

## crowdis 参考资料

**概要:** 拥挤距离计算 (Crowding Distance)。

**描述:**

该函数实现了目标空间中各点的拥挤距离计算。

**语法:**

```
dis = crowdis(ObjV, levels)
dis = crowdis(ObjV, levels, enhanceFlag)
```

**详细说明:**

ObjV 表示种群个体的目标函数值矩阵。

levels 是 Numpy array 类型行向量，代表种群个体的非支配排序分级，如 1、2、3 等，未被分级的个体的 levels 值为 Inf。

enhanceFlag 是可选参数，表示是否采用增强拥挤距离计算策略，以便后面筛选出分布更加均匀的个体，缺省或为 None 时，默认 enhanceFlag 的值为 False。

**特别注意:**

本函数是根据传入参数 ObjV 来计算拥挤距离的，但无论最小化目标还是最大化目标，都对拥挤距离计算没有影响，因此在调用本函数前，不需要对传入的 ObjV 乘上“maxormins” (最大最小化标记)。

**应用实例:**

考虑一个双目标优化问题，设种群规模为 4，这 4 个个体的目标函数值如下：

(4,1),(3,2),(2,3),(1,4)

调用 ndsortESS 或 ndsortDED 等非支配排序算法对其进行非支配排序（详见相应的参考资料），可得这 4 个个体对应的帕累托分级为：

$$levels = \begin{pmatrix} 1 & 1 & 1 & 1 \end{pmatrix}$$

调用 crowdis 即可计算出各个个体所在目标空间点的拥挤距离：

```
import numpy as np
import geatpy as ea
ObjV = np.array([[4,1],[3,2],[2,3],[1,4]])
[levels, criLevel] = ea.ndsortESS(ObjV) #
    非支配排序，得到帕累托分级levels以及临界层所在的级数criLevel
dis = ea.crowdis(ObjV, levels) # 计算拥挤距离
```

得到的拥挤距离为：

$$dis = \begin{pmatrix} \text{inf} & 1.3333334 & 1.3333334 & \text{inf} \end{pmatrix}$$

**参考文献:**

[1] Deb K , Pratap A , Agarwal S , et al. A fast and elitist multiobjective genetic algorithm: NSGA-II[J]. IEEE Transactions on Evolutionary Computation, 2002, 6(2):0-197.