crtfld 参考资料

概要: 区域描述器生成函数。

描述:

该函数根据输入的参数生成区域描述器 FieldD 或 FieldDR。

语法:

1) FieldDR = crtfld(ranges)
2) FieldDR = crtfld(ranges, borders)
3) FieldD = crtfld(ranges, borders, precisions)
4) FieldD = crtfld(ranges, borders, precisions, codes)
5) FieldD = crtfld(ranges, borders, precisions, codes, scales)

详细说明:

区域描述器的结构较为复杂，该函数提供了一个自动化的方法来生成区域描述器。

ranges 是一个 2 行 n 列的矩阵 (注意是 numpy 的 array 类型)，代表 n 个变量的边界范围。

其中第 0 行代表各个变量的下界；第 1 行是代表各个变量的上界。

borders 是一个 2 行 n 列的矩阵，代表 n 个变量是否包含区间的边界，0 表示不包含该边界，1 表示包含。

其中第 0 行代表是否包含各个变量的下界；第 1 行是代表是否包含各个变量的上界。

precisions 是可选参数，是一个一维 list，表示变量的编码精度，其元素必须是非负的，缺省或为 None 时默认元素全为 0。例如其中一个元素是 4，表示对应变量的编码可以精确到小数点后 4 位。

注意：该“精度”仅是对采用二进制或格雷编码而言的，若使用的是实值编码，则 precisions 不再表示其精度，而是作为表示实值连续型变量的边界精度。例如：执行 FieldDR = crtfld(ranges, borders)，由于 precisions 缺省，因此 FieldDR 表示的控制变量为离散实值。若想要表示连续型的实值控制变量，则需要设置 precisions 的元素为任意正整数即可。例如：precisions = [1, 1]。

上面所说的“边界精度”是指当控制变量范围不包含边界时，crtfld 函数会根据 precisions 把范围收缩一定的程度。上面所说的“设置 precisions 的元素为任意正整数”就体现在这里。比如精度设为 1 时，而边界为 0（即不包含范围边界），则 crtfld 函数会把范围往里收缩 0.1。

codes 是可选参数，是一个一维 list，表示变量是用什么方式编码的，例如其中一个元素为 0 时表示对应的变量是采用标准二进制编码，1 表示格雷编码，当 codes 缺省或为 None 时，函数将生成只有 2 行的区域描述器 FieldDR。

scales 是可选参数，是一个一维 list，指明变量用的是算术刻度还是对数刻度，其元素为 0 或 1。例如其中一个元素是 0，表示对应变量是采用算术刻度；1 表示采用对数刻度。缺省或为 None 时默认其元素全为 0。

因此特别注意：crtfld 函数会自动对变量的 ranges 范围以及 borders 边界进行处理，最终返回一个规范的区域描述器。

有关区域描述器的概念详见 bs2int 以及 crtip 的参考资料。

应用实例:

例 1：下面欲创建包含变量x_1, x_2 的整数值种群，2 个变量的区间范围分别是 [-3,5) 和 [2,10), 分别使用对数刻度的标准二进制编码和算术刻度的格雷编码，创建一个区域描述器来描述它。

```
x1 = [-3, 5]                    # 自变量1的范围
x2 = [2, 10]                    # 自变量2的范围
b1 = [1, 0]                     # 自变量1的边界
b2 = [1, 1]                     # 自变量2的边界
codes = [0, 1]                  # 变量编码方式，分别采用二进制和格雷编码
precisions = [0, 0]             # 各变量的精度，0表示精确到个位
scales = [1, 0]                 # 采用算术刻度
ranges = np.vstack([x1, x2]).T # 生成自变量的范围矩阵
borders = np.vstack([b1, b2]).T # 生成自变量的边界矩阵
# 调用crtfld函数生成区域描述器
FieldD = crtfld(ranges, borders, precisions, codes, scales)
```

$$\text{FieldD} = \begin{pmatrix} 3 & 4 \\ -3 & 2 \\ 4 & 10 \\ 0 & 1 \\ 1 & 0 \\ 1 & 1 \\ 1 & 1 \end{pmatrix}$$

解析：crtfld 函数对变量的区间范围以及边界进行了处理，返回的区域描述器中x_1 的边界值已经被合理地调整为 [-3,4]，相应地，原本应该为 [1,0] 的$lbin$ 已经被修改为 [1,1]。

FieldD 中，第一行的$lens$ 参数是根据变量的范围计算得到的。本例中修正后的变量范围是 [-3,4] 和 [2,10]，意味着分别至少需要用 3 位和 4 位的二进制数来进行编码，因此$lens$ 参数的值是 [3 4]。

例 2：欲创建一个包含变量x_1, x_2, x_3 的实数值种群，3 个变量的区间范围分别是 (-2.5,2), [3,5], [-4.8,3.6)，分别精确到小数点后 2、3、4 位。创建一个描述它的区域描述器：

```
x1 = [-2.5, 2]                  # 自变量1的范围
x2 = [3, 5]                     # 自变量2的范围
x3 = [-4.8, 3.6]                # 自变量3的范围
b1 = [0, 0]                     # 自变量1的边界
b2 = [1, 1]                     # 自变量2的边界
b3 = [1, 0]                     # 自变量3的边界
precisions =[3, 3, 4]           # 各变量的精度，3表示精确到小数点后3位
ranges = np.vstack([x1, x2, x3]).T # 生成自变量的范围矩阵
borders = np.vstack([b1, b2, b3]).T # 生成自变量的边界矩阵
# 调用crtfld函数生成区域描述器
FieldDR = crtfld(ranges, borders, precisions)
```

$$\text{FieldDR} = \begin{pmatrix} -2.99 & 3.0 & -4.8 \\ 1.99 & 5.0 & 3.5999 \end{pmatrix}$$