实验二：卷积码编码及解码算法设计

519021910162 鲁为涛

一、实验原理

1.1 卷积码编码原理

卷积码是一种差错控制编码，由 P.Elias 于 1955 年发明。因为数据与二进制多项式滑动相关故称卷积码。卷积码在通信系统中应用广泛，如 IS-95，TD-SCDMA，WCDMA，IEEE 802.11 及卫星等系统中均使用了卷积码。它将输入的 k 个信息比特编成 n 个比特输出，特别适合以串行形式进行传输，时延小。卷积码编码器的一般形式如图 1 所示。

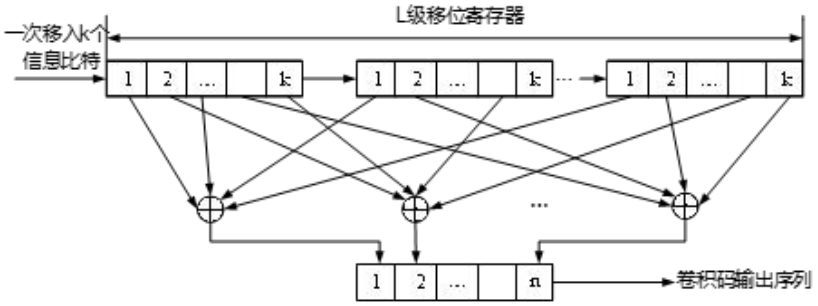

图 1 卷积码的框图形式

若以（n,k,m）来描述卷积码，其中 k 为每次输入到卷积编码器的 bit 数，n 为每个 k 元组码字对应的卷积码输出 n 元组码字，m 为编码存储度，也就是卷积编码器的 k 元组的级数，

编码速率为 k/n，称 m+1= K 为编码约束度 m 称为约束长度。卷积码将 k 元组输入码元编成 n 元组输出码元，但 k 和 n 通常很小，特别适合以串行形式进行 传输，时延小。与分组码不同，卷积码编码生成的 n 元组元不仅与当前输入的 k 元组有关，还与前面 m-1 个输入的 k 元组有关，编码过程中互相关联的码元个数为 n*m。卷积码的纠错性能随 m 的增加而增大，而差错率随 N 的增加而指数下降。在编码器复杂性相同的情况下，卷积码的性能优于分组码。

卷积码有多种表示方法，如生成多项式、状态图、网格图等，不同卷积码的表示方式（状态图等）也不相同。

1.2 Viterbi 算法译码原理

Viterbi 译码算法（简称 VA 算法）是由 Viterbi 在 1967 年首先提出的，是一种针对卷积码的最大似然译码算法。Viterbi 译码算法不是在网格图上依次比较所有的可能路径，而是接受一段，计算、比较一段，保留最有可能的路径，从而达到整个码序列是一个最大似然序列。Viterbi 算法主要由路径度量的"加比选"运算、度量的更新、路径的更新、最大似然路径的回溯过程组成，如图 2 所示。

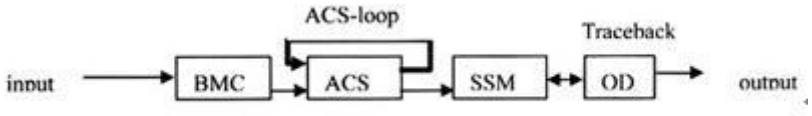

图 2 Viterbi 译码算法流程图

从理论而言，Viterbi 译码是根据接收序列在码的格图上找出一条与接收序列距离（或其他量度）为最小的一种算法。它和运筹学中求最短路径的算法相类似。若接收序列为 R=(10100101100111），译码器从某个状态，例如从状态 a出发，每次向右延伸一个分支（对于 l<L，从每个节点出发都有 2=2 种可能的延伸，其中 L 是信息序列段数，对 l≥L，只有一种可能），并与接收数字相应分支进行比较，计算它们之间的距离，然后将计算所得距离加到被延伸路径的累积距离值中。对到达每个状态的各条路径（有 2=2 条）的距离累积值进行比较，保留距离值最小的一条路径，称为幸存路径（当有两条以上取最小值时，可任取其中之一），对给定 R 的估值序列为=(10111）。这种算法所保留的路径与接收序列之间的似然概率为最大，所以又称为最大似然译码。这种译码的译码约束长度常为编码约束长度的数倍，因而可以纠正不多于（df/2）个错误。

Viterbi 译码算法优点是在码的约束比较小时，它比序列译码算法效率更高、速度更快，译码器也较简单。缺点就是随着约束长度的增加算法的复杂度增加很快。因此 Viterbi 译码一般应用在约束长度小于 10 的场合中，此外也有很多算法针对剪枝，或使用硬件结合等算法提供了效率更高的 Viterbi 译码算法。

更广义而言，在计算机科学领域中，Viterbi 算法属于一种动态规划算法，常用于寻找有向无环图（篱笆网络）当中两个点之间的最短路径（实际应用于地图导航、语音识别、分词、机器翻译等等），效率较高，若假设整个网格的宽度为 D，网格长度为 N，那么若使用穷举法整个最短路径的算法复杂度为 O（D^N）而使用这种算法的计算复杂度为 O（$N*D^2$）。若 D 与 N 都非常大，使用维特比算法的效率将会提高几个数量级。

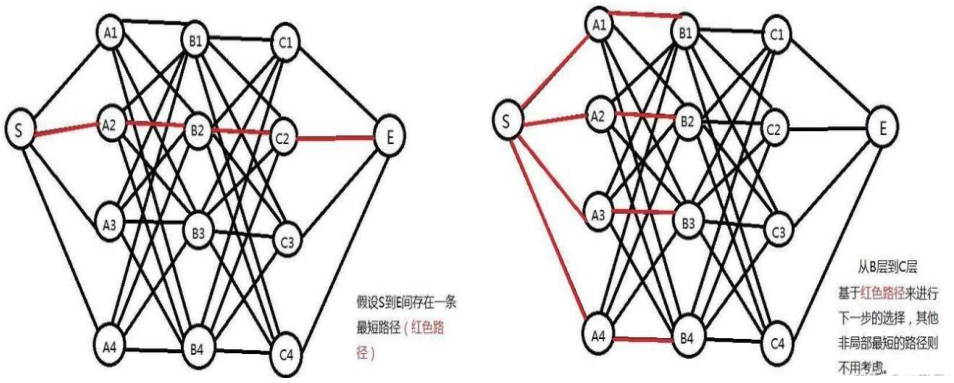

图 3　图论中的 Viterbi 算法

二、实验内容

2.1 设计卷积码编码器

首先，根据实验要求的生成多项式 $g^1(x)=1+x+x^2$ ，$g^2(x)=1+x^2$ 要求绘制编码器框图如图 4 所示：

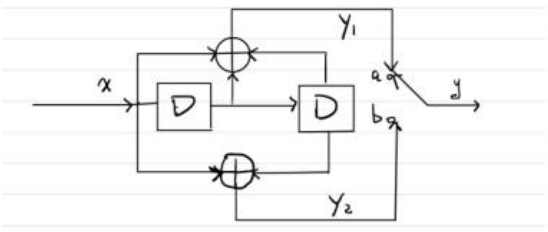

图 4　（2，1，2）卷积码编码器

将上述框图的输入输出寄存器关系使用 matlab 语言描述，代码如图 5 所示：

```
function code_data=mycc2_1_2(input_data,D)
    %code_data=zeros(1,2*length(input_data(:)));
    for i=1:length(input_data(:))
        y1=mod(input_data(i)+D(1)+D(2),2);
        y2=mod(input_data(i)+D(2),2);
        D(2)=D(1);
        D(1)=input_data(i);
        code_data(2*i-1)=y1;
        code_data(2*i)=y2;
    end
end
```

图 5　（2，1，2）卷积码编码器的 matlab 实现

2.2　信源（传输数据）设计

根据实验要求需要设计 498 位随机信息比特+2 位归零码的输入，并使用超过 1000 帧的输入数据，通过调用 matlab 的随机函数实现，如图 6 所示：

```
for k=1:total%total对应仿真的帧数
D=[0,0];%(D(i)为第i个寄存器)
input_data = [randi([0,1],[1,498]),[0,0]; %498位随机编码+2位归零码
```

input_data

1x500 double

9	490	491	492	493	494	495	496	497	498	499	500
1	1	1	1	0	1	1	1	0	0	0	0

图 6　生成 total 帧数（1200）的 500 位随机信息比特

2.3 信道编码及 QPSK 调制解调、瑞利白噪声信道设计

首先调用图 5 所示的卷积码编码函数将 500 位的信息比特生成 1000 位卷积码，然后仿真让卷积码通过实验一中设计的 QPSK 调制、瑞利信道、QPSK 解调，最后输出接收端收到的经过 QPSK 信道解调后的卷积码二进制信号。

```
code_data2=mycc2_1_2(input_data,D);
[QPSK_data,~]=QPSK_Modulation(SNR(n),code_data2);
```

```
if((dsource1==0)&(dsource2==0)),r = Ric.*s00+n;          % go through Ricean
elseif((dsource1==0)&(dsource2==1)),r = Ric.*s01+n;
elseif((dsource1==1)&(dsource2==0)),r = Ric.*s10+n;
else r = Ric.*s11+n;
end;

c00 = dot(r,s00);c01 = dot(r,s01);
c10 = dot(r,s10);c11 = dot(r,s11);
c_max = max([c00 c01 c10 c11]);

if (c00==c_max) decis1 = 0;decis2 = 0;
elseif(c01==c_max) decis1 = 0;decis2 = 1;
elseif(c10==c_max) decis1 = 1;decis2 = 0;
else decis1 = 1;decis2 = 1;
end;
```

图 7　卷积码编码、QPSK 调制、通过信道、QPSK 解调

QPSK_data 是卷积码经过 QPSK 调制，经过瑞利信道和 QPSK 解调后的信号，QPSK_Modulation 函数的具体设计可见实验 1 和代码，有两个参数分别为信道的信噪比和输入信号，包含了 QPSK 调制、解调和加性白噪声的瑞利信道。

2.4 使用 Viterbi 算法对每帧比特数据解码

首先设计 Viterb 算法，根据课程所学原理，首先绘制格图，如图 8 所示，由于共 4 个状态，仅需使用四个寄存器记录到达该状态节点的路径，由格图可知，从第三个时刻

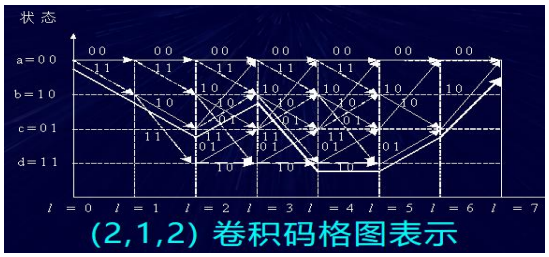

图 8　（2，1，2）卷积码格图

开始，相邻的时间节点间每个状态都有两条“出”路径到下一节点，对于下一节点而言，有两条“入”路径，因此相邻节点间共有 8 条新路径。采用课程所学的剪枝算法，每个时间节点都可以仅保留两条“入”路径中短的一条，即只需要记录四条路径即可。
因此在实际算法设计中每个时间节拍（信号输入）时都判断两条入路径保留哪一个，该逻辑的代码实现如图 9 所示，并且使用四个数组记录路径，最终保留累计汉明距离最小的路径。

```
if val_a + dis(gra(1,:), x(2*i-1:2*i)) >= val_c + dis(gra(5,:),x(2*i-1:2*i))%判断入路径aa和ca究竟是由a到a还是由c到a
    tempa = mc; %ca更近，继承之前到C路径，这两个状态a为00，c为01
    val_a_t = val_c + dis(gra(5,:),x(2*i-1:2*i));%更新此时到a状态的累计距离，为之前到c的距离和此刻c到a距离之和。
    tempa(i)=0;%记录此时的输入（aa和ca均是输入0）。最终作为解码结果
else
    val_a_t = val_a + dis(gra(1,:),x(2*i-1:2*i));
    tempa = ma;
    tempa(i)=0;
end
```

图 9　入路径的判断和剪枝的核心代码

2.5 实验结果

使用上述设计的卷积码编码器、QPSK 调制解调、瑞利信道做仿真，输入信息比特为 498 位+2 位归零码，1000 位卷积码计为 1 帧，共测试 1200 帧，用 **Viterbi** 算法对每一帧数据解码，绘制不同 SNR 下这 1200 帧的平均误比特率如图 10 所示：由实验结果可知，采用卷积编码的纠错能力很强，引入归零码后，最后寄存器状态为 00。随 SNR（dB）增加，误比特率近似指数级下降（y 轴取对数坐标）。

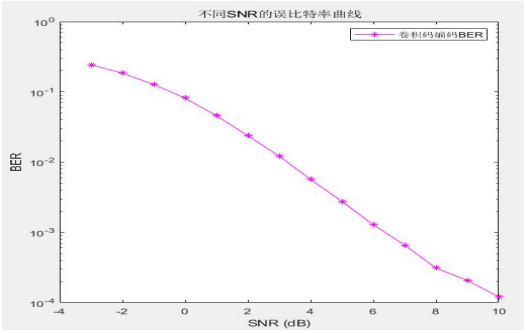

图 10　不同 SNR 下的误比特率曲线（帧数：1200）

2.6 （扩展）各函数正确性测试

Matlab 本身带有卷积码和 Viterbi 译码的函数，在编程时为了保证结果的正确性，我学习了 Matlab 库中的卷积码编码函数和译码函数的调用方法，并与自己编写的函数的运行结果做对比，为此编写了对比两个码字有多少字节重复的工具函数，测试过程如图 11 所示。（源代码中的注释部分）

```
%trellis = poly2trellis(3,[7 5]);
%code_data=convenc(input_data,trellis);
```

```
%tmp=issame(code_data,code_data2);
%decode = vitdec(code_data2,trellis,15,'term','hard');
%decode2=vitdec(QPSK_data,trellis,15,'term','hard');
```

图 11　调用 matlab 内置的编码和译码函数和运行结果对比

通过该测试方法，严格保证了我设计的 212 卷积码编码器、Viterbi 译码算法与内置的库函数运行结果完全相同，并保证了 QPSK 瑞利信道函数的准确性（高信噪比时误比特率为 0）。此外，我在查看源码时注意到内置的 Viterbi 算法传入信道的参数后可以自动估计出状态转移图的输出，而我自己编写的函数则需要手动计算出状态转移图，因此内置函数的泛用性更强。

三、与实验一无编码信道比较

3.1　实验过程及实验结果

在实验一中计算了无信道编码时仅依靠 QPSK 调制，不同 SNR 时的误比特率，在本实验中，对每一帧输入比特（500 位），不经过卷积码编码，直接通过 QPSK 调制解调并计算所有帧的平均比特率，与 2.5 的实验结果做对比，取 2000 帧，最终的实验结果如图 12 所示：

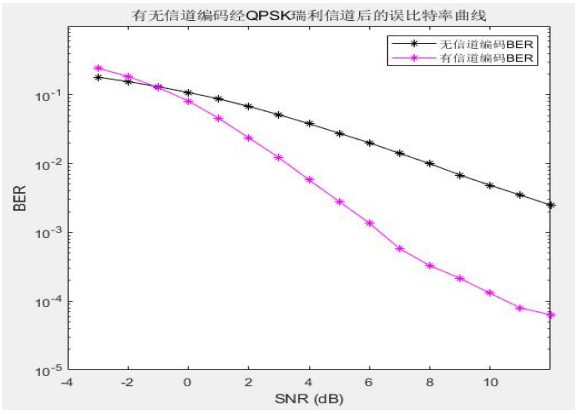

图 12　有无信道编码情况下的误比特率曲线

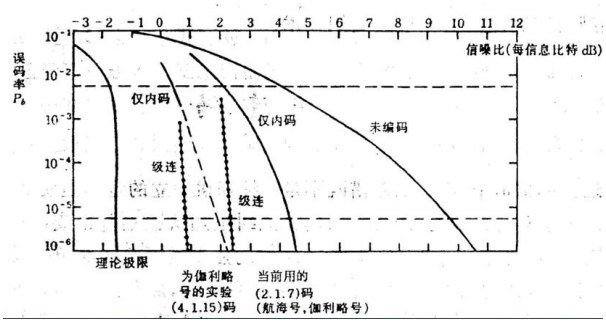

图 13 理论误比特率曲线

3.2 实验结果分析

在实验一中已经计算了无信道编码时仅依靠 QPSK 调制，不同 SNR 时的误比特率，在本实验中，对每一帧输入比特（500 位），不经过卷积码编码，直接通过 QPSK 调制解调并计算所有帧的平均比特率，与 2.5 的实验结果做对比，并大量增加采样率（以 0.1 区分 SNR），最终的实验结果如图 12 所示：

由实验结果可知，通过（2，1，2）卷积码的信道编码，虽然码元较未编码的数据多了一倍，但相同 SNR 情况下，误比特率有非常明显的下降。更重要的是收敛到 0 的速率更快，意味着通过信道编码后的卷积码可以在更严峻的信道条件中完成信息传输，并且对硬件要求更低。

此外一个有趣的现象是当 **SNR 较低的时候，卷积编码反而会导致更高的误比特率**，如图 12 中大约在-1.5db 处，而理论图中的（2，1，7）卷积码也有类似的趋势。此时信号功率已经小于噪声功率（0.7079），因此我推测由于噪声功率已经大于信号功率，错码比例已经超过正确码比率。仅由 QPSK 解调时，由于 QPSK 不具备纠错能力，因此可能出现负负得正的情况，而卷积码纠错能力较强，因此译码出的反而是“负”的错码，故在 SNR 极低的极端信道中，卷积码效果反而不如无编码。如果采用软判决的设计，即不经过 QPSK 解调，采用欧式距离判断，从理论上可以改善该现象。

前人仿真时（10 年左右）曾遇到的相似情况（卷积编码并没有提升误比特率），但只提出了增大 SNR 的解决办法，而未对现象本身分析。经吴教授指点，常见的通信系统理论上仅需考虑 BER 小于 10^-1 时的场景，实际应用是要求小于 10^-2 次方场景，因此考虑极低 SNR 时两者的性能差异没有实际价值(移动通信系统已经无法正常工作)。卷积码效率低的原因，吴教授认同我推测的合理性，并认为更多的冗余可以解决该问题，教授指出，相较增加编码冗余（极难证明有实际增益），采用分集技术可以更好的解决噪声问题。

此外，也有前人发现（2，1，2）卷积码较（2，1，8）卷积码在低 SNR 时有更好的误比特率，我认为与上述现象原理相似。以深度学习训练类比，采用复杂的编码方式可能反而导致过拟合的现象，因此在噪声较大时表现反而差。是否可以将深度学习中常见的处理噪声的算法应用于改善编码算法，例如 meanteacher，cross-pseudo 等算法迁徙到移动通信编码中，针对低信噪比且难以应用分集技术的极端信道（例如宇宙、深海环境），也许是一个很好的研究方向。本文暂且不做更多讨论。

四、拓展内容：Viterbi 译码算法的改进

4.1 截短译码

4.1.1 截短译码原理和编程实现

截短译码的设计是为降低译码时延，当译码器中存储的路径长度达到某个指定的译码深度 L0 时，就对幸存路径最前面的一级支路做出判决，输出对应的信息码元，然后再计算下一级的幸存路径，以此类推。从代码实现的角度而言，仅需在 2.4 设计的 Viterbi 译码函数中增加路径长度到达 L0 时刻的判决输出和剪枝两个功能即可。该部分的核心代码逻辑如图 14 所示：

```
%累计深度达到L的整数倍，则输出数组并且把其余路径删除（同化为一条路径）
 if(mod(i,L)==0)
     bestval=min([val_a,val_b,val_c,val_d]);%通过四条路径的累计深度判决最佳路径
     if val_a==bestval%第一路径是最佳路径
         m = ma;%输出数组，由第一路径确定
         mb = ma;%剪枝操作，将第二、第三、第四路径均取与第一路径相同
         mc = ma;
         md = ma;
     end
```

图 14 截短译码的判决、输出、剪枝逻辑

从复杂度而言，上述逻辑实际上是“伪剪枝”，复制路径虽然实现了剪枝效果，但浪费了很多空间资源。修改图 9 代码中的路径继承逻辑即可实现真正的“剪枝”（从内存中删除）。由于截短译码主要用于降低通信系统的时延，并且本次实验要求中也未对空间复杂度有要求，因此此处仅提供思路，暂不做空间复杂度的优化，此外后续笔者提出的进一步优化算法需要设计动态 L，因此需要提供足够的内存。

4.1.2 截短译码误比特率仿真实验结果

首先，采用不同 L0 深度，不同 SNR 时，在相同卷积码、QPSK 调制解调、输入信息比特和帧数的条件下，不同 L0 情况的截短译码仿真结果如图 15 所示：

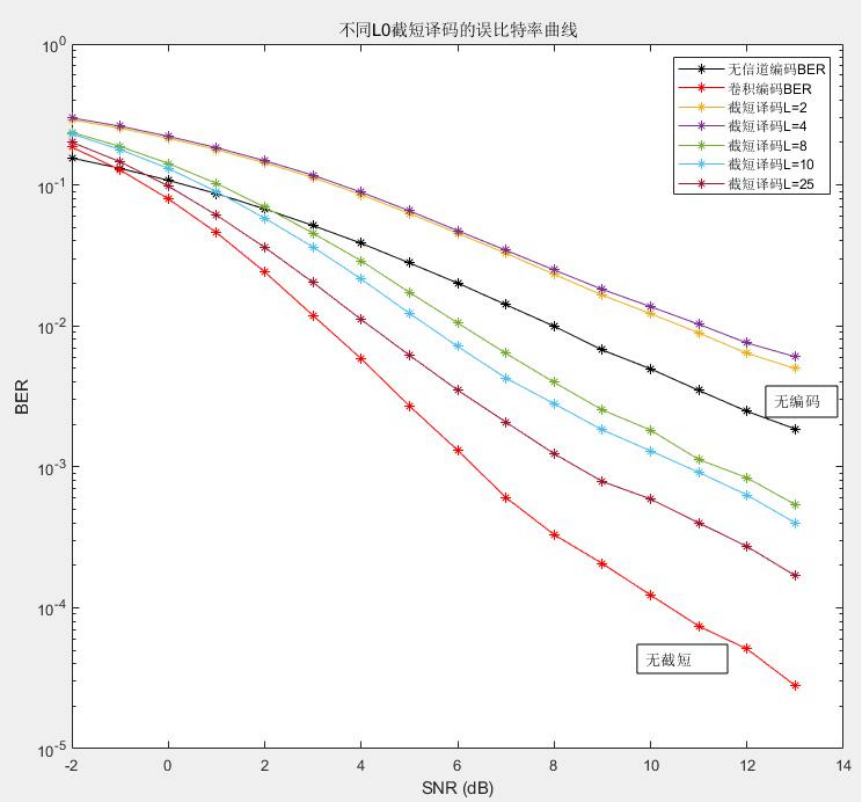

图 15　不同截短深度 L0 时的截短译码误比特率曲线（帧数：2000 卷积码长：1000）

由实验结果可知，随截短深度增加，截短译码的误比特率不断逼近未截短时的误比特率曲线，当截短深度极少（2，4）时，截短译码表现甚至不如无编码曲线，此时编码反而引入更多误差。此外，不同的截短深度下误比特率均随 SNR（db）增加呈现指数级下降的趋势（取对数坐标后为线性下降）。

就复杂度而言，截短译码在译码时理论上可以占用更少的内存，空间复杂度较低，仅需 4 个等于截短深度 L 的数组记录路径（原算法为码长 s）。此外，截短译码最大的优势在于减少了输出时延，无截短的译码需要 S 位输入才能输出，当码长较大时输出时延很高，对累计距离的记录也较多，很可能出现路径的累计距离溢出现象，而截短译码每 L 位即可输出一位，输出时延大大降低。此外，实际应用时，以应用层协议为例，截短译码更容易快速发现错码，从而请求重传，而无截短译码虽然误比特率较低，但由于时延的存在，请求重传的时间更慢。可能导致拥塞或丢包。就时间复杂度而言，采用截短译码并没有减少运算次数，复杂度仍等于无截短 viterbi 译码算法 O（$N*D^2$），远高于不采用剪枝的遍历最佳路径求解 O（D^N）。

4.2 自适应 Viterbi 截短译码算法

4.2.1 截短译码的不足分析

通过 4.1 的仿真可知，截短译码可以通过条件截短深度 L0 控制误比特率和输出时延之间的 trade-off，但是在较低 L0 时性能很差。我推测低 L0 时性能差的主要原因是因为在输出的同时只保留了当前路径，裁剪了另外 3 个分支（包含正确的分支），从而造成误比特率较高，并且也可能出现两分支累计距离相等的情况（均不为 0），此时裁剪另一分支会导致正确路径被裁剪造成误码。

由上述分析可知，如果在输出的同时不裁剪所有其他分支，即将输出和裁剪较差分支的触发条件用不同的条件控制，便可以避免该误差的发生。

此外，截短译码中的超参数 L0 难以设置，在实际通信系统中，信息比特之间可能有较强的相关性（非随机比特），此时若将通信系统中的 L0 固定，便是忽略了信息比特之间的相关性，或是限制了系统中信息的类型（例如将 L0 针对 8bit 信息设计，但对于 16bit 的信息适应性就不足）。实际上已有许多 Viterbi 译码算法的自适应设计，我参考一篇文献中[1][2]的设计阈值和路径保留的思想，在原有的截短译码算法上重新设计出了自适应 Viterbi 译码算法，主要基于输出和裁剪分别进行和自适应 L0 的设计，与截短译码算法相比，该算法在保留了与其相同短时延优点的同时，大大提升了误比特率。

4.2.2 算法说明

本算法的根本设计思想是在输出的同时不剪枝，而是通过不同条件触发输出和剪枝事件，解决上述分析的不足之处。剪枝的方式与 4.1 相同，即合并路径为另一个路径。

首先记录当前的有效路径数量为 4，并每隔一定的输入比特（生命周期默认为 1）触发一次是否剪枝的判定。设计了一个阈值 Thr，当目前的最佳路径和最差路径的累计距离差大于 Thr，则将最差路径合并为最佳路径（触发剪枝），并将有效路径数量减一。

若一直无法触发 Thr，说明当前路径均已接近最佳，此时判断，若最佳路径与最差路径的累计距离完全相同，则不触发剪枝并进入下一位输入，但是将有效路径数量减一，此处减一是为了加快输出，若这两个路径完全相同，则会在接下来的循环中迅速减一从而输出，若不相同，则不会继续减一，但最差的和最佳相同也说明此时译码路径已经很稳定，极大概率使用 Thr 已经无法继续触发，因此通过该方式减少有效路径数量从而输出（此处也可以在下一轮仍相等时直接合并最差和最佳路径，仿真结果相似）。

当有效路径数量仅为 1 时，触发输出，并通过变量记录输出之间的时延（L0）和统计所有时延的平均值。算法较长此处略去展示，详见 cc.m,line478

由于本算法并没有控制截短深度 L0，而是通过设计阈值和生命周期，让算法在译码时自动评估译码结果，在最差路径和最佳路径累计距离相近的时刻输出译码，从而实现算法自动控制 L0，因此我将其成为自适应 L0 截短译码算法，该算法没有固定的 L0，仅可以统计平均 L0。

4.2.3 自适应 Viterbi 截短译码算法的仿真结果及分析

较小的阈值和生命周期可以更容易触发输出，从而导致 L0 较小。我选择了阈值为 2，生命周期为 1（每一轮均触发）的超参数组合，与实验 4.1 的数据做对比，实验结果如表 1 和图 16 所示：

表 1 不同 SNR（dB）时的自适应 L0（平均）

SNR：	-2	-1	0	1	2	3	4	5
L0	7.31	7.17	7.02	6.84	6.69	6.54	6.41	6.34
SNR：	6	7	8	9	10	11	12	13
L0	6.28	6.24	6.20	6.19	6.19	6.18	6.16	6.16

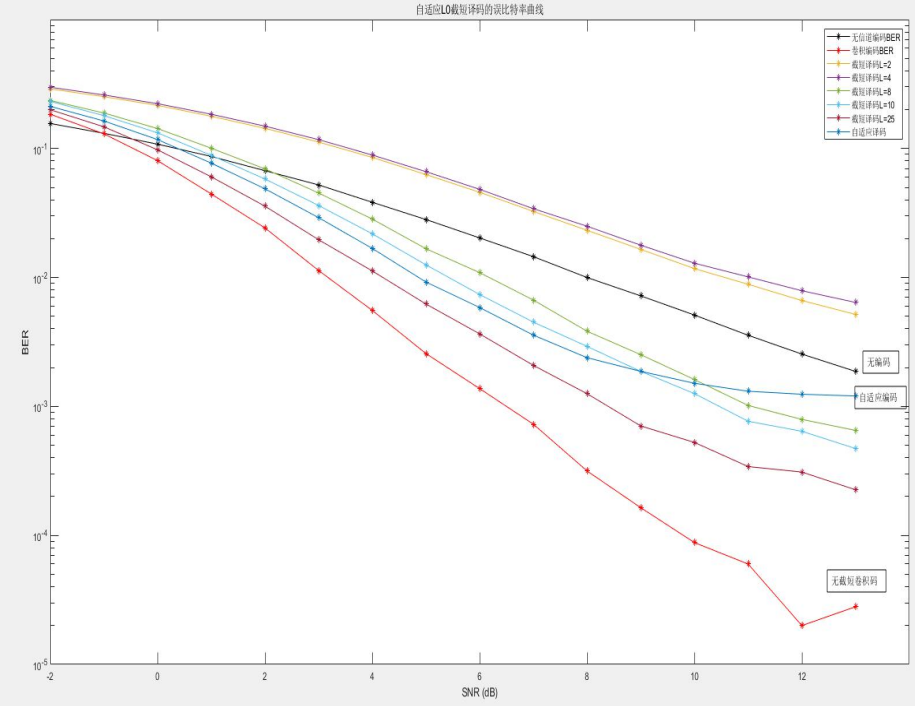

图 16　自适应 L0 截短译码算法的误比特率曲线（蓝色曲线，帧数：1000 卷积码长：1000）

由实验结果可知，自适应 L0 算法在平均 L0 =6 ~ 7 的时延条件下，在小于 9dB SNR 条件下误比特率优于 L0=10 的截短译码，在时延和误码率上都比原先的截短译码算法有优势。调节不同的 thr 和超参数，也可以改善该算法在高信噪比时的性能，在仿真时也可以取得接近甚至在低 SNR 时超过无截短卷积码的性能，但 L0 较大。

4.2.4 自适应 Viterbi 截短译码算法的优点及不足

从算法的角度而言，该算法针对截短算法“一刀斩”的剪枝思想做了改进，由算法自动根据当前的译码最佳和最差累计距离之差计算是否剪枝，是否强合并，并在最佳和最差相差不多的时刻自动输出，实现自适应 L0。在低 SNR 情况下，主要由剪枝触发输出，在高 SNR 条件下，主要由强合并条件触发输出，通过设计合理的超参数，可以很容易调整该算法在低 SNR 和高 SNR 时的平均 L0 和误码率性能。

该算法证明了截短算法确实存在“一刀斩”的问题，并且将输出和剪枝分开，可以使译码算法在相同的平均 L0 取得更好的平均误比特率。也证明了截短算法在低 L0 的时候仍存在通过改进算法提高误比特率的潜力。

该算法仍然存在许多不足，首先仍有部分超参数需要设计，如阈值 Thr，可以采用参考文献[2]中的算法或其他自适应/渐进算法改进，仿真时不同的超参数对算法性能有很大的影响。笔者能力和精力有限，对于该算法超参数设计和性能提升难以给出数学证明，目前仅提供定性的分析。

其次是在实际应用的场景中，不确定时延也可能引入更多误差。固定输出，自适应剪枝，也许也能起到相似的提升效果。

致谢：感谢吴泳彭教授在我实验三遇到困惑时的耐心指点，让我对问题和解决思路有了更深刻的理解。

参考文献：

[1] 李宗伯,张普珩,张波涛,胡文敏,刘衡竹.一种 Viterbi 译码算法的改进[J].北京交通大学学报,2008,32(06):69-72+77.

[2] 梁振,蒋宇中,张涛涛,衣军波.一种自适应维特比译码算法研究[J].通信技术,2013,46(09):35-37.