

SSP-MMC 练耳算法工程落地说明书

v1.0 (Cold Start)

Papajoey

2026 年 1 月 27 日

目录

1 系统核心逻辑 (The Core Loop)	2
1.1 步骤 A: 准备阶段 (Pre-Computation)	2
1.2 步骤 B: 交互与反馈 (Interaction)	2
1.3 步骤 C: 物理引擎更新 (Update State)	2
1.4 步骤 D: 查表定计划 (Lookup & Schedule)	2
2 物理引擎公式 (DHP Model Physics)	2
2.1 核心常数定义	2
2.2 状态更新逻辑	3
2.2.1 1. 更新难度 (d_{next})	3
2.2.2 2. 更新半衰期 (h_{next})	3
3 查表策略 (Lookup Navigation)	3
3.1 查表算法	3
3.2 极小值修正 (Sanity Check)	3
4 初始数据表 (Initialization)	4
5 开发自检清单	4

1 系统核心逻辑 (The Core Loop)

整个算法在 Demo 中表现为一个无限循环。每次用户进行训练时，系统必须严格按照以下顺序执行：

1.1 步骤 A：准备阶段 (Pre-Computation)

在用户开始听声音之前，根据数据库中该音程的当前状态，计算回忆概率。

1. **读取状态**：获取当前音程的 **难度** (d) 和 **半衰期** (h)。
2. **计算时间间隔** (Δt)：当前时间 - 上次复习时间 (单位：天)。
3. **计算回忆概率** (P)：

$$P = 2^{-\frac{\Delta t}{h}} \quad (1)$$

注： P 用于衡量用户当前的记忆残留，将作为后续更新公式的重要参数。

1.2 步骤 B：交互与反馈 (Interaction)

- 播放音程声音。
- 获取用户反馈结果 (r)： $r = 1$ (听对) 或 $r = 0$ (听错)。

1.3 步骤 C：物理引擎更新 (Update State)

调用 **DHP 模型函数** (见第 2 节)，根据 r 和 P 计算出新的记忆状态：

$$(d, h) \xrightarrow{\text{Update}} (d_{\text{next}}, h_{\text{next}})$$

1.4 步骤 D：查表定计划 (Lookup & Schedule)

调用 **SSP 查表策略** (见第 3 节)，根据新的状态规划下一次复习：

1. 根据 d_{next} 锁定对应的 CSV 文件 (例如 $d = 5.2 \rightarrow \text{ivl-5.csv}$)。
2. 根据 h_{next} 锁定 CSV 中的行。
3. 读取 **最优间隔** (Interval)。
4. **写入数据库**：更新 d, h 以及 $\text{Next_Review} = \text{Now} + \text{Interval}$ 。

2 物理引擎公式 (DHP Model Physics)

请在代码中实现名为 `update_state` 的函数。这是系统的核心驱动力。

2.1 核心常数定义

为了适应音乐听觉训练 (比背单词更依赖高频复习)，我们需要引入缩放因子 K 。

常数名	值	说明
K_{scale}	0.5	关键参数。用于压低记忆增长速度。
D_{max}	10.0	难度上限
$W_{penalty}$	0.1	听错时难度增加步长

2.2 状态更新逻辑

输入参数: d (难度), h (半衰期), P (概率), r (结果 0/1)

2.2.1 1. 更新难度 (d_{next})

- 如果 $r = 0$ (错): $d_{next} = \min(10.0, d + 0.1)$
- 如果 $r = 1$ (对): $d_{next} = \max(1.0, d - 0.05)$

2.2.2 2. 更新半衰期 (h_{next})

情况 A: 用户听对了 ($r = 1$) 使用 回忆成功增强公式 (引入 K_{scale}):

$$\text{Gain} = K_{scale} \cdot e^{3.81} \cdot d_{next}^{-0.534} \cdot h^{-0.127} \cdot (1 - P)^{0.970} \quad (2)$$

$$h_{next} = h \cdot (1 + \text{Gain}) \quad (3)$$

情况 B: 用户听错了 ($r = 0$) 使用 遗忘衰减公式:

$$\text{Retention} = e^{-0.041} \cdot d_{next}^{-0.041} \cdot h^{0.377} \cdot (1 - P)^{-0.227} \quad (4)$$

$$h_{next} = h \cdot \text{Retention} \quad (5)$$

3 查表策略 (Lookup Navigation)

3.1 查表算法

输入: d_{next}, h_{next}

- 文件定位: 找到文件名数字与 d_{next} 差值最小的 CSV 文件。
- 行定位: 遍历该 CSV 的第一列 (代表 h), 找到与输入 h_{next} 数值最接近的那一行。
- 读取结果: 读取该行 第二列 的数值, 即为 Optimal_Interval (单位: 天)。

3.2 极小值修正 (Sanity Check)

```

1 if h_next < 0.1: # 如果半衰期小于 2.4 小时
2     next_interval = 0 # 强制设为 0 (代表"立即"或"10分钟后")
3 else:
4     next_interval = lookup_from_csv(d_next, h_next)

```

4 初始数据表 (Initialization)

Demo 数据库初始化时的经验参数：

Item ID	音程	初始难度 d_0	初始 h_0 (天)
P8	纯八度	1.0	0.20
P5	纯五度	2.0	0.10
P4	纯四度	3.0	0.05
TT	三全音	6.0	0.02
m2	小二度	7.0	0.01

5 开发自检清单

- **时间单位：**公式里的 Δt 必须是 “天”。1 小时 = $1/24 \approx 0.0417$ 。
- **K_{scale} 参数：**确保 $r = 1$ 时乘了 0.5。
- **边界检查：**计算 $(1 - P)$ 时确保 $P \leq 0.999$ 。