- 简单示例
- 代表性改进方法
- 小结

4. 简单示例

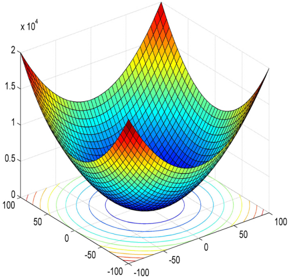
示例: 求 Sphere 函数最小值

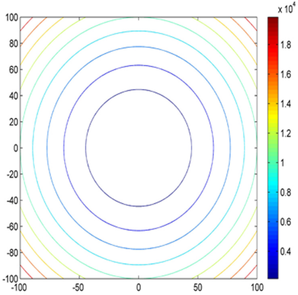
- Sphere 函数

$$f(x_1, x_2) = x_1^2 + x_2^2$$

- 求解 $\mathbf{x}^* \in [-100, 100]$ 使得 $f(\mathbf{x}^*) \leq f(\mathbf{x}), \forall \mathbf{x} \in [-100, 100]$.
- $f(\mathbf{x}^*) = 0$, $\mathbf{x}^* = (0, 0)$.

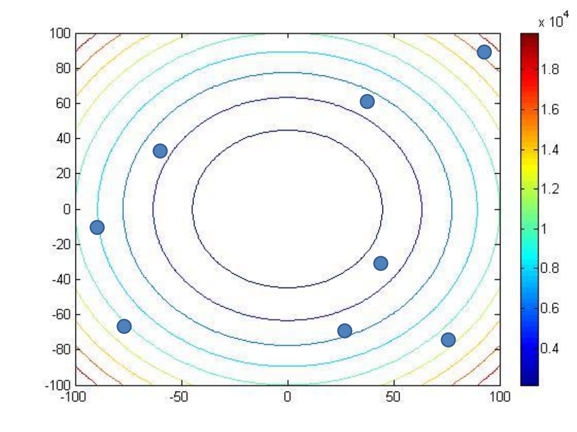
4. 简单示例


x 10^4
2
1.5
1
0.5
0
100
50
0
-50
-100
-100
-50
0
50
100


100
80
60
40
20
0
-20
-40
-60
-80
-100
-100
-50
0
50
100
x 10^4
1.8
1.6
1.4
1.2
1
0.8
0.6
0.4

4. 简单示例

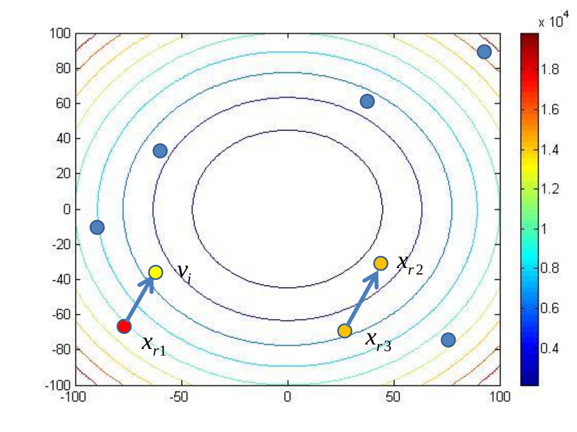
示例: 初始化



x 10^4
1.8
1.6
1.4
1.2
1
0.8
0.6
0.4
100
80
60
40
20
0
-20
-40
-60
-80
-100
-100
-50
0
50
100

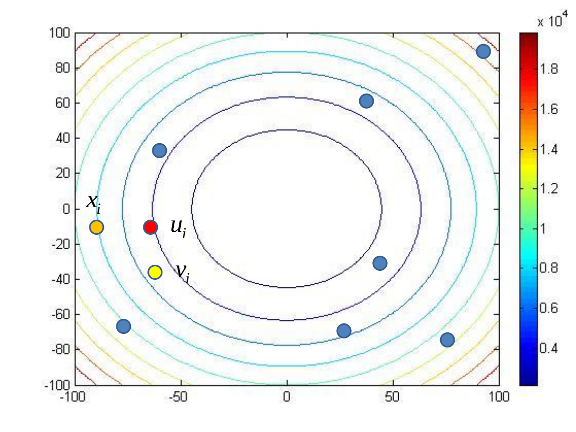
4. 简单示例

示例: 变异算子



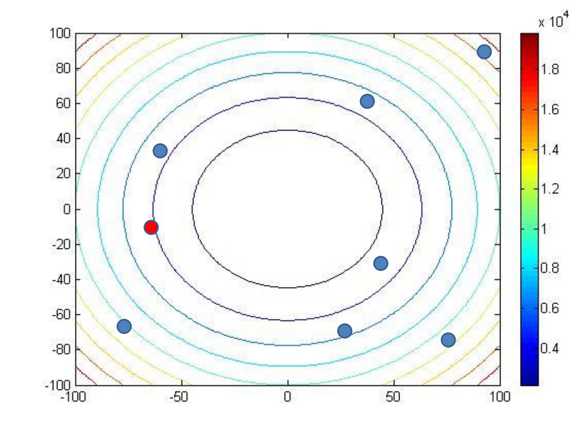
4. 简单示例

示例: 重组 (杂交) 算子



4. 简单示例

示例: 选择算子



5. 代表性改进方法

5. 代表性改进方法

基本 DE 的可改进之处

- 参数设置的敏感性
- 算法后期收敛速度较慢
- 变异策略选取困难
- 能否扩展到离散问题求解?
- 如何求解复杂优化问题（如：多模态问题、多目标问题、约束问题）？
- 对实际问题的求解

jDE：算法原理

The better values of encoded into the chromosome are able to lead to better individuals which, in turn, are more likely to survive and generate the promising offspring and, hence, propagate these better values.

Brest, J.; Greiner, S.; Boskovic, B.; Mernik, M.; Zumer, V. “Self-adapting control parameters in differential evolution: A comparative study on numerical benchmark problems," *IEEE Trans. on Evol. Comput.*, 2006, 10(6), 646-657.

jDE：具体实现

个体表示：

$$X_i = \langle \mathbf{x}_i, Cr_i, F_i \rangle = \langle x_{i,1}, \cdots, x_{i,D}, Cr_i, F_i \rangle$$

参数自适应：

$$F_i = \begin{cases} \mathrm{rndreal}_i[0.1, 1], & \mathrm{rndreal}[0, 1] < \tau_1 \\ F_i, & \mathrm{otherwise} \end{cases}$$

$$Cr_i = \begin{cases} \mathrm{rndreal}_i[0, 1], & \mathrm{rndreal}[0, 1] < \tau_2 \\ Cr_i, & \mathrm{otherwise} \end{cases}$$

JADE：算法原理

Better control parameter values tend to generate individuals that are more likely to survive and thus these values should be propagated.

Zhang, J. & Sanderson, A. C. “JADE: Adaptive differential evolution with optional external archive," *IEEE Trans. on Evol. Comput.*, 2009, 13(5), 945-958.

JADE：具体实现

1. Cr **自适应调整**:

$$Cr_i = \mathrm{rndn}_i(\mu_{Cr}, 0.1),$$

$$\mu_{Cr} = (1 - c) \cdot \mu_{Cr} + c \cdot \mathrm{mean}_A(S_{Cr}).$$

其中S_{Cr}保留了上一代中所有成功的Cr_i.

JADE：具体实现

1. Cr **自适应调整**:

$$Cr_i = \mathrm{rndn}_i(\mu_{Cr}, 0.1),$$

$$\mu_{Cr} = (1-c)\cdot\mu_{Cr} + c\cdot\mathrm{mean}_A(S_{Cr}).$$

其中S_{Cr}保留了上一代中所有成功的Cr_i.

2. F **自适应调整**:

$$F_i = \mathrm{rndc}_i(\mu_F, 0.1)$$

$$\mu_F = (1-c)\cdot\mu_F + c\cdot\mathrm{mean}_L(S_F)$$

$$\mathrm{mean}_L(S_F) = \frac{\sum_{j=1}^{|S_F|} F_j^2}{\sum_{j=1}^{|S_F|} F_j}$$

其中S_F保留了上一代中所有成功的F_i.

SaDE：算法原理

- 不同变异算子所适合求解的问题不同，且在进化不同阶段所需要的变异算子也可能不同;
- 具有互补的多算子协同自适应进化将会增加算法的普适性和求解能力;
- 不同算子在进化过程中的贡献不同，若能有效根据其贡献分配算子库中不同算子的选择概率，将能实现多算子的自适应选择;
- SaDE 通过一段时间的学习，统计不同算子的成功和失败次数，并据此计算不同算子的选择概率，可以反应不同算法针对不同问题的适应性，进而增强算法的性能.

A. K. Qin, V. L. Huang, and P. N. Suganthan, “Differential evolution algorithm with strategy adaptation for global numerical optimization," *IEEE Trans. Evol. Comput.*, vol. 13, no. 2, pp. 398 – 417, Apr. 2009.

SaDE：具体实现-算子库

- “DE/rand/1/bin"
- “DE/rand-to-best/2/bin"
- “DE/rand/2/bin"
- “DE/current-to-rand/1"

SaDE：具体实现-算子库

- “DE/rand/1/bin"
- “DE/rand-to-best/2/bin"
- “DE/rand/2/bin"
- “DE/current-to-rand/1"

SaDE：具体实现-计算选择概率

$$\begin{cases} p_k & = & \frac{S_{k,G}}{\sum_{k=1}^{K} S_{k,G}} \\ S_{k,G} & = & \frac{\sum_{g=G-LP}^{G-1} ns_{k,g}}{\sum_{g=G-LP}^{G-1} ns_{k,g} + \sum_{g=G-LP}^{G-1} nf_{k,g}} + \epsilon \end{cases}$$

其中，LP为学习期，$ns_{k,t}$和$nf_{k,t}$和是第k个算子在第t代时成功和失败子个体的次数。

Rank-DE：算法原理

- 自然界中存在一个普遍现象：优秀个体对整个群体具有更大的影响;
- 在 DE 中，如果能合适考虑优秀个体更多参与到进化搜索中将会增强算法的 exploitation 能力，进而加快算法的收敛速度.

- W. Gong & Z. Cai, “Differential evolution with ranking-based mutation operators," *IEEE Trans. Cybern.*, vol. 43, no. 4, pp. 2066 - 2081, 2013.
- W. Gong, Z. Cai & D. Liang, “Adaptive ranking mutation operator based differential evolution for constrained optimization," *IEEE Trans. Cybern.*, vol. 45, no. 4, pp. 716-727, 2015.

Rank-DE：具体实现

- 排序值分配：种群中个体按照某种准则从好到坏排序

$$R_i = NP - i, \quad i = 1, 2, \cdots, NP$$

- 选择概率计算：好的个体获得更高的选择概率

$$p_i = \frac{R_i}{NP}, \quad i = 1, 2, \cdots, NP$$

- 以“DE/rand/1"为例：

$$\mathbf{v}_i = \mathbf{x}_{r_1} + F \cdot \left(\mathbf{x}_{r_2} - \mathbf{x}_{r_3}\right)$$

Rank-DE：具体实现

Algorithm 1: Ranking-based Vector Selection for “DE/rand/1" Mutation

Input: The base vector index i

Output: The selected vector indexes r_1, r_2, r_3

Randomly select $r_1 \in [1, NP]$;
while $\text{rndreal}[0, 1) > p_{r_1}$ ***or*** $r_1 == i$ **do**
    Randomly select $r_1 \in [1, NP]$;
Randomly select $r_2 \in [1, NP]$;
while $\text{rndreal}[0, 1) > p_{r_2}$ ***or*** $r_2 == r_1$ ***or*** $r_2 == i$ **do**
    Randomly select $r_2 \in [1, NP]$;
Randomly select $r_3 \in [1, NP]$;
while $r_3 == r_2$ ***or*** $r_3 == r_1$ ***or*** $r_3 == i$ **do**
    Randomly select $r_3 \in [1, NP]$;

DVR-DE：算法原理

- Preserving the successful directions of perturbation in the differential mutation of DE can be very effective;
- Probabilistic reuse of the previously successful search directions can discover more promising solutions in future generations with a high probability, while still preserving considerable population diversity.

A. Ghosh, S. Das, A. Das, and L. Gao, “Reusing the past difference vectors in differential evolution—A simple but significant improvement," *IEEE Trans. Cybern.*, 2019, in press.

DVR-DE：具体实现

Algorithm 1 DE/rand/1/bin With DVR

1: Initialize a population $\boldsymbol{P}_0$ of Np D-dimensional vectors within the pre-specified upper and lower limits. Set the initial difference vector archive $\boldsymbol{A}_0 = \varnothing$.

2: **for** $G = 1$ to G_{max}

  for $i = 1$ to Np **do**

    if $G < 2$ **do**

      Sample a base vector $\vec{X}_{r1,G}$ and a difference vector $\vec{\Delta}_{i,G}$ from the current population $\boldsymbol{P}_G$.

      Generate the mutant vector as $\vec{V}_{i,G} = \vec{X}_{r1,G} + F.\vec{\Delta}_{i,G}$.

    end do

      Generate a uniform random number $rand_i \in [0,\ 1]$.

        if $(rand_i \le p)$

         Randomly sample a difference vector $\vec{\Delta}_k$ from $\boldsymbol{A}_G$ and set $\vec{\Delta}_{i,G} = \vec{\Delta}_k$.

        else sample $\vec{\Delta}_{i,G}$ and a base vector $\vec{X}_{r1,G}$ from the current population $\boldsymbol{P}_G$.

        end if

        Generate the mutant vector as $\vec{V}_{i,G} = \vec{X}_{r1,G} + F.\vec{\Delta}_{i,G}$.

    end if

    Generate the trial vector $\vec{U}_{i,G}$ by using binomial crossover as shown in (3).

    if $f\left(\vec{U}_{i,G}\right) \le f\left(\vec{X}_{i,G}\right)$ **do**

      Replace $\vec{X}_{i,G}$ in $\boldsymbol{P}_G$ with $\vec{U}_{i,G}$.

      Store the difference vector $\vec{\Delta}_{i,G}$ into archive $\boldsymbol{A}_G$.

    end if

  end for

    if $|\boldsymbol{A}_G| > Np$ **do**

     Discard $|\boldsymbol{A}_G| - Np$ difference vectors, selected uniform at random from $\boldsymbol{A}_G$ and set $\boldsymbol{A}_{G+1} = \boldsymbol{A}_G$.

    end if

end for

3: Return the best individual $\vec{X}_{best,G_{max}}$ from the final population and the corresponding objective function value.

求解复杂问题的 DE

- 求解多模态问题：
  B. Qu, P. Suganthan, & J. Liang, "Differential evolution with neighborhood mutation for multimodal optimization," *IEEE Trans. Evol. Comput.*, 16(5): 601-614, 2012.
- 求解约束优化问题：
  E. Mezura-Montes, C. A. Coello Coello, J. Velaquez-Reyes, & L. Munz-Daila, "Multiple trial vectors in differential evolution for engineering design," *Eng. Optim.*, vol. 39, no. 5, pp. 567-589, 2007.
- 求解多目标优化问题：
  Robič, T. and Filipič, B. "DEMO: Differential evolution for multiobjective optimization," *EMO-05*, 520 - 533, 2005.
- 求解离散优化问题：
  L. Wang, Q. Pan, P. Suganthan, et al, "A novel hybrid discrete differential evolution algorithm for blocking flow shop scheduling problems," *Computers and Operations Research*, 509-520, 2008.

6. 小结

算法简介

算法基本流程

简单示例

代表性改进方法

小结

6. 小结

本章小结

1. 差分进化算法简介
2. 差分进化算法流程
3. 差分进化算法示例
4. 代表性改进方法

6. 小结

思考

在网上下载差分进化算法源程序，读懂，并尝试独立实现。思考以下问题：

1. 调整算法的三个参数，看是否对结果有影响？
2. 选择不同的差分变异算子，看是否对结果有影响？

6. 小结

进一步资料

- R. Storn, K. Price, “Differential evolution – A simple and efficient heuristic for global optimization over continuous spaces,” *J. of Global Optim.* 1997, 11 (4): 341 - 359.
- S. Das, P. N. Suganthan, “Differential evolution: A survey of the state-of-the-art,” *IEEE Trans. on Evol. Comput.* 2011, 15 (1): 4 - 31.
- K. Price and R. Storn, “Differential evolution homepage,” http://www1.icsi.berkeley.edu/˜storn/code.html, 2018.

6. 小结

DE 书籍

- K.V. Price, R.M. Storn, and J.A. Lampinen (2005), Differential Evolution: A Practical Approach to Global Optimization, Springer-Verlag.
- V. Feoktistov (2006), Differential Evolution: In Search of Solutions, Springer-Verlag.
- U.K. Chakraborty (2008), Advances in Differential Evolution, Springer-Verlag.
- G. Onwubolu and D. Davendra (2009), ‘Differential Evolution: A Handbook for Global Permutation-Based Combinatorial Optimization, Springer-Verlag, Berlin Heidelberg.
- J. Zhang and A.C. Sanderson (2009), Adaptive Differential Evolution, Springer-Verlag.

Thank you!

AUTHOR: GONG, Wenyin
ADDRESS: School of Computer Science,
China University of Geosciences,
Wuhan, 430074, China
E-MAIL: wygong@cug.edu.cn
HOMEPAGE: http://www.escience.cn/people/wygong