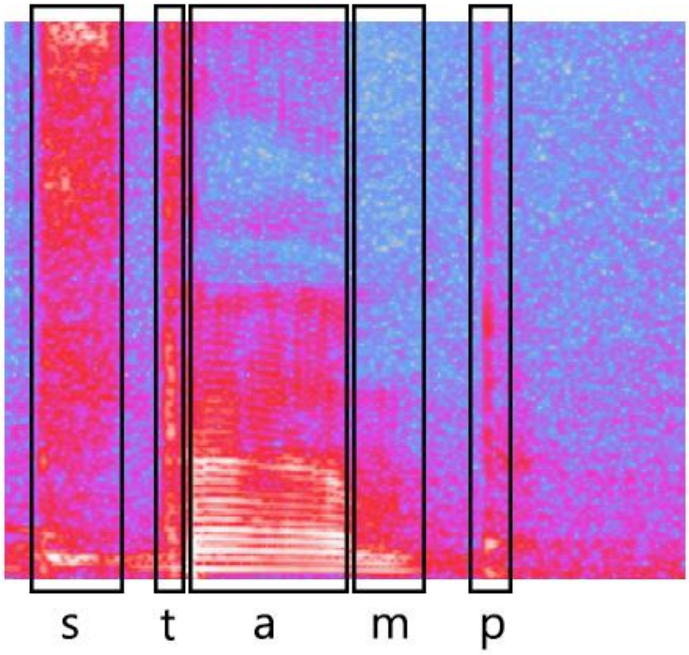
گزارش تمرین سری دوم

سیستم‌های چندرسانه‌ای

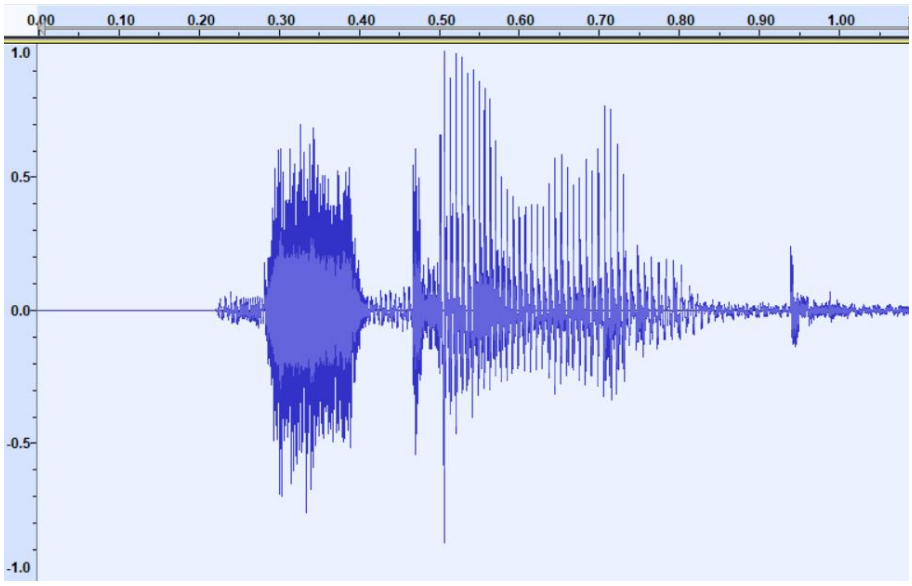
رضا اکبریان بافقی - ۹۵۱۰۰۰۶۱

۱.۱

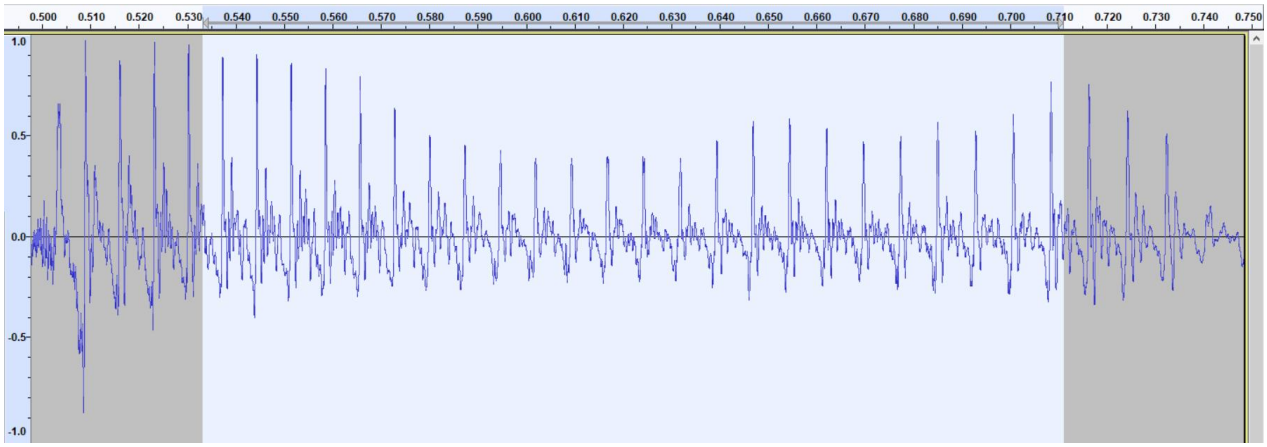
در شکل ۱ spectrogram حروف کلمه stamp مشخص شده است. در شکل ۲ waveform این کلمه نشان داده شده است. در شکل ۳ در waveform حرف a زوم شده است. همان‌طور که در شکل ۳ مشخص شده است در حرف a انگار یک سری از فرکانس‌های خاص به صورت پترن تکرار می‌شوند. چون یک حرف صدادار می‌باشد. پس چون در هر ستون spectrogram یک PSD گرفته شده است پس می‌توان مشاهده کرد که یک سری فرکانس‌های غالب در آن مشخص است. که نشان دهنده پترنی هستند که در waveform این حرف تکرار شده است.



شکل ۱ spectrogram حروف کلمه stamp



شکل ۲ waveform کلمه stamp



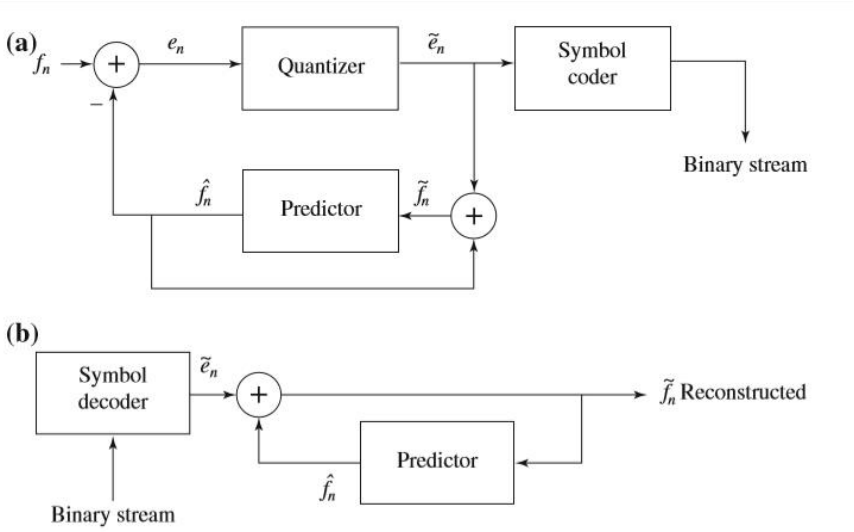
شکل ۳ waveform حرف a در کلمه stamp

۲.۱

حالت خام ذخیره صدا در حالت دیجیتال همان PCM می‌شود. کوانتیزه کردن مستقیم یک نمونه اولیه با طول ثابت در کد کردن PCM می‌باشد. در واقع ما در PCM یک rate برای عمل نمونه‌برداری انتخاب می‌کنیم و در این فواصل زمانی مشخص مقدار نمونه‌برداری شده quantized می‌شود. سپس این مقدار توسط تعدادی بیت کد می‌شود. که این عمل هم توسط Quantization Levels کنترل می‌شود. مثلا می‌توان با sampling rate ۸ کیلوهرتز و quantization level ۸ بیت در هر sample، عمل sampling و quantization را انجام داد. در این‌جا هر مقداری که نمونه‌برداری می‌شود از ۰ تا ۲۵۵ مقداردهی می‌شود که بتوانیم آن را کد کنیم.

۳.۱

در DPCM سیگنال خطای پیش‌گویی کد می‌شود. در واقع DPCM یک عمل برای compress کردن صوت می‌باشد که به صورت lossy عمل می‌کند. که براساس دامنه زمان عمل می‌کند. در این‌کار ما به‌صورت پیش‌گویانه ما تعیین می‌کنیم که مقدار در زمان خاص چقدر باشد. به عنوان مثال ساده‌ترین پیش‌گویی می‌تواند این باشد که ما تفاوت بین مقدار فعلی و قبلی را quantized کنیم، در واقع به این معنی است که ما با فرض این‌که مقدار فعلی ما برابر مقدار قبلی است، مقدار فعلی را پیش‌بینی کرده‌ایم. در نتیجه با فرض اینکه سیگنال دارای خاصیت locality می‌باشد، تعداد بیت‌هایی که برای نگه‌داری استفاده می‌شود، کم‌تر می‌شود. پیش‌گویی‌های بهتر می‌تواند میانگین گرفتن عادی یا وزن دار از داده‌های قبلی برای تعیین مقدار فعلی باشد. در شکل ۴ می‌توانید نمونه‌ای از یک DPCM را مشاهده کنید.



شکل ۴ در این DPCM a، encoder و decoder.b می‌باشد.

در این‌جا ما می‌توانیم سیگنال اصلی را f_n، سیگنال پیش‌بینی‌شده را $\hat{f}_n$ و سیگنال quantized شده و دوباره ساخته شده را $\tilde{f}_n$ بنامیم. در واقع DPCM مقدار پیش‌بینی‌شده را از مقدار واقعی کم می‌کند تا e_n به‌دست بیاید سپس آن را quantized می‌کند تا $\tilde{e}_n$ به‌دست بیاید. این فرآیند در ادامه آورده شده است:

$$\hat{f}_n = \text{function\_of } (\tilde{f}_{n-1}, \tilde{f}_{n-2}, \tilde{f}_{n-3}, \ldots)$$
$$e_n = f_n - \hat{f}_n$$
$$\tilde{e}_n = Q[e_n]$$
$$\text{transmit codeword}(\tilde{e}_n)$$
$$\text{reconstruct: } \tilde{f}_n = \hat{f}_n + \tilde{e}_n$$

۴.۱

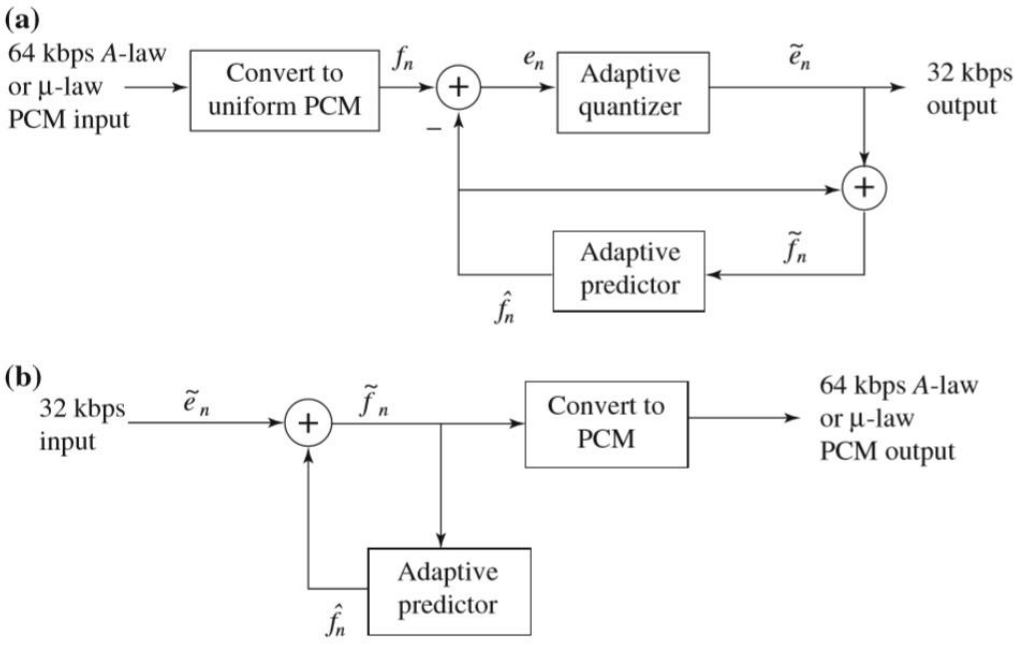
می‌دانیم که DPCM از دو بخش quantizer و predictor تشکیل شده‌است. در ADPCM می‌توانیم quantizer را به‌صورت تطبیقی تغییر دهیم بدین صورت که step size یا Δ را براساس یک سیاستی تغییر دهیم.

ما این تغییر step size را از دو راه می‌توانیم انجام دهیم: استفاده از اطلاعات سیگنال ورودی که forward-adaptation نامیده می‌شود یا استفاده از خصوصیات خروجی quantized شده. چون در این حالت اگر خطاهای quantized شده زیاد شوند ما باید quantizer غیریکنواخت Lloyd-Max را تغییر بدهیم. که این عمل هم backward-adaptation نامیده می‌شود.

ما هم‌چنین می‌توانیم predictor را هم بر اساس forward-adaptation یا backward-adaptation به‌صورت تطبیقی تغییر دهیم. در واقع ما به دنبال ضرایبی برای پیش‌بینی هستیم که تفاوت مقدار پیش‌بینی شده را از مقدار واقعی را کمینه کند. می‌توانید در شکل ۵ نمای کلی از یک ADPCM را مشاهده کنید.

به عنوان مثال می‌توان همانطور که در اسلایدها آمده ابتدا تعیین می‌شود که باید مقدار قبلی را با Δ جمع کنیم یا از آن Δ را کم کنیم. زمانی که تفاضل مقدار پیش‌بینی‌شده از مقدار واقعی یعنی e_n مثبت باشد، $\tilde{e}_n$ باید برابر Δ باشد و زمانی که e_n منفی باشد، $\tilde{e}_n$ باید برابر $-\Delta$ باشد. در واقع M در عبارت $\Delta[n] = M\Delta[n-1]$ هم بر این اساس تعریف می‌شود که زمانی که علامت e_n با

e_{n-1} یکی باشد، $M = P > 1$ و زمانی که علامت e_n با e_{n-1} برابر نباشد آنگاه $M = Q < 1$ می‌شود. از این طریق Δ در طول سیگنال تغییر می‌کند.



شکل ۵ در این ADPCM a، encoder و decoder.b می‌باشد.

۵.۱

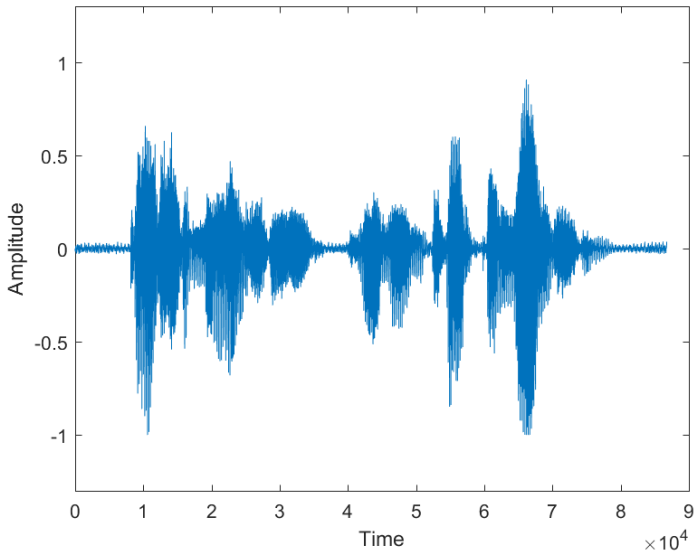
ایرادی که در LPC وجود دارد این است که اگر voiced و unvoiced در یک pitch period با هم قاطی شده باشند، عملکرد خوبی ندارد. چون فرض کرده است که در یک pitch period صوت یا voiced است یا unvoiced. در بعضی از آواها این اتفاق می‌افتد. در CELP این مسئله حل شده‌است. راه حلی که در CELP استفاده شده است encode residue نام دارد. روش کار ما استفاده از vector quantization یا codebook نام دارد. در CELP همان LPC استفاده می‌شود، اما جاهایی که LPC دارد خطا ایجاد می‌کند، آن خطا را با استفاده از یک codebook (که از تعداد زیادی نمونه که باعث ایجاد خطا شده است تشکیل شده است و از آن، تعداد محدودی مثلا ۲۵۶ خطا را جدا کرده‌ایم که آن خطاها رایج هستند) تشخیص داده می‌شود. مراحل CELP پس به این‌گونه می‌باشد که ابتدا سیگنال ورودی می‌آید را از آن LPC می‌گیریم در همان encoder، سیگنال اصلی را با نتیجه LPC مقایسه می‌کنیم، یک خطایی می‌دهد. آن خطا را در codebook پیدا می‌کنیم و index آن خطا را هم برای گیرنده می‌فرستیم. در گیرنده همان خروجی LPC را می‌سازد با این تفاوت که اگر خطایی وجود داشته باشد هم در codebook پیدا می‌کند و به نتیجه اضافه می‌کند.

۲ کار با متلب

۱,۲ فیلتر کردن سیگنال صوتی و نمونه‌برداری از صوت

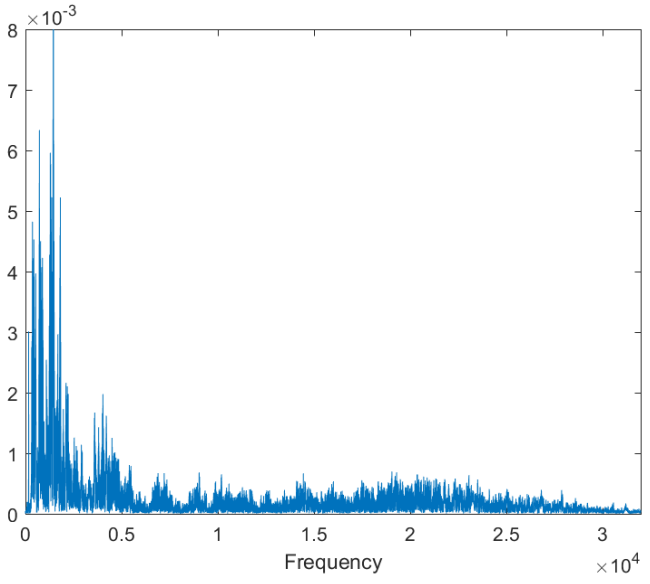
صوت‌ها و نمودارهای به‌دست آمده در فولدر Q2_1 قرار دارد.

۱,۱,۲



شکل ۶ سیگنال x32 به عنوان تابعی از زمان

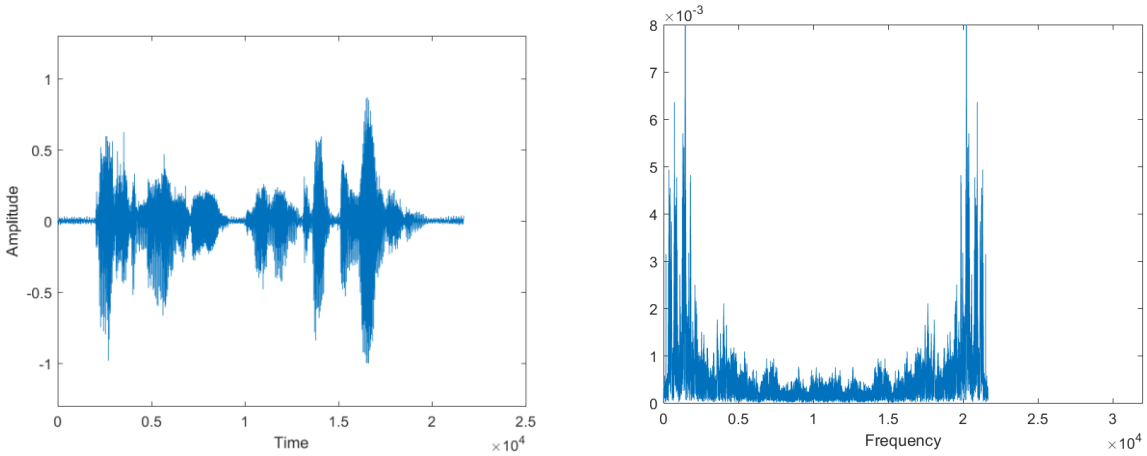
۲,۱,۲



شکل ۷ طیف فرکانسی سیگنال x32

۳,۱,۲

کیفیت صوت x8 به‌طور واضح کاهش یافته است و صدای خش خش به آن اضافه شده‌است.

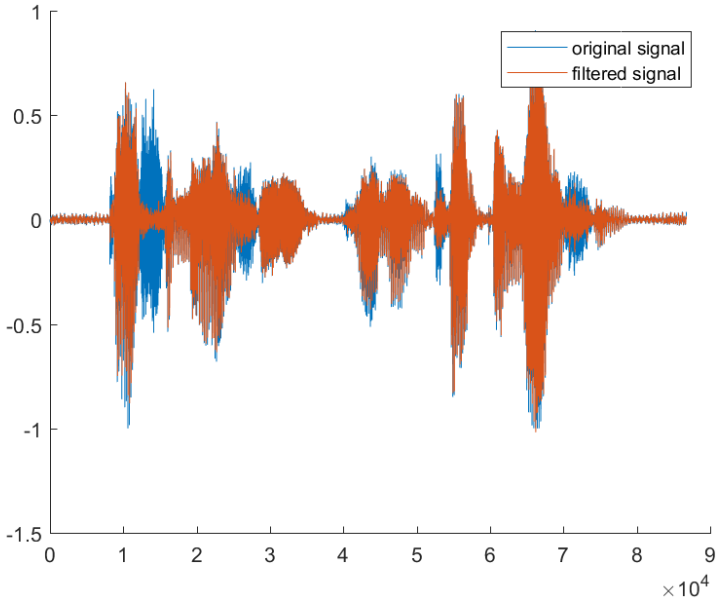


شکل ۸ در سمت راست طیف فرکانسی سیگنال x8 و در سمت چپ سیگنال x8 به عنوان تابعی از زمان آورده شده است.

۴,۱,۲

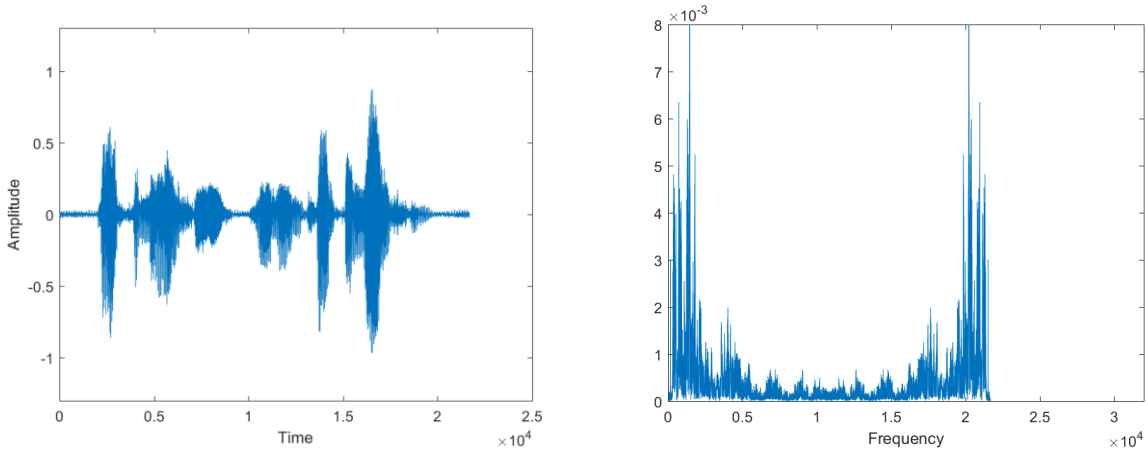
برای حذف اثر aliasing خوب است که قبل از نمونه‌برداری کاهشی، از یک فیلتر پایین‌گذر برای فیلتر کردن سیگنال استفاده کنیم. با این کار فرکانس‌های بالا حذف می‌شود و سیگنال پس از نمونه‌برداری کاهشی به صوت اصلی نزدیک‌تر است. هم‌چنین باید فرکانس cutoff فیلتر پایین‌گذر پایین‌تر از فرکانس ناکوئیست باشد، چون نرخ نمونه‌برداری باید حداقل برابر نرخ ناکوئیست باشد.

۵,۱,۲



شکل ۹ مقایسه سیگنال اصلی و فیلترشده x32

۶،۱،۲



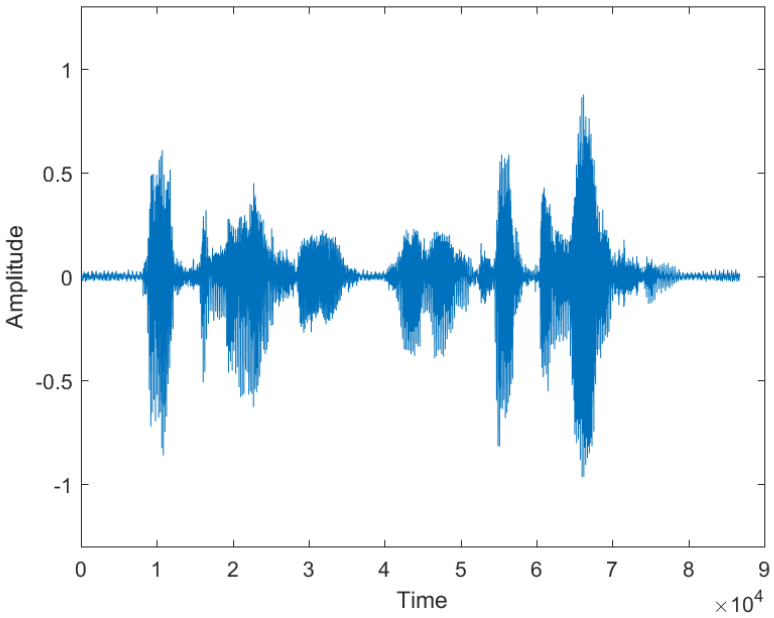
شکل ۱۰ در سمت راست طیف فرکانسی سیگنال x8f و در سمت چپ سیگنال x8f به عنوان تابعی از زمان آورده شده است.

۷،۱،۲

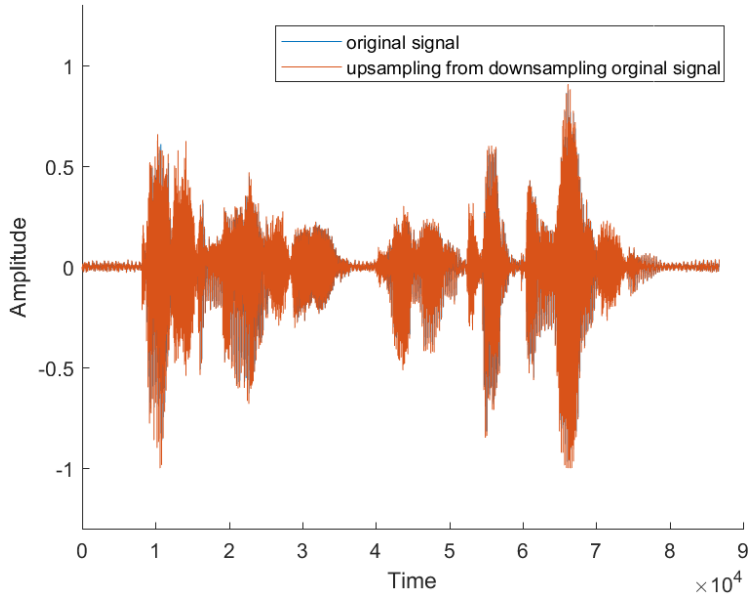
من هنگام ذخیره صوت‌های x8 وx8f، fs را برابر ۸۰۰۰ قرار دادم تا قابل شنیدن باشند. کیفیت صدا در x8f بهتر شده است. چون با گذراندن فیلتر پایین‌گذر فرکانس‌های بالا را در x32 حذف کرده بودیم، پس زمانی که نمونه‌برداری کاهشی انجام می‌دهیم کیفیت صدا بیشتر می‌شود و ما کمتر شاهد پدیده aliasing هستیم.

۸،۱،۲

در شکل ۱۲ دو سیگنال x8f32 و x32 در یک نمودار رسم شده‌اند. در نمودار به‌نظر بر هم منطبق هستند اما وقتی صوت‌شان را می‌شنویم متوجه افت کیفیت در x8f32 نسبت به x32 می‌شویم.



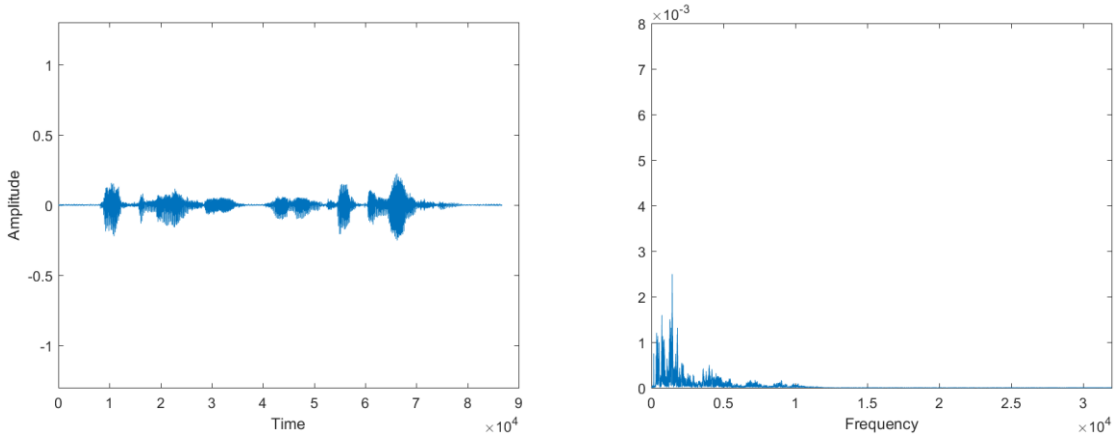
شکل ۱۱ سیگنال x8f32 به عنوان تابعی از زمان



شکل ۱۲ در این نمودار x8f32 با رنگ قرمز و x32 با رنگ آبی مشخص شده‌اند. همانطور که مشخص است تا حد خوبی بر هم منطبق هستند.

۹,۱,۲

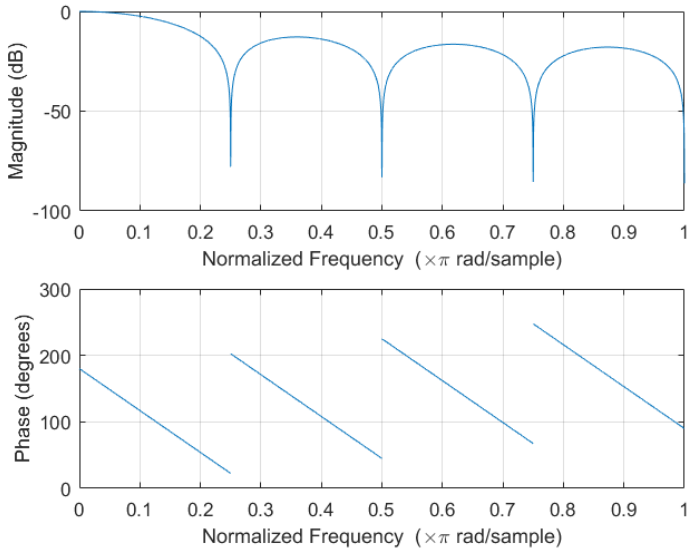
کیفیت صدا در x8f32f نسبت به x8f32 بهتر می‌شود.



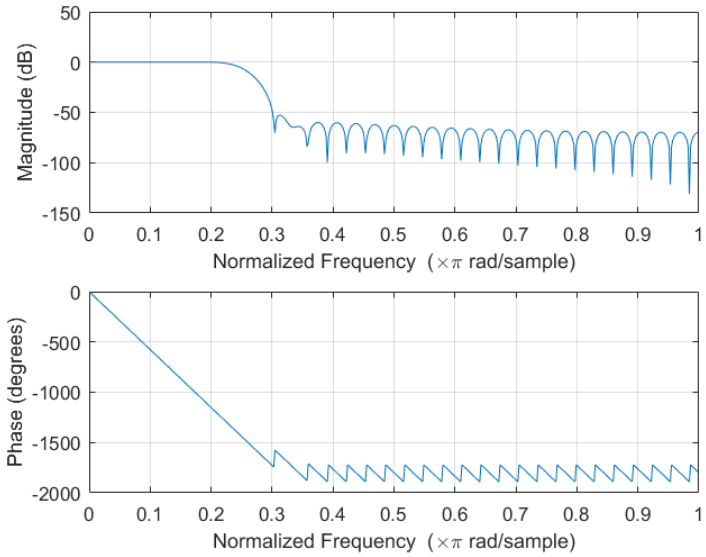
شکل ۱۳در سمت راست طیف فرکانسی سیگنال x8f32f و در سمت چپ سیگنال x8f32f به عنوان تابعی از زمان آورده شده است.

۱۰,۱,۲

می‌توانیم مشاهده کنیم که فیلتر cic نسبت به فیلتر ساخته‌شده در قسمت ۴,۱,۲ بهتر عمل می‌کند و در این حالت فرکانس‌های پایین‌تر را فیلتر می‌کند.



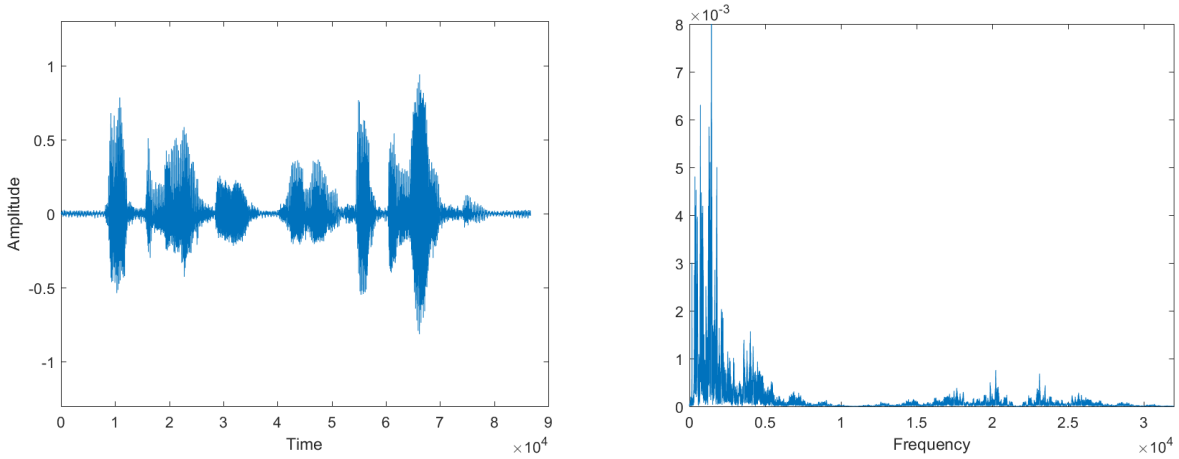
شکل ۱۴ پاسخ فرکانسی به فیلتر cic



شکل ۱۵ پاسخ فرکانسی به فیلتر ساخته‌شده در قسمت ۴,۱,۲

۱۱,۱,۲

کیفیت صدا در x8f32cic نسبت x8f32 و x8f32f بهبود یافته است. چون فیلتر cic بهتر از فیلتر قسمت ۴,۱,۲ عمل کرده است.



شکل ۱۶ در سمت راست طیف فرکانسی سیگنال x8f32cic و در سمت چپ سیگنال x8f32cic به عنوان تابعی از زمان آورده شده است.

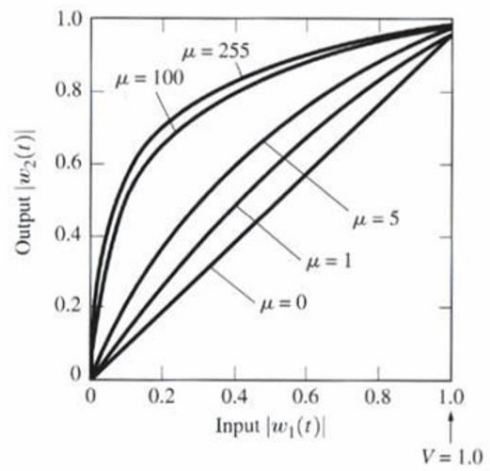
۱۲,۱,۲

طیف‌های تکراری‌ای می‌باشد که در یک نمونه تکرار می‌شود و علت آن می‌تواند این باشد که مثلا در بین هر یک از نمونه‌ها صفر قرار بدهیم. آنگاه فرکانس نمونه برداری اولیه اگر Fs باشد آنگاه این نمونه جدید در نرخ دو برابر Fs تکرار می‌شود و اگر بخواهیم با همان نمونه‌برداری قدیم نمونه‌برداری کنیم aliasing رخ می‌دهد. این کار مانند حذف فرکانس‌های بالا در قسمت‌های قبل می‌باشد.

۲,۲ کوانتایزر غیرخطی

۲,۲,۲

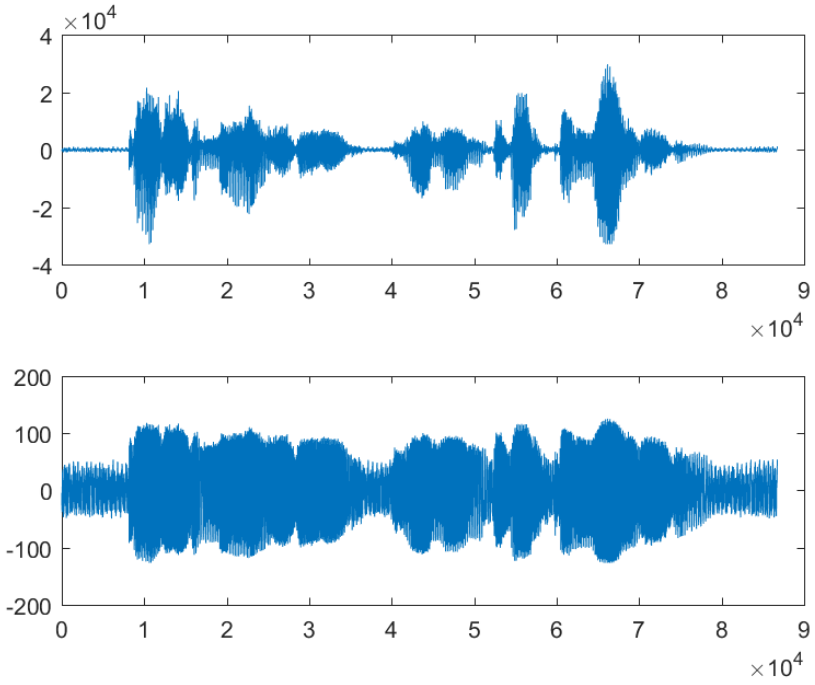
کیفیت صدا در حالت کوانتایز غیرخطی کمتر می‌شود و حاوی مقدار زیادی نویز می‌شود. به این دلیل است که در کوانتایز غیر خطی μ-law ما مقادیری که نزدیک صفر است، مقدار بالاتری پیدا می‌کند. که این مسئله در شکل ۱۷ قابل مشاهده می‌باشد. پس در جاهایی که در صوت اصلی نزدیک به صفر است در صوت کوانتایز شده مقادیری بالاتر پیدا می‌کند که صدایی شبیه نویز می‌باشد. صوت به دست آمده در پوشه Q2_2 به نام Q2_2_1.wav ذخیره شده است.



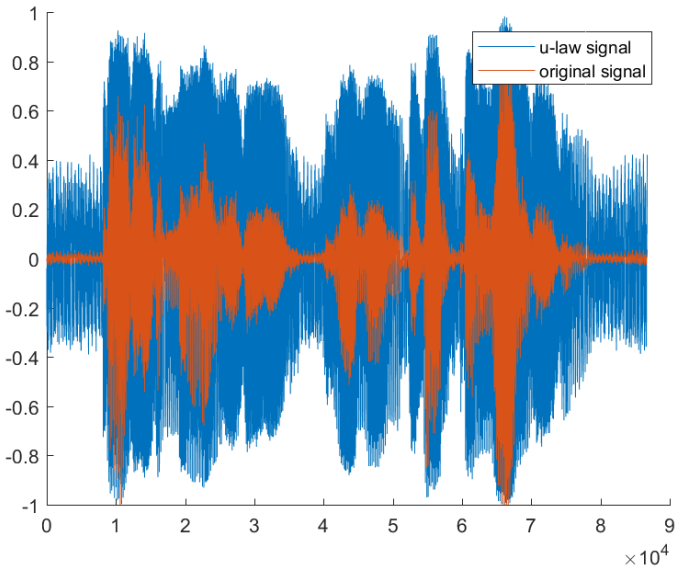
شکل ۱۷ نمودار μ-law

۳,۲,۲

در فاصله ۰ تا ۱ ثانیه همان‌طور که مشخص است، آن نوسان‌های کوچک حول صفر تبدیل به نوسان‌های بزرگتری می‌شود که صدایی شبیه نویز دارد.



شکل ۱۸ در نمودار بالا صوت اصلی و در نمودار پایین صوت کوانتایزشده غیرخطی قرار دارند.



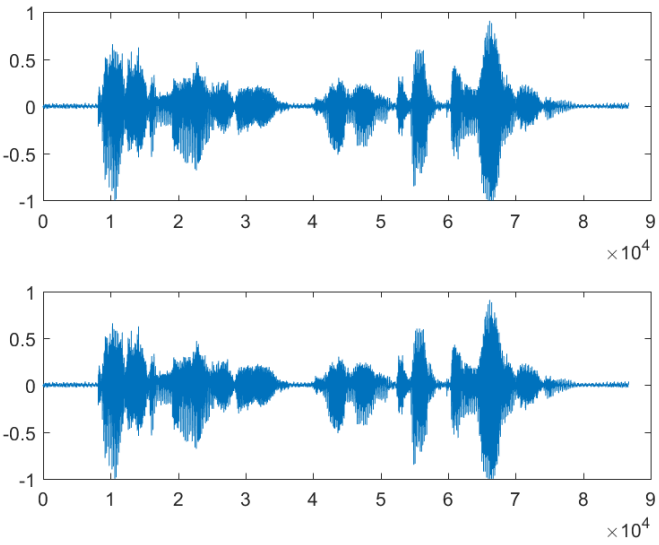
شکل ۱۹ هر دو صوت نرمال شده در یک نمودار

۴,۲,۲

همان‌طور که مشاهده می‌شود شباهت زیادی دارند. با شنیدن صوت‌های آن‌ها هم تفاوتی احساس نمی‌شود. این صوت در پوشه Q2_2 با نام Q2_2_4.wav ذخیره شده است.

۵,۲,۲

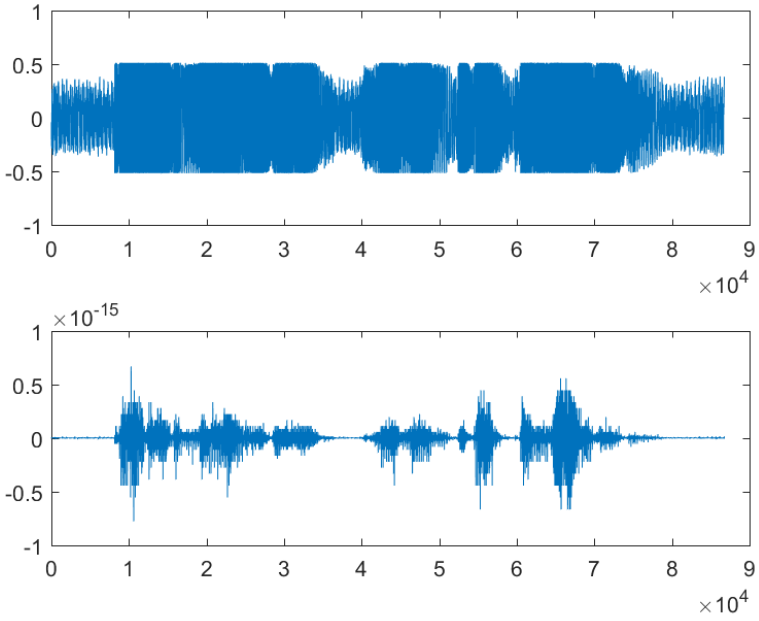
در شکل ۲۰ نمودار دو صوت آورده شده است.



شکل ۲۰ نمودار بالا صوت اصلی و در نمودار پایین صوت دی کوانتایز شده قرار دارند.

۶،۲،۲

ابتدا صوت‌ها را به بازه ۰ تا ۱ برده و سپس با استفاده از آن خطا را محاسبه می‌کنیم. همان‌طور که مشاهده می‌شود در حالت کوانتایز شده خطا مقدار بسیار بیش‌تری می‌باشد.



شکل ۲۱ در نمودار بالا خطای صوت اصلی نرمال شده و صوت کوانتایز شده نرمال شده و در نمودار پایین خطای صوت اصلی نرمال شده و صوت دی کوانتایز شده نرمال شده می‌باشد.

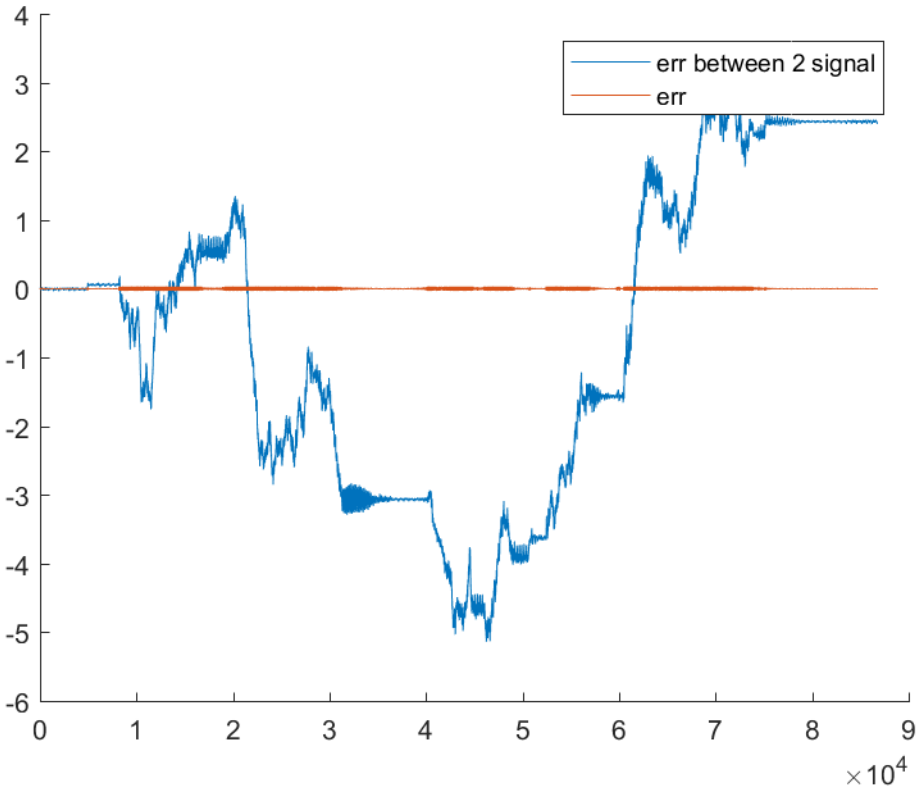
۳،۲ کدکردن پیشویانه

۱،۳،۲

در داخل فایل Q2_3 تابع mydpcm قرار داده شده است. در بازه $[-۲^{n-۱}.۲^{n-۱}-۱]$ به تعداد ۲^n پله مساوی کوانتیزیشن را انجام می‌دهیم. که با n بیت قابل نمایش باشد.

۳،۳،۲

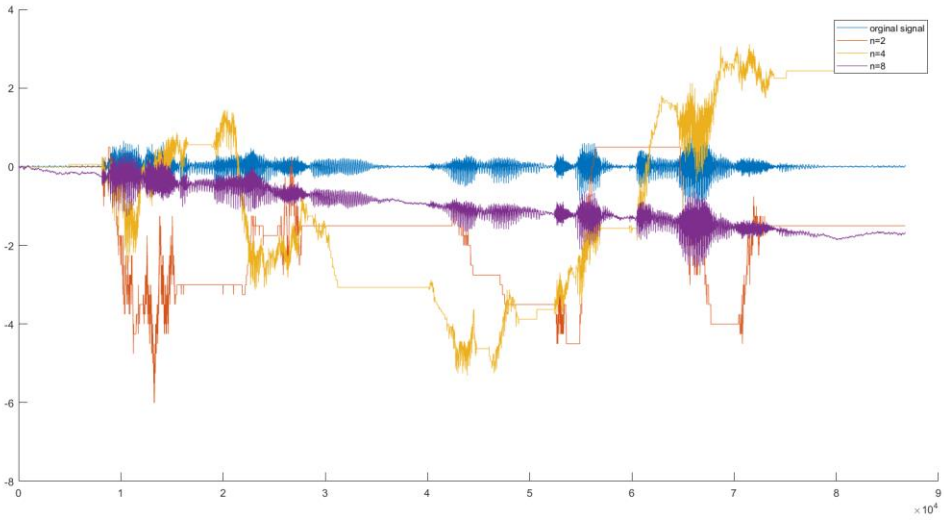
با دیدن خطای بین سیگنال اصلی و سیگنال بازسازی‌شده می‌توانیم مشاهده کنیم که با گذر زمان این خطا بیش‌تر شده‌است. به صورتی که از یک جایی به بعد دیگر قابل شنیدن نمی‌باشد و دوباره خطا کم می‌شود. چون خطا در این حالت انتشار می‌یابد.



شکل ۲۲ خط قرمز err خروجی تابع mydpcm می‌باشد و خط آبی خطای بین تفاوت سیگنال بازسازی‌شده و سیگنال اولیه را نشان می‌دهد.

۴,۳,۲

با بزرگ‌تر شدن n کیفیت صدا بهتر می‌شود و مقدار خطا کم‌تر می‌شود. در شکل ۲۲ می‌توانید سیگنال اصلی و سیگنال بازسازی شده در nهای مختلف نشان می‌دهد. همان‌طور که مشاهده می‌شود با افزایش n سیگنال به سیگنال اصلی که به رنگ آبی می‌باشد نزدیک‌تر می‌شود.



شکل ۲۳ سیگنال ها با nهای خروجی متفاوت