

بسم الله الرحمن الرحيم

نام و نام خانوادگی: علی کریمی

مقطع تحصیلی: کارشناسی ارشد

تمرین هفته : ارور ها DMT

نقش و کاربرد Error-Correction Code در سیستم‌های DMT

سیستم DMT مثل ADSL، VDSL، و G.fast و بسیاری از استانداردهای مبتنی بر OFDM سیگنال را به صدها یا هزاران زیرحامل تقسیم می‌کند. هر زیرحامل نرخ، کیفیت و نسبت سیگنال به نویز متفاوتی دارد. به همین دلیل وجود کدهای تصحیح خطا کاملاً حیاتی است.

اهداف استفاده از ECC در DMT

1. کاهش BER در کانال‌هایی با نویز زیاد
2. افزایش مقاومت در برابر نویز ضربه‌ای (impulse noise)
3. افزایش نرخ انتقال با اجازه دادن به بیت‌لودینگ تطبیقی
4. بهبود پایداری لینک به خصوص در خطوط تلفن با تداخل زیاد

کدهای تصحیح خطای رایج در DMT

1. Reed-Solomon (RS) Coding

ویژگی‌ها:

- بلوکی، خطی و بسیار مقاوم در برابر burst-error
- از پرکاربردترین کدها در سیستم‌های DSL
- معمولاً به صورت (n, k) مثلاً RS(255, 239)

دلیل استفاده:

در DMT ممکن است نویز ضربه‌ای چندین نمونه را خراب کند؛ RS بهترین انتخاب برای جبران خرابی‌های پشت سرهم است.

2. Convolutional Coding

در بسیاری از استانداردهای DMT ، RS همراه با یک کد کانولوشنال استفاده می شود.

ساختار رایج:

- $R = 1/2$ یا $4/3$
- معماری Viterbi Decoder برای دیکد کردن

3. فوروارد اینترلیونگ (Interleaving)

Interleaver به تنهایی یک ECC نیست، اما نقش مهمی دارد.

چرا لازم است؟

- Burst-error ها را «پخش» می کند تا کد RS بتواند آن ها را رفع کند.
- بدون interleaver ، حتی کدهای قوی RS هم نمی توانند جلوی نویز ضربه ای را بگیرند.

نکته:

- تأخیر سیستم را افزایش می دهد (مهم در ADSL و VDSL).

4. Trellis Coding (TCM)

در برخی سیستم های DMT جهت افزایش coding gain استفاده می شود.

مزایا:

- افزایش حدود 3 dB در SNR بدون افزایش نرخ ارسال
- سازگار با مدولاسیون QAM در زیرحامل ها

5. LDPC Codes (G.fast در نسل های جدیدتر مثل)

خصوصیات:

- کارایی نزدیک به حد شانون

- مناسب برای کانال‌های با نویز زیاد
- پیچیدگی محاسباتی بالاتر نسبت به RS/Convolutional

ترکیب ECC با بیت‌لودینگ (Bit Loading)

در DMT هر زیرحامل سطح SNR مخصوص خودش را دارد، بنابراین ECC همراه با تخصیص تطبیقی بیت‌ها عمل می‌کند:

- زیرحامل‌های قوی → بیت بیشتر QAM چند سطحی
- زیرحامل‌های ضعیف → بیت کمتر
- ECC یک تخمین دقیق از میانگین BER کل سیستم را ثابت و کوچک نگه می‌دارد

این همکاری بین ECC و Bit Loading اجازه می‌دهد تا نرخ انتقال نهایی بسیار بالاتر از یک سیستم بدون کد تصحیح خطا باشد.

نقش ECC در برابر انواع نویز در DMT

نوع نویز	مکانیزم مقابله در DMT
نویز سفید گوسی (AWGN)	کد کانولوشنال TCM +
نویز ضربه‌ای (Impulse)	RS + Interleaving
کراس‌تاک (NEXT/FEXT)	محصول مشترک ECC و bit-loading
تداخل فرکانسی	TCM/LDPC در نسل‌های جدید

★ جمع‌بندی

در سیستم‌های DMT استفاده از ECC یک بخش اساسی و غیرقابل‌جداست و معمولاً به صورت ترکیبی استفاده می‌شود:

- ✓ RS برای خطاهای پشت‌سرهم
- ✓ Convolutional Coding برای خطای تصادفی
- ✓ Interleaving برای مقاوم‌سازی در برابر burst
- ✓ TCM برای افزایش بهره کدینگ
- ✓ LDPC در نسل‌های جدیدتر مانند G.fast

شبیه سازی:

کد شبیه سازی کامل را اجرا کردم و خروجی ها (نمودار + جدول نتایج) داخل نوت بوک تولید شدند. در ادامه خلاصه کار، تفسیر سریع از خروجی ها، کد اجرا شده (همان چه اجرا شد) و پیشنهادهای عملی برای بهبود شبیه سازی و گرفتن نتایج معنادارتر را به زبان فارسی می بینی.

خلاصه شبیه سازی انجام شده

- مدل: یک پیاده سازی ساده $DMT \approx OFDM$ با $N=64$ زیرحامل و $CP=16$
- مدولاسیون: **QPSK**
- دو حالت مقایسه شدند:
- 1. **Uncoded OFDM (QPSK)**
- 2. **Convolutional coded (rate 1/2, polynomials 171/133 octal) + block interleaver + OFDM**
- کانال AWGN: به علاوه نویز ضربه ای (**impulse noise**) با احتمال نمونه ای p_{impulse} که شدت بالایی می تواند اضافه کند) برای مدل سازی. (**burst/impulse noise**)
- کد و اجرای Viterbi ورودی سخت (**hard-decision**) — برای دیکود کردن کد کانولوشنال پیاده سازی شد.
- نقاط آزمایش: $E_b/N_0 = [0, 2, 4, 6, 8, 10] \text{ dB}$
- تعداد OFDM symbol ها در هر آزمایش: 100 (برای اجرای سریع؛ قابل افزایش).

نتایج (خلاصه چاپ شده در خروجی)

برای هر E_b/N_0 چاپ شد BER: حالت Uncoded و BER حالت Coded. جدول و نمودار نیز تولید شد. در این اجرای اولیه مقدار BER ها تقریباً ~ 0.5 (یعنی نیم) گزارش شد — که معنی اش این است خطای خیلی زیاد است (نسبتاً نزدیک به رفتار «تصادفی»).

نمونه چاپ شده (خلاصه):

$E_b/N_0=0 \text{ dB} \rightarrow$ Uncoded BER=5.0141e-01, Coded BER=4.9688e-01
 $E_b/N_0=2 \text{ dB} \rightarrow$ Uncoded BER=4.9797e-01, Coded BER=4.9992e-01
...

نمودار BER vs E_b/N_0 هم در خروجی نشان داده شد

چرا BER خیلی بد (≈ 0.5) درآمد؟

این رفتار می‌تواند دلایل متعددی داشته باشد؛ مهم‌ترین‌ها:

1. نویز ضربه‌ای (impulse noise) بسیار قوی انتخاب شده (amp) خیلی بزرگ و/یا احتمال p_{impulse} نسبتاً بالا. (اگر نویز ضربه‌ای بخش بزرگی از نمونه‌ها را بشکند، حتی کدها) خصوصاً اگر interleaver خیلی کوتاه باشد یا دیکودر سخت-تصمیم استفاده شود (قادر به تصحیح همه خطاها نیستند).
2. دیکدر Viterbi که پیاده‌سازی کردم «hard-decision» است — یعنی ما بیت‌های سخت QPSK را به Viterbi می‌دهیم. برای بهره‌گیری کامل از مزایای کدنویسی، مخصوصاً در حضور AWGN و سیگنال‌های QPSK، معمولاً از soft-decision Viterbi (LLR-based) استفاده می‌شود که میتواند ~ 2 dB برتری بدهد.
3. ممکن است طول شبیه‌سازی (نمونه‌ها) کم باشد و نویز ضربه‌ای تصادفی باعث شده توزیع خطاها نرمال نشود.
4. طراحی این interleaver ساده‌ی بلوکی ممکن است با الگوی نویز ضربه‌ای که روی نمونه‌های زمان دامنه می‌نشیند (به درستی هم‌تراز نشده باشد) یعنی burstها بعد از FFT روی مجموعه‌ای از ساب‌کاریرها پراکنده می‌شوند و interleaver فعلی کارایی لازم را ندارد.
5. احتمال خطا در پیاده‌سازی Viterbi/ترلیس (پیاده‌سازی ساده شده) — همیشه ممکن است باگ منطقی وجود داشته باشد.

کد اجرا شده

همان کدی که اجرا شد — شامل encoder، interleaver، viterbi (hard)، OFDM، اضافه کردن AWGN/impulse و رسم نتایج

پیشنهادهای

1. نرم‌تر کردن نویز ضربه‌ای: کاهش impulse_amp و/یا کاستن از p_{impulse} تا تاثیر AWGN قابل مشاهده شود. (در حالت فعلی نویز ضربه‌ای غالب است).
2. استفاده از soft-decision Viterbi: به جای دادن بیت‌های سخت به ویتربی، از LLRهای نمادها (soft metrics) استفاده کنیم — تداخل کمتری در عملکرد کد حاصل می‌شود و gain مشخص می‌شود. (می‌توانم این را پیاده کنم).
3. افزایش تعداد نمادها / تستهای Monte-Carlo: افزایش ofdm_symbols یا اجرای تکرار متعدد تا آمار پایدارتر شود.
4. افزایش طول interleaver یا interleaving بر پایه OFDM-symbol نماد به نماد (burstهای زمانی بهتر پراکنده شوند).
5. استفاده از کد بلوکی قوی برای burst مثل Reed-Solomon همراه با RS — interleaver مخصوص burst-error است و در استانداردهای DSL رایج است. می‌توانم یک پیاده‌سازی ساده RS فرآیند interleaving اضافه کنم پیاده‌سازی RS کمی سنگین‌تر است اما امکان‌پذیر.
6. بررسی صحت پیاده‌سازی Viterbi: تست با سناریوی بدون نویز (یا نویز سفید کم) تا مطمئن شویم که encoder/decoder درست عمل می‌کنند.

گزینه‌های پیشنهادی

به صورت پیش فرض می‌تونیم یکی از موارد زیر را اجرا کنیم و همان‌جا نتایج کامل را نشان دهم

- A کاهش شدت نویز ضربه‌ای مثلاً $p_{\text{impulse}}=0.0005$ و $\text{impulse_amp}=5$ و اجرای مجدد سریع و اغلب کافی برای دیدن اثر کد
- B اضافه کردن **soft-decision Viterbi** پیاده‌سازی و اجرا
- C جایگزینی یا افزودن **Reed-Solomon + interleaver** برای مقابله با burst پیاده‌سازی RS و اجرای مقایسه
- D افزایش تعداد **OFDM symbols** و تکرارها تا منحنی BER صاف‌تر و آماری‌تر شود مثلاً $\text{ofdm_symbols}=2000$
- E همه موارد بالا وضعیت سنگین‌تر؛ زمان اجرا بیشتر

با تشکر

پایان