基于艾宾浩斯记忆曲线的拟合函数分析和自适应单词记忆程序

黎兴东，齐箫剑，吴睿政

摘要：近些年外语学习的氛围越来越浓厚，背单词成为必要需求。但是大多数传统背单词的方法效率低下，于是着重考虑结合艾宾浩斯记忆曲线记忆单词。本文前半篇研究了艾宾浩斯记忆曲线的性质，验证了艾宾浩斯曲线规律，并用Matlab对艾宾浩斯记忆曲线进行拟合，提出新的优化记忆曲线模型。艾宾浩斯记忆曲线表明了记忆的规律，对寻找长期记忆的方法提供了启示。本文后半篇提出了一种能够较好记忆单词的复习策略，并用java开发出一种自适应背诵和考察单词记忆的程序。

关键词：艾宾浩斯记忆曲线；单词记忆；拟合函数；java；Matlab

Analysis of Fitting Function and Adaptive Word Memory Program Based on Ebbinghaus Memory Curve

Li Xingdong, Qi Xiaojian, Wu Ruizheng

Abstract: In recent years, the atmosphere of foreign language learning has become more and more intense, and memorizing words has become a necessary requirement. However, most of the traditional methods of memorizing words are inefficient, so we focus on memorizing words in combination with the Ebbinghaus memory curve. In the first half of this article, the nature of the Ebbinghaus memory curve was studied, the law of the Ebbinghaus curve was verified, and the Ebbinghaus memory curve was fitted with Matlab to propose a new optimized memory curve model. The Ebbinghaus memory curve shows the rules of memory and provides inspiration for finding long-term memory. The second half of this article proposes a review strategy that can better memorize words, and uses java to develop a program for adaptive recitation and review of word memory.

Keywords: Ebbinghaus memory curve; word memory; fitting function; java; Matlab

**引言**

随着社会的发展，人们对英语学习的需求越来越浓厚。词汇是英语学习的基本构成单位，对英语学习是非常重要的，然而单词“记不住”“记不牢”却成为常态。在信息化社会，传统纸质记单词的方法不够方便且效率很低，用电子词典记单词在一定程度上改善了记单词的效率，但却仍需要自己规划记单词的安排日程。英语词汇繁多而负责，人们投入大量时间和精力来背单词却往往并没有达到理想的记忆效果，如何有效地记忆词汇已经成了近年来的研究热点。

艾宾浩斯记忆曲线对我们的算法设计具有启示作用。艾宾浩斯记忆曲线是德国著名心理学家艾宾浩斯于1885年在对遗忘现象进行系统研究后提出的。他认为遗忘的进程不是均衡的，不是固定的一天丢掉几个，在记忆的最初阶段遗忘的速度很快，后来逐渐减慢了，到了相当长的时候，几乎就不再遗忘了，即遵循“先快后慢”的原则。观察这条遗忘曲线，你会发现,学得的知识在一天后，如不抓紧复习,就只剩下原来的25%。随着时间的推移,遗忘的速度减慢，遗忘的数量也就减少。继艾宾浩斯之后很多研究人员做过类似实验，一些研究人员总结出有助于增强记忆的8个记忆周期：短时记忆周期分为5分钟、30分钟、12小时，长期记忆周期分为1天、2天、4天、7天、15天。这些记忆周期的总结增强了人们在实际应用中的可操作性。（百度百科）

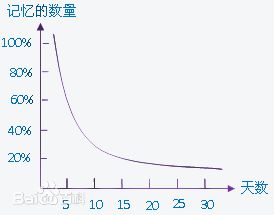


图 1 艾宾浩斯记忆曲线

艾宾浩斯记忆曲线给我们提供了一种记忆模型，经前人研究和理论分析，它为用户提供了记忆的规划日程，可以有效地提高背单词的效率。但是在适应每个用户的不同知识水平和使用习惯上，此模型仍可优化。本文通过分析艾宾浩斯模型和有关算法，旨在为用户提供自适应性记忆规划模型，采用艾宾浩斯记忆曲线刻画用户对每个单词的记忆情况，在单词遗忘的临界点提醒并安排用户复习，根据测试结果再次调整曲线模型，帮助其提高背单词的效率。我们开发了拥有此功能的智能程序，为开发相应软件打下基础。

本文正文分为三个部分。第一个部分对艾宾浩斯记忆曲线模型进行分析，第二部分为艾宾浩斯记忆曲线的函数拟合，第三部分介绍我们自己结合艾宾浩斯记忆曲线利用java语言设计出的自适应背单词

[1 验证艾宾浩斯记忆曲线模型的实验 2](#_Toc33744432)

[1.1 实验目的 2](#_Toc33744433)

[1.2 实验过程 2](#_Toc33744434)

[1.3 实验结果 2](#_Toc33744435)

[1.4 艾宾浩斯记忆曲线拟合函数 2](#_Toc33744436)

[2 艾宾浩斯记忆曲线的函数拟合 3](#_Toc33744437)

[2.1 艾宾浩斯拟合曲线 4](#_Toc33744438)

[2.2 单一模型下的拟合情况 5](#_Toc33744439)

[2.2.1 斜率为负数的线性函数。Y=-aX+b ；（a, b>0）。 5](#_Toc33744440)

[2.2.2 反比例函数及其幂函数。 5](#_Toc33744441)

[2.2.3 指数模型。 6](#_Toc33744442)

[2.2.4 b-cX1/a为例的幂函数。 8](#_Toc33744443)

[2.3 多模型的混合拟合 8](#_Toc33744444)

[2.4 对于长时间记忆的新情况： 9](#_Toc33744445)

[3 自适应背单词程序 13](#_Toc33744446)

[3.1 User类的基本属性和一些主要方法： 13](#_Toc33744447)

[3.2 GlossaryBook类的基本属性和一些主要方法： 14](#_Toc33744448)

[3.3 “addword”方法是备注用户用于向自建单词本中添加新条目的方法。 15](#_Toc33744449)

[3.4 相应的：“deleteWord”方法是用于删除条目的方法 15](#_Toc33744450)

[3.5 向Examination传递词书内容的方法 16](#_Toc33744451)

[3.6 Examination类的基本属性和一些主要方法： 16](#_Toc33744452)

[3.7 Examination类的学习方法： 17](#_Toc33744453)

[3.8 Examination类的测试方法： 18](#_Toc33744454)

[3.9 Examination类的复习方法： 19](#_Toc33744455)

[总结 20](#_Toc33744456)

## 验证艾宾浩斯记忆曲线模型的实验

### **实验目的**

探究在单词记忆中的人类遗忘过程，验证艾宾浩斯遗忘曲线是否符合大致遗忘规律。

### **实验过程**

在实验中，以组内同学A为实验对象进行记忆和遗忘实验。在材料选择上，我们选取基本不认识的专八单词作为实验材料，以避免已认识单词对最终实验结果造成影响。我们将实验时间设为每天上午10:00，避免因在一天不同时间段对实验结果造成影响。其他环境也尽量保持相同。让A同学学习一定量单词，在达到学习标准后，再间隔不同时间对A同学进行测试，以之后测试仍能记住的单词量作为量化数据。

实验第一天，A同学记忆五百个单词，将其使用时间记为T，之后将间隔不同时间的学习时间记为t，我们以第一次能记住的单词量记为Q，之后每次间隔能记住的单词量记为q可用以下公式表示：

Q——第一次学习后记住的单词量

q——之后每次重新测试记住的单词量

R——记忆保持量

R=q／Q x 100%

不断增加学习间隔时间，30分钟、1天、2天、4天、6天、10天、15天，计算不同时间间隔的记忆保持量。

### 实验结果

表格 1 不同时间间隔下的记忆保持度

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 时间间隔 | 30分钟 | 1天 | 2天 | 4天 | 6天 | 10天 | 15天 |
| 记忆保持量 | 60% | 33.7% | 30% | 28% | 25% | 23% | 21% |

将实验结果拟合，我们能发现其大致符合艾宾浩斯记忆曲线。

通过曲线可以了解到在记忆初期遗忘速度最快而且遗忘内容也最大，但在实验中期遗忘速度和数量都逐渐趋于平缓。在之前做过类似研究的其他研究人员那里也能大体得到这个结论。

### 艾宾浩斯记忆曲线拟合函数

艾宾浩斯采用对数函数来描述遗忘曲线，给出的数学公式为公式（1）:

但是，艾宾浩斯在论文中没有提到采用其他函数来拟合遗忘曲线，即没有比较其他函数模型对于遗忘实验数据的拟合效果。这使得后来越来越多的心理学家对遗忘曲线的拟合提出了更多的数学模型，先后提出了 100 多种函数来拟合遗忘曲线，其中最有名的有对数函数、指数函数、幂函数，对应的数学公式如表 1 所示:

|  |  |
| --- | --- |
| 候选函数 | 数学公式 |
| 对数函数 |  |
| 指数函数 |  |
| 幂函数 |  |

Chris和Robert的研究指出幂函数比其他候选函数更适合用来描述艾宾浩斯记忆曲线，得到的数学公式为公式（2）

其中，m为记忆保持量，无量纲；M为记忆系数常量，无量纲；T为时间间隔，单位为min；b为记忆衰减系数，无量纲。【3】

## 艾宾浩斯记忆曲线的函数拟合

艾宾浩斯记忆曲线是由艾宾浩斯与1885年提出的一种记忆曲线模型，其通过对无意义单词的记忆进行调查总结出了人的遗忘曲线规律。他利用如CAK,BUW等无意义但是由双辅音加上一个元音组成的人造单词测试了人的遗忘曲线，【1】，【2】，为了还原艾宾浩斯的成果我们小组也设计了一个辅助记忆软件KeepIt并且利用Excel表绘制出了对应的曲线图。

在完成实验前我们进行了大概的实验结果预测。我们不难发现，任何记忆都是会随着时间推移而逐渐遗忘，并且其满足一个规律即记忆效果的下降是先快后慢的。例如，在记忆完成后的第10天，我们的记忆效果并不会随着增加一小时而下降过多。在对艾宾浩斯遗忘曲线有一定的主观认识之后我们设计了一个用java写的项目来测试在背六级单词时所对应的曲线于艾宾浩斯记忆曲线的区别。

为了方便统计，我们测试了100个单词以下是我们的测试环节

图片包含 屏幕截图, 监视器, 室内, 计算机

描述已自动生成如图运行程序时可以选择注册或者登录已有的信息，当用户选择注册时，类似于QQ的注册，用户会被分配一个ID，并且可以自己设置密码。不同用户的信息是不同的。这样就可以避免交叉而导致的错误。将信息写入文件中也可以方便下一次读取避免不同的文件产生不必要的交互。

随后用户可以添加自己的词书，我们目前是加入了四级和六级的词书可供用户选择。记忆的方式如下图所示

图片包含 屏幕截图, 监视器

描述已自动生成

keepIt一次性显示两个单词，用户既可以一次性显示自己今天的所有背诵目标也可以显示。

### 艾宾浩斯拟合曲线

利用keepit我们随机记忆了100个单词，并以自己为实验对象进行了测试。经过三十天的比对我们形成了excel表格

图片包含 屏幕截图

描述已自动生成

图片包含 屏幕截图

描述已自动生成同时讲记录下的背诵情况导入matlab中生成了如下的函数图像。

就此我们发现该散点分布的情况基本符合我们最初的判断即人的遗忘情况为，快速下降后下降速度会逐渐放缓，以下是实现该效果的具体代码。

>> x=xlsread('D:\test.xlsx',1,'A1:V1');

>> y=xlsread('D:\test.xlsx',1,'A2:V2');

>> plot(x,y);

但是该图像是折线图并不好预测接下来的趋势利用拟合工具我们可以大概绘出图像。在绘图前我们需要对艾宾浩斯曲线有一个基本认识，也就是确定表达式的基本样式。所以问题的关键在于确定函数的模型。以下是对于模型的猜测以及排除。

在这里我们使用matlab中的inline函数进行拟合，订出回归系数以后用nlinfit随后再使用plot函数画图并与原图像进行对比。

### 单一模型下的拟合情况

#### 斜率为负数的线性函数。Y=-aX+b ；（a, b>0）。

图片包含 屏幕截图

描述已自动生成

如上图所示，就本次拟合来讲效果并不好，而且并不符合导数为负增函数的基本特点，而且截距过低，并不符合我们的预期，所以斜率为负数的线性函数是最容易排除的情况。那么接下来的猜想思路也就无非三种。

#### **反比例函数及其幂函数**。

对于这类函数，也即y=a\*x-b,我们也可以用matlab推算他们的系数推算的最终结果如下图所示，同时展示了反比例函数与线性函数的区别，在进行拟合我也发现了一些问题，由于反比例函数在x=0时为正无穷，而我们给定的x轴坐标是从0开始考虑的所以会出现在x=0时无法拟合的情况，nlinfit()函数会报错，其报错内容为。

1. 错误使用 nlinfit>checkFunVals (line 649)
2. The function you provided as the MODELFUN input has returned Inf or NaN values.
4. 出错 nlinfit (line 251)
5. **if** funValCheck && ~isfinite(sse), checkFunVals(r); end

在发现这个问题的时候我也查阅了一些资料发现问题的症结就是x=0的情况，为了拟合成功我采用将函数整体小范围左移，也就是在自变量上加一个常数，但是这个常数要足够小，最终我的函数模型为f(x)=beta1(1)\*(x+0.01).^beta1(2)。回归系数我根据目前已有的艾宾浩斯曲线的系数估计为图片包含 屏幕截图

描述已自动生成[20,-0.1]。结果还是比较理想的，我们可以注意到，上图的蓝线与黄线的拟合程度还是比较高的，而相比红线也就是线性函数，反比例的幂函数也更接近。最终得到的系数为38.2785064145845 -0.202406376366046，我们取到两位三位有效数字，其结果为Y=38.2\*X-0.202。

#### 指数模型。

指数函数也有类似反比例函数的性质，即导数为负增函数。但是指数函数有几个问题：第一，指数函数在x趋于0时的值并不是无穷，第二，指数函数在x趋于正无穷时也不可能无限小。不过在这一点上，指数函数与反比例函数有一定的共同点，但与我们接下来说的x1/a（a>0）这一种函数模型有本质区别。

那么对于指数模型我们需要先假定以下它的基本表达式。1.我们可以添加截距，y=b+ae-cx。2，没有截距的情况y=ae-bx。

对于第二种情况我们计算的结果为：

图片包含 地图, 文字

描述已自动生成

上图的红线为所求，matlab给出的拟合系数为beta2=[30.332846973702360,0.003145576854313]。取到三位有效数字为Y=30.3\*e-0.00315X。对于第一种情况也就是有截距的情况在做拟合。其结果为22.0431379118950，取到三位有效数字为Y=22.0+77.8\*e-1.87X。图片包含 屏幕截图

描述已自动生成

上图的绿线为我们所求的答案，不难发现，指数函数在添加斜率以后也有一个快速下降段，matlab的拟合结果也反映出确实可以到100，但是问题在于，指数函数的基本性质，指数函数是有下界的这次我们的拟合系数为77.7830294878332 1.86688684139831 22.0431379118950那么该函数永远都不会小于22，这与遗忘的根本性质完全不符，并不是我们希望得到的解。而如果不添加截距，那么在x趋于0时其结果又不能于我们期望相符，所以指数函数我们小组认为并不是艾宾浩斯记忆曲线的曲线函数，但有可能在一定部分确实指数函数存在。

#### b-cX1/a为例的幂函数。

考虑到幂函数的基本特点也就是导数为负增函数，那么其相反数就应为减函数，所以这种添加了截距的幂函数也应该在考虑范围内，但是这种幂函数在x增大的过程中总会出现函数值小于0的情况，这种情况应该予以考虑，也就是完全遗忘会不会在记忆大量内容中出现。

经过matlab拟合其结果为：

图片包含 屏幕截图

描述已自动生成

这一次的拟合结果是单一模型中最好的，基本上完全和我们的预期相同，其系数为100.022229780746 70.0198923611180 29.8717077615478，取三位有效数字为Y=100-70.0X1/29.9

所以可以说艾宾浩斯记忆曲线的函数中，这种函数应该是占主流的，其比例也会更大。

### 多模型的混合拟合

在考虑了上述的五种情况后我们小组也对艾宾浩斯曲线有了一定的猜测，我认为艾宾浩斯曲线前人的研究可能有一定的局限性，如此复杂的函数实际上应该是有多个函数部分组成的。所以我尝试通过更复杂的函数模型来寻找更贴合的表达式。

在学习傅里叶级数时，我们知道傅里叶对于每个三角函数都给分配了一个系数，我们也可以尝试类似的操作结合指数函数，幂函数，和反比例函数的优点，互相补充。为此为做了一个函数模型。

Y=beta(1)\*exp(-beta(2)\*x)+beta(3)+beta(4)-beta(5)\*x.^(1/beta(6))+beta(7)\*(x+0.01).^(-beta(8))

也就是将上文的三个函数结合在一起，最后用matlab拟合得到的beta矩阵为：

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| -175.370827306030 | 3.00921261747387e-05 | 109.394832968419 | 90.3948334587397 | 1.94367574807728 | 3.24829097533459 | 9.68009867626878 | 0.446263916113097 |

其图像为如上所示，这次拟合效果非常好，基本上完全和我们原函数相同，所以我们小组的猜想也得到了证明，艾宾浩斯记忆曲线用单一函数是很难拟合的，其结果为不同函数模型的和。

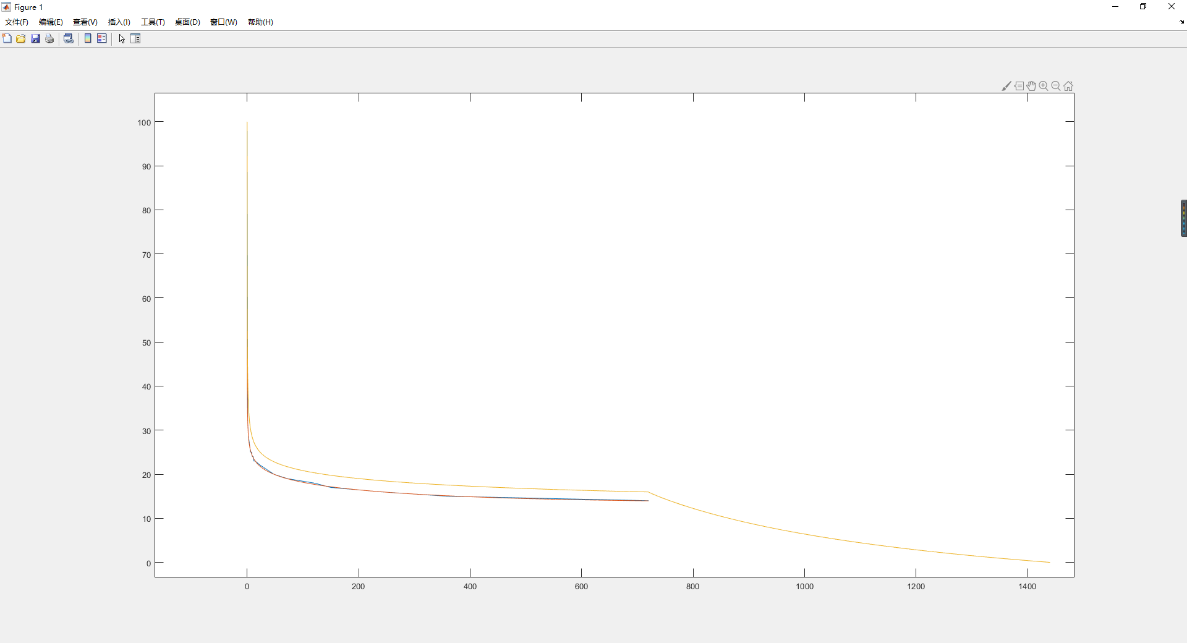
图片包含 屏幕截图

描述已自动生成

### 对于长时间记忆的新情况：

我们都知道艾宾浩斯记忆曲线是针对于百分百记忆之后的情况但他没有考虑一个问题也就是记忆遗忘问题，在记忆过程当中，遗忘曲线就已经在发生了，我们只是不断地巩固记忆才确保了艾宾浩斯类型的记忆，才会有这样的记忆曲线，而大部分情况下我们是做不到这样的记忆的，在一个小时的记忆过程中一定会发生遗忘，对此我提出了新的模型。

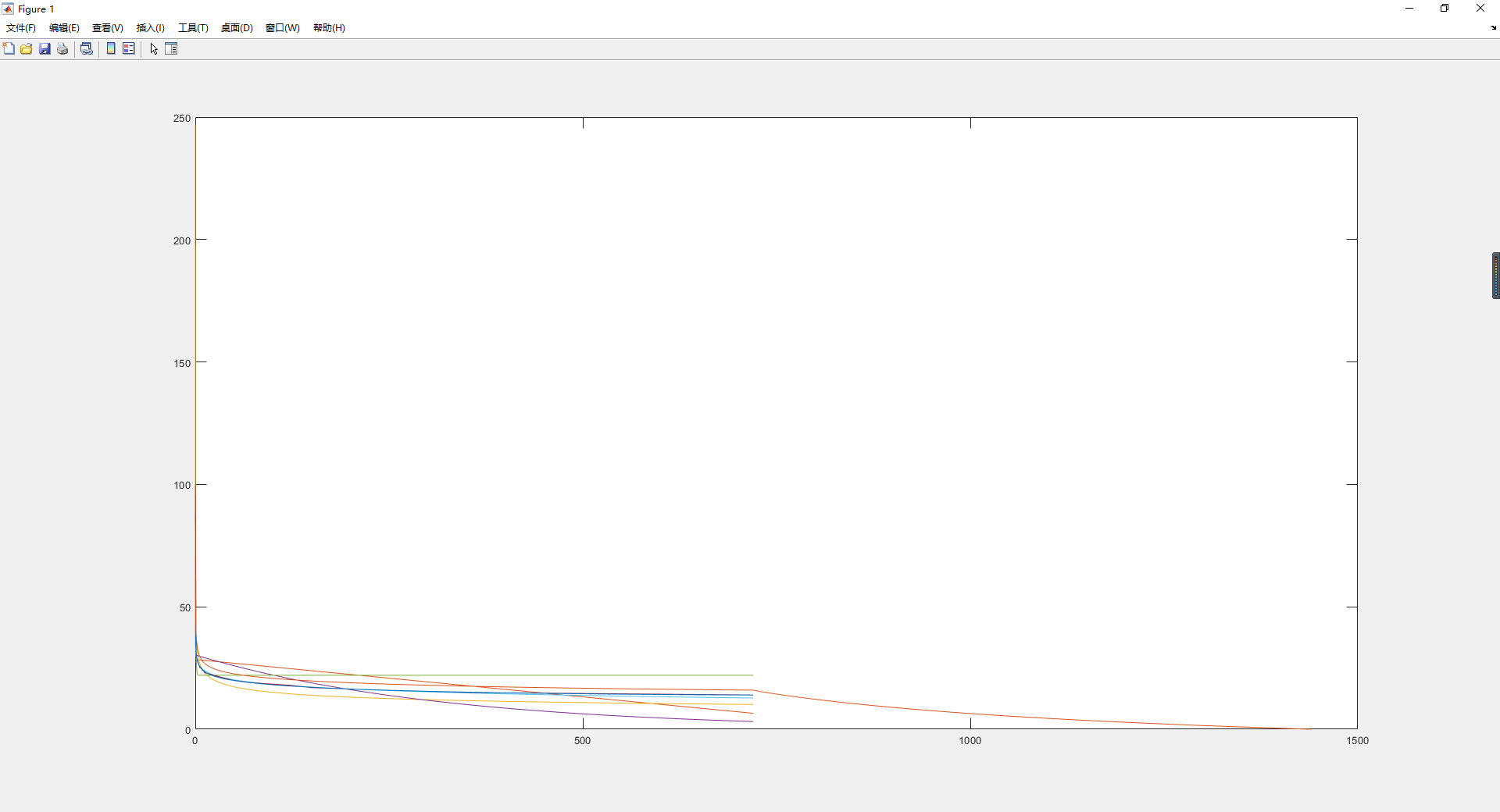
在记忆中的遗忘实际上就是艾宾浩斯记忆曲线于阶跃函数的卷积，所以我们考虑一个“门函数“在x从0到1时该函数为1，其余为0。与我们已得的艾宾浩斯记忆曲线拟合函数卷积，就可以得到我们在一般学习过程中的记忆情况。



如上图所示，我采取计算在一小时内的背诵所对应的遗忘情况，不难发现，实际上真正的记忆曲线并不是艾宾浩斯记忆曲线因为他忽略了遗忘过程中的记忆，新的记忆曲线模型利用卷积可以有效避免这个问题。可以考虑在一段时间内的背诵效果不难发现实际上是优于艾宾浩斯记忆曲线的。我也尝试对这个函数进行拟合。

最终我还是利用之前的函数模型求系数得到的结果为。





以上是所有此次拟合实验的函数图像。

两次拟合所涉及的代码：

Fit2.m

1. clear;
2. x=xlsread('D:\matlab\bin\paper\test.xlsx',1,'A1:W1');
3. y=xlsread('D:\matlab\bin\paper\test.xlsx',1,'A2:W2');
4. plot(x,y);
5. hold on;
6. myfunc=**inline**('beta(1)\*exp(-beta(2)\*x)+beta(3)+beta(4)-beta(5)\*x.^(1/beta(6))+beta(7)\*(x+0.01).^(-beta(8))','beta','x');
7. **try**
8. beta0=[20,0.01,20,1,1,1,1,1];
9. **catch**
10. end
11. beta=nlinfit(x,y,myfunc,beta0);
12. k=0:0.01:720;
13. t=beta(1)\*exp(-beta(2)\*k)+beta(3)+beta(4)-beta(5)\*k.^(1/beta(6))+beta(7)\*(k+0.01).^(-beta(8));
14. plot(k,t);
15. hold on;
16. s=ones(1,72001);
17. w=ones(1,144001);
18. **for** i=1:144001
19. w(i)=1/i;
20. end
21. con1=conv(t,s).\*w;
22. r=0:0.01:1440;
23. myfunc1=**inline**('beta1(1)\*exp(-beta1(2)\*r)+beta1(3)+beta1(4)-beta1(5)\*r.^(1/beta1(6))+beta1(7)\*(r+0.01).^(-beta1(8))','beta1','r');
24. beta1=nlinfit(r,con1,myfunc1,beta0);
25. plot(r,con1);
26. hold on;

fit1.m

1. clear;
2. x=xlsread('D:\matlab\bin\paper\test.xlsx',1,'A1:W1');
3. y=xlsread('D:\matlab\bin\paper\test.xlsx',1,'A2:W2');
4. plot(x,y);
5. hold on;
6. %猜测1：线性函数
7. myfunc1 = **inline**('beta(1)\*x+beta(2)','beta','x');
8. beta0 = [0.01,100];
9. beta = nlinfit(x,y,myfunc1,beta0);
10. k=0:0.0001:720;
11. t=beta(1)\*k+beta(2);
12. plot(k,t);
13. hold on;
14. clear beta0;
15. %反比例函数的幂函数
16. myfunc2 = **inline**('beta1(1)\*(x+0.01).^beta1(2)','beta1','x');
17. beta0 = [10,-0.1];
18. beta1 = nlinfit(x,y,myfunc2,beta0);
19. t1=beta1(1)\*k.^beta1(2);
20. plot(k,t1);
21. hold on;
22. clear beta0;
23. %指数函数
24. myfunc3=**inline**('beta2(1)\*exp(-beta2(2)\*x)','beta2','x');
25. beta0=[50,0.01];
26. beta2=nlinfit(x,y,myfunc3,beta0);
27. t2=beta2(1)\*exp(-beta2(2)\*k);
28. plot(k,t2);
29. hold on;
30. clear beta0
31. %有截距的指数函数
32. myfunc4=**inline**('beta3(1)\*exp(-beta3(2)\*x)+beta3(3)','beta3','x');
33. beta0=[50,0.01,50];
34. beta3=nlinfit(x,y,myfunc4,beta0);
35. t3=beta3(1)\*exp(-beta3(2)\*k)+beta3(3);
36. plot(k,t3);
37. hold on;
38. clear beta0;
39. %幂函数
40. myfunc5=**inline**('beta4(1)-beta4(2)\*x.^(1/beta4(3))','beta4','x');
41. beta0=[1,1,1];
42. beta4=nlinfit(x,y,myfunc5,beta0);
43. t4=beta4(1)-beta4(2)\*k.^(1/beta4(3));
44. plot(k,t4);
45. hold on;
46. clear beta0;

## 自适应背单词程序

我们使用Java语言来将我们所研究的记忆曲线模型运用到实例当中。我们通过导入外部单词本的方式让用户得以通过本项目进行基本的英语单词背诵、复习、检测的过程，模拟学生平时的正常学习状态，并尝试在这一学习过程中运用到记忆曲线相关的知识，让整体学习记忆更为高效。

项目构建于本地，已经拥有基本的用户登录、注册功能，可以做到对用户姓名、性别、生日甚至学习自勉座右铭的收集与记录。我们基于本地数据文件，对用户数据相关的健壮性进行了足够的优化，能坐到剔除重复数据、对常见输入错误的勘误提示。完好的数据收集与记录功能将有益于项目今后的进一步发展。

本项目的核心功能是帮助用户在电脑端背诵英语单词，并且实现符合记忆曲线的复习与考试流程。其次我们根据参考资料，找寻了当代大学生需求量最大的托福、四级考试、六级考试等英语能力测验的核心词汇以及与国际学院专业相关的一些专业术语用作我们初始的词库。与用户数据类似的，目前我们使用外部文档的方式进行词库保存。目前，我们已经支持让用户通过键入文本的形式向数据库中输入自己想要复习的内容，并可将这些内容分门别类、整理成册，便于之后进行复习、测验等。至此，项目的应用层面更加广阔，从单纯的根据记忆曲线模型进行英语单词背诵过程拓展到了对任何文本形式的内容的背诵，例如枯燥的数学公式、繁多的古诗词句子等。

系统设计上，可以分为管理用户的User类，管理公共的和用户自己创建的词书的GlossaryBook类和根据词书和用户设计考试方案的Examination类以及前端模拟类（Main）。User类为每一个用户的学习，创建和管理账户提供了基本的方法，以及定义和存储了用户的属性和行为。

### User类的基本属性和一些主要方法：

1. **public** **class** User **implements** Serializable{
2. **static** Scanner input=**new** Scanner(System.in);
3. **static** String regex="^(?![0-9]+$)(?![a-zA-Z]+$)[0-9A-Za-z]{8,16}$";
4. **private** **static** String userListPath; //用户列表文件路径
5. **private** **static** Properties userInfo=**null**;
6. **private** **static** BufferedReader readUser=**null**;
8. **private** **static** String passwordPath; //用户密码文件路径
9. **private** **static** Properties passwordInfo=**null**;
10. **private** **static** BufferedReader readPassword=**null**;
11. **private** String name;
12. **private** String ID; //用户唯一的ID
13. **private** String password;
14. **private** String gender="Unknown";//以下为用户的其它属性
15. **private** Calendar birthday=**null**;
16. **private** String aboutMe;
17. **private** File myInfo;
18. **private** String userPath;
19. **private** String myBookPath;//folder
20. **private** String myRecordPath;//folder
21. **private** Properties bookLoader;
22. **private** GlossaryBook book; //用户选择的词书
23. **private** **int** numberOfBook; //用户拥有词书的数量
24. **private** **int** plan;
25. **private** **boolean** hasRecord;
27. **public** **void** study(Examination e,String path){
28. e.studyHelper();
29. e.saveRecord(path);
30. }
32. **public** **void** review(Examination e){
33. e.reviewHelper();
34. }
36. **public** **void** exam(Examination e,String path){
37. e.testHelper();
38. e.saveRecord(path);
39. }
41. **public** **void** viewMyRecord(Examination e){
42. e.showMyRecord();
43. }
44. }

### GlossaryBook类的基本属性和一些主要方法：

1. **public** **class** GlossaryBook **implements** Serializable {
2. **static** Scanner input=**new** Scanner(System.in);
3. **static** Properties publicBookLoader;
4. **private** **static** **int** numberOfPBook; //公共词书的数量
5. **private** **static** String publicBookPath;
6. **private** Properties bookLoader;
7. **private** HashMap<Integer,String[]> word; //单词
8. **private** User owner;
9. **private** String name;
10. **private** **int** number;
11. **private** **boolean** isPublic;
12. **private** **boolean** beingTest;

### “addword”方法是备注用户用于向自建单词本中添加新条目的方法。

1. **public** **boolean** addWord(String word, String meaning){
2. **if** (isPublic){
3. System.out.println("This book is a public book!");
4. **return** **false**;
5. }
6. **if** (word.equals("") || meaning.equals("")){
7. System.out.println("Word or meaning cannot be nothing!");
8. **return** **false**;
9. }
10. **if** (bookLoader.containsKey(word)){
11. System.out.println("This book is already has "+"'"+word+"'!");
12. **return** **false**;
13. }
14. bookLoader.setProperty(word, meaning);
15. saveWord();
16. number++;
17. System.out.println("Addition successful!");
18. **return** **true**;
19. }

### 相应的：“deleteWord”方法是用于删除条目的方法

1. **public** **boolean** deleteWord(String word){
2. **if** (isPublic){
3. System.out.println("This book is a public book!");
4. **return** **false**;
5. }
6. **if** (number==0){
7. System.out.println("This book does not have any words!");
8. }
9. **if** (!bookLoader.containsKey(word)){
10. System.out.println("This book is not include "+"'"+word+"'!");
11. **return** **false**;
12. }
13. bookLoader.remove(word);
14. saveWord();
15. number--;
16. System.out.println("Delete successful!");
17. **return** **true**;
18. }

### 向Examination传递词书内容的方法

1. **public** HashMap<Integer, String[]> getWordMap() {
2. HashMap<Integer, String[]> studyMap=**new** HashMap<>();
3. Set<String> set=bookLoader.stringPropertyNames();
4. **int** index=0;
5. **for** (String s : set){
6. index++;
7. String[] content={s,bookLoader.getProperty(s)};
8. studyMap.put(index,content);
9. }
10. **return** studyMap;
11. }
12. }

目前本项目新增条目需要提供”word”和”meaning”两个参数，即单词和中文释义。事实上用户可以灵活自我调整这两项的内容，从而达到随意添加自身所需复习内容的目的。如：”word”参数填入数学公式，”meaning”中填入”公式”即可。

在接下来的开发中，我们将不断扩充支持背诵的外部资源库，并且优化用户自我添加背诵内容时的用户体验。我们也在积极尝试将已有的自然语言识别模块嵌入我们的项目中，实现用户拍照上传笔记、软件识别笔记并记录。用户原本只能随性复习的内容在导入本项目后，就能够按照记忆曲线模式、有效率的进行复习。

复习和考试模式能够记录学生目前的学习进度。本程序对外部拓展的单词册以及用户自己上传的用作背诵的文本资料中的内容条数都有单独的记录，并且在用户每一次通过本项目进行学习时所背诵的单词及单词数量都有记录。对于这些参数，我们做了充分的利用。例如，“studyhelper”类能够调用参数，提示用户目前的学习进度。

### Examination类的基本属性和一些主要方法：

1. **public** **class** Examination **implements** java.io.Serializable{
2. **static** Scanner input=**new** Scanner(System.in);
3. **private** User user;
4. **private** GlossaryBook book;
5. **private** TreeMap<Integer,String[]> bookContent;//Get the book content
6. **private** HashMap<Integer,Integer> wrongRecord;//For review and show the overall record!
7. **private** **int**[] faultTracker;//record the order you learned
8. **private** **int** studyTracker;
9. **private** **int** frequency;
10. **private** **int** number;//How many word a day/How many word for a test
11. **private** **int** learned;
12. **private** **boolean** hadStudy;
13. **private** **int** incorrect;
14. **private** **double** accuracy;

### Examination类的学习方法：

根据用户设定的计划和频率向用户设计一个学习策略。用户一旦设立了，一旦用户学习完成后重新进入，系统就会根据用户此前学习的数据，优化出一轮新的学习方案。

1. **public** **void** studyHelper(){
2. number=user.getPlan();
3. **int** all=bookContent.size();
4. **if** (learned==all){
5. System.out.println("You already learned this book! Let's study it again!(Or this book does not have enough words)");
6. learned=0;
7. studyTracker=0;
8. }
9. **if** (!hadStudy)
10. studyTracker=learned;
11. **if** (learned+number>all){
12. number=all-learned;
13. System.out.println("You only left "+number+" words to learn!");
14. }
16. System.out.println("Ready? Go!");
17. Set<Integer> set=bookContent.keySet();
19. **for** (**int** index=learned+1;index<=learned+number;index++){
20. System.out.print(Arrays.toString(bookContent.get(index))+"      ");
21. **if** (index%2==0){
22. System.out.println("按Enter键继续学习！");
23. input.nextLine();
24. }
25. }
26. learned+=number;
27. number=user.getPlan();
28. System.out.println("Today's goal completed!");
29. hadStudy=**true**;
30. }

Examination类通过设立faultTracker这一参数，对用户的测试过程中错误的次数进行记录，这一参数是我们记忆曲线模型中的重要组成部分。我们在算法中针对用户错误频次高的背诵单元，在复习时进行优先级、重复次数加权计算中会针对性调高其权重，达到更好的记忆效果。

### Examination类的测试方法：

该测试方法通过用户传入的词书数据，记录用户的学习范围和记录，向用户进行测试，题型目前仅局限于选择题，以后我们会加入更多的测试方式，以完善我们对用户记忆曲线形成的参数需求。

1. **public** **void** testHelper(){
2. **if** (learned<20){
3. System.out.println("You cannot attend a test if you do not learn more than 20 words!");
4. **return**;
5. }
6. **int** wrong=0;
7. **double** percent=0;
8. System.out.println("You got "+(learned-studyTracker)+"to test! Ready? Go!");
9. **int** all=bookContent.size();
10. Random random=**new** Random();
11. **for** (**int** i=studyTracker+1;i<=learned;i++){//Start a test
12. String[] content=bookContent.get(i);
13. **int**[] temp=**new** **int**[3];//Generate 3 fault answers index
14. System.out.println("--------------------------------------------");
15. **do** {
16. **for** (**int** j = 0; j < 3; j++)
17. temp[j] = i + random.nextInt(all - i + 1);
18. } **while** (temp[0] == i || temp[1] == i || temp[2] == i || temp[0] == temp[1] || temp[0] == temp[2] || temp[1] == temp[2]);
19. String[] problem=**new** String[4];
20. **for** (**int** j=0;j<3;j++){//Get fault answers Chinese meaning
21. problem[j]=bookContent.get(temp[j])[1];//Other answers
22. }
23. problem[3]=content[1];//The right answer
24. //            System.out.println(i+" "+problem[3]);
25. HashMap<String,String> answers=**new** HashMap<>();
26. answers.put(problem[(i+1)%4],"a");
27. answers.put(problem[(i+3)%4],"b");
28. answers.put(problem[(i+4)%4],"c");
29. answers.put(problem[(i+2)%4],"d");
30. System.out.println(content[0]);
31. System.out.print("A. "+problem[(i+1)%4]+"   ");
32. System.out.print("B. "+problem[(i+3)%4]);
33. System.out.println();
34. System.out.print("C. "+problem[(i+4)%4]+"   ");
35. System.out.print("D. "+problem[(i+2)%4]);
36. System.out.println();
37. System.out.println("Please choose your answer!");
38. String answer=input.nextLine();
39. **if** (answers.get(problem[3]).equals(answer.toLowerCase())){//get the represent of right answer and compare with enter
40. System.out.println("You got it");
41. **if** (faultTracker[i]>0)
42. faultTracker[i]--;
43. }
44. **else** {
45. System.out.println("Sorry! The right answer is "+answers.get(problem[3]));
46. faultTracker[i]++;
47. incorrect++;
48. wrong++;
49. }
50. }
51. percent=1-(**double**)wrong/(learned-studyTracker);
52. System.out.println("Finish! Your got "+wrong+" incorrect, the accuracy is "+percent);
53. hadStudy=**false**;
54. }

### Examination类的复习方法：

通过学习和考试功能收集用户的数据，结合我们的算法，按照用户记忆曲线的变化的单词错误的频率，给予最人性化的复习方案。

1. **public** **void** reviewHelper(){
2. System.out.println("Those are the words you do not remember well:");
3. System.out.println("(If you test and got it incorrect, the fault time will increase one! But if you got it right");
4. System.out.print(", it will decrease one until 0, accordingly)\n");
5. **if** (!haveFault())
6. System.out.println("You got these words perfect by now! Don't worry about it! Go for fun!");
7. **else** {
8. **for** (**int** i=0;i<faultTracker.length;i++) {
9. **if** (faultTracker[i]!=0){
10. System.out.println(i+"."+Arrays.toString(bookContent.get(i))+"  "+faultTracker[i]+" times!");
11. }
12. }
13. }
14. }

总结： 本文分析了艾宾浩斯记忆曲线的性质和特点，即记忆保持量呈下降走势并基本满足“先快后慢”的趋势，并得到有助于增强记忆的8个记忆周期。我们通过实验测试绘制得到与艾宾浩斯基本一致的记忆曲线，在前人的经验基础上用matlab中的inline函数对其进行拟合，订出回归系数以后用nlinfit随后再使用plot函数画图并与原图像进行对比，试验了线性函数、反比例函数、幂函数、指数函数以及其混合函数的拟合，最终我们得出艾宾浩斯记忆曲线用单一函数很难拟合，其结果为不同函数模型的和。在此基础上，我们考虑到记忆过程中本身的遗忘问题，认为在记忆中的遗忘实际上就是艾宾浩斯记忆曲线于阶跃函数的卷积，并提出了新的记忆模型，结果也证明新模型的记忆效果优于艾宾浩斯记忆曲线。我们由此得到启发并设计了我们自己的单词记忆程序，用户可以创建id，并根据自己的需要选取单词书，程序将帮助用户在电脑端背诵英语单词，并且实现符合记忆曲线的复习与考试流程，用户也可随时查看自己的学习进度。

然而我们的研究还有所不足，鉴于本应用还处于一个研究的阶段以及使用群体相对是少数，没有足够的数据支撑，目前仍在试运行阶段，用户数据通过序列化的方式将数据写入本地文件实现保存。相信有一天能建立起我们的数据库，优化我们的算法，完善我们的功能，为用户提供更为优质的学习评估和帮助。

我们研究小组的几位成员参加北京邮电大学“雏燕计划”的项目也是关于记忆曲线的内容。我们之后的研究方向是：更加细化项目算法，让复习单词的过程更加贴近记忆曲线以达到更好的应用效果；目前我们的项目只是用于实验算法的第一个测试版本，今后我们将进行前端界面的优化、做图形化界面，将其搭建在服务器上、尝试通过微信小程序等方式进行线上推广，真正产生应用价值。

1. Xing C X，Gao F R ，Zhan S N ，eta1．A Collab-orative filtering recommendation algorithm in-corporate with user interest change．Journal of Computer Research and Development，2007，44(2)：296～301．(邢春晓，高风荣，战思南等．适应用户兴趣变化的协同过滤推荐算法．计算机研究与发展，2007，44(2)：296～3O1)．
2. Zhang Y C，Liu Y Z． A Collaborative filtering algorithm based on time period partition． The proceeding of 3 International Symposium on Intelligent Inform ation Technology and Security Informatics，2010，777～ 780．
3. 蔡乐，熊万墙，孙晓光 基于遗忘曲线的自适应词汇记忆模型 1007-757X（2016）05-0016-04。