

## ndomindeb 参考资料

**概要:** Deb 非支配排序 (nondominated-sorting of Deb)。

**描述:**

该函数利用 Deb 提出的非支配排序法 (Deb et al,2000) 来对种群个体按非支配关系进行分级，函数返回种群个体的适应度以及分级情况 (1,2,3,4……)，其中处在第 1 级的个体即为该种群的帕累托最优个体。若传入 exIdx 参数，则会对非可行解的个体进行排除。

**语法:**

```
[FitnV, levels] = ndomindeb(ObjV)
[FitnV, levels] = ndomindeb(ObjV, needLevel)
[FitnV, levels] = ndomindeb(ObjV, needLevel, LegV)
```

**详细说明:**

ObjV 是种群个体的目标函数矩阵，每一行对应一个个体，每一列对应一个目标。  
needLevel 是一个可选参数，表示要对种群个体最大分多少级，缺省或为 None 时默认是 5。  
LegV 是一个保存着个体对应的可行性的列向量，0 表示该个体是非可行解，1 表示是可行解。  
返回参数中，FitnV 是一个 array 类型的列向量，每一行对应于一个个体的适应度。  
levels 是一个 array 类型的行向量，每一行对应于一个个体所在的等级，如 1,2 或 3 等等。  
**注意:** Geatpy 的非支配排序均遵循最小化目标的约定。

**特别注意:**

本函数是根据传入参数 ObjV 来进行非支配排序的，且遵循“最小化目标”的约定，因此在调用本函数前，需要对传入的 ObjV 乘上'maxormin'(最大最小化标记) 来让其符合约定。但是，由于返回的是 FitnV，它与 ObjV 在含义上无关了，因此不需要对其乘上'maxormin' 进行还原。

**应用实例:**

考虑一个两个目标的优化问题，设种群规模为 20，这 20 个个体的目标函数值如下：  
(9,1),(7,2),(5,4),(4,5),(3,6),(2,7),(1,9),(10,3),(8,5),(7,6),(5,7),(4,8),(3,9),(10,5),(9,6),(8,7),  
(7,9),(10,6),(9,7),(8,9)  
使用 Deb 非支配排序法计算该种群个体进行分级：

```
ObjV =
    np.array([[9,1],[7,2],[5,4],[4,5],[3,6],[2,7],[1,9],[10,3],[8,5],
[7,6],[5,7],[4,8],[3,9],[10,5],[9,6],[8,7],[7,9],[10,6],[9,7],[8,9]])
[FitnV, levels] = ndomindeb(ObjV)
```

得到的个体分级情况如下：  
1 1 1 1 1 1 1 2 2 2 2 2 2 3 3 3 3 4 4 4

再次提醒的是，levels 是 numpy 的 array 类型的行向量，行向量和行矩阵的关系在“Geatpy 数据结构” 章节中有详细描述。可以使用 print(levels.shape) 来查看 levels 的规格，可以发现结果为 (20,)，若是行矩阵的话，输出的是 (1,20) 而不是 (20,)。