

稳录取分数线（PAVA + 线性插值）提示词手册

任务

根据按初试分数“分段统计”的复试与录取数据，计算**稳录取分数线**：使“单点录取率首次 \geq 阈值 T （默认 0.75）”的**最小分数**。

注意：只看分段**条件概率**，不要用总体累计比例替代；允许分段样本不均衡与比例失衡。

输入格式（其一）

1. **表格**：score_range, candidates, admitted（或另给 rate%）
 - score_range 支持：230-239、230~239、230-239，或单点 262
2. **纯文本**：每行 区间，复试X人，录取Y人（应自动抽取数字）
3. **可选参数**： $T=0.75$ （阈值）、min_bin_n=8（最小样本提醒阈）、ci=0.95（Wilson 置信）

步骤要求

1. **解析分段**
 - 对每行区间取下界 lo、上界 hi；若单点则 hi=lo。
 - 段中心 $c = (lo+hi)/2$ ；样本 $n=candidates$ ；录取 $k=admitted$ ；比例 $p = k/n$ （若已给百分比则校验并使用）。
2. **排序**：按分数升序，形成点列 (c_i, n_i, k_i, p_i) 。
3. **单调约束（PAVA 等增回归）**
 - 若出现 $p_i < p_{i-1}$ ，把相邻“违背单调”的块合并：
 - 合并样本： $N = \sum n_i$ ；合并录取： $K = \sum k_i$ ；**新比例**： $\hat{p} = K/N$ （样本数加权平均）；
 - **代表横坐标**（用于插值）： $\hat{x} = (\sum n_i \cdot c_i)/N$ （样本数加权中心 / 重心）。
 - 迭代直到整体**非降**。不要用简单“前缀最大”直接抬高比例。
4. **查找阈值穿越点**
 - 设阈值 T （默认 0.75）。在校正后的序列中，找到第一次满足 $p_1 < T \leq p_2$ 的相邻两点/块 $(x_1, p_1) \rightarrow (x_2, p_2)$ 。
 - 若所有点 $p < T$ ：输出“无稳分”，并给 p_{max} 及其对应分数。
 - 若首点即 $p \geq T$ ：阈值退化到复试线附近；建议同时给 $T'=0.80/0.85$ 的结果作参考。

5. 线性插值（细化到单分）

- $t = (T - p1)/(p2 - p1)$;
- **稳分小数**: $x^* = x1 + t \cdot (x2 - x1)$; **报告整数稳分**: $\text{round}(x^*)$ (四舍五入)。
- 同步给出触发两点 $(x1, p1)$ 与 $(x2, p2)$ 。

6. 稳健性与置信区间

- 对触发块 $p2$ (以及相邻合并前的原段) 给 **Wilson 区间** (默认为 95%)。如下限 $< T$, 标注“边界略不稳”; 可提示“与更高分相邻段合并验证”。
- 对极小样本段 (如 $n < \text{min_bin_n}$) 给醒目提示。

7. 对照口径（可选）

- **分箱常数法（保守）**: 不插值, 直接取首次 $p \geq T$ 的**该段下界**; 仅作对照输出。

8. 输出结构

- 主结果: 稳录取分数 (整数)、 x^* (小数)、阈值 T ;
- 触发点: $(x1, p1) \rightarrow (x2, p2)$; 是否发生 **PAVA 合并** (列出被合并的原段与合并后 (\hat{x}, \hat{p}));
- 置信: 相关段/块的 n, k, p 与 Wilson 区间;
- 对照: 分箱常数法的保守稳分;
- 异常: 如“无稳分”“阈值退化”“上限未达 T ”。

最小示例（示意）

输入：

```
201-209, 复试30, 录取1
210-219, 复试36, 录取10
220-229, 复试23, 录取19
230-239, 复试13, 录取9
240-249, 复试1, 录取1
251,      复试1, 录取1
T=0.75
```

关键输出要点：

- 先用 **PAVA** 合并 220-229 与 230-239 \rightarrow 得 $(\hat{x} \approx 228.61, \hat{p} = 0.7778)$;
- 与 $(215, 0.2778)$ 插值得 $x^* \approx 227.84 \rightarrow$ **稳分=228**;
- Wilson 提示: 给出触发块的区间下限/上限并标注是否“边界略不稳”;
- 对照: 分箱常数法保守稳分 = **220**。

Excel 计算片段（假设触发两行是 i 与 i+1；阈值放在 E1）

$$= A_{\{i\}} + (E_{\{i\}} - B_{\{i\}}) / (B_{\{i+1\}} - B_{\{i\}}) * (A_{\{i+1\}} - A_{\{i\}})$$

Python 计算片段 (核心逻辑, 伪代码)

```
rows = [(lo, hi, n, k), ...] # 低到高
pts = []
for lo,hi,n,k in rows:
    c = (lo+hi)/2
    p = k/n
    pts.append([c, n, k, p])

# PAVA
stack = []
for c,n,k,p in pts:
    stack.append([c,n,k,p])
    while len(stack)>=2 and stack[-1][3] < stack[-2][3]:
        c2,n2,k2,p2 = stack.pop()
        c1,n1,k1,p1 = stack.pop()
        N, K = n1+n2, k1+k2
        C = (n1*c1 + n2*c2)/N # 加权中心
        P = K/N # 加权比例
        stack.append([C, N, K, P])

# 找首次跨越阈值
T = 0.75
for (x1,_,_,p1),(x2,_,_,p2) in zip(stack, stack[1:]):
    if p1 < T ≤ p2:
        x_star = x1 + (T-p1)/(p2-p1) * (x2-x1)
        steady = round(x_star)
        print(steady, x_star, (x1,p1), (x2,p2))
        break
```

提示：使用浏览器打印 (Ctrl/Cmd+P) 时选择 A4 纸张即可导出为 PDF。