המרכז האקדמי רופין
Ruppin Academic Center

ספר הפרוייקט - פרוייקט גמר תשפ"ד עיבוד דיגיטלי של אותות אודיו

מגיש :רותם צלישר 203773601

מנחה :ד"ר בני גור סולומון

Contents

הקדמה

מטרת ספר הפרוייקט תהיה להציג את שלבי הפרוייקט שבוצעו עד כה, להסביר בצורה מפורטת את החלקים, השלבים ופיתוחיהם. בספר זה, אציג את חלקיו השונים של הפרוייקט, אתייחס למה בוצע, לפיתוחים, לתיאוריה ודוגמאות למימושים.

בתצוגת החומר איעזר, במידת הצורך, בגרפים, הוכחות מתמטיות, נוסחאות רלוונטיות ודיאגרמות בלוקים רלוונטיות.

להמחשת החומר (במידת בצורך בלבד) יצורפו להגשה קבצי מטלב המיישמים את המובא בטקסט.

רקע וחלקי הפרוייקט

רקע:

בעולם המודרני שלנו, בו כל כך הרבה מכשירים משדרים מולטימדיה רוב שעות היממה, עולם עיבוד אותות האודיו תופס תאוצה. הצורך בשידור, שמיעה והעברת אודיו יוצר צורך בשיטות עיבוד טובות ויעילות, על מנת שנוכל להשתמש בשימושים השונים של המכשירים שלנו.

אותות אודיו, וביניהם נכללים גם אותות הדיבור, ניתנים לניתוח ועיבוד גם במישור הזמן וגם מתוך הסתכלות על התכולה הספקטרלית שלהם. האותות השונים, מורכבים מתכולת תדרים שניתן לנתח, לאפיין וגם לייצר מערכות שיגבירו (או ינחיתו) נוכחות של תדרים מסויימים. לכל כלי נגינה, כולל מיתרי הקול של האדם, יש תכולה ספקטרלית אופיינית לאקוסטיקה של המערכת המייצרת את הצליל. עובדה זו תעזור לנתח ולעבד את אותות האודיו.

יחד עם הצורך בשיטות העיבוד, ועם התפתחות הAI בעולם, נולדו גם שימושים חדשים המקשרים את תחום למידת המכונה לעולם עיבוד האודיו. למשל - סירי ואלקסה, זאת של אפל וזאת של אמאזון, דורשות רמה גבוהה של speech recognition. בפרוייקט זה אנסה למדל יחידה שתדע לבצע Musical Genre Classification. אשתמש בdata set גדול על מנת לנסות ללמד את המודל להצליח להבדיל בין סגנונות מוסיקה (רוק, היפ הופ, ג'ז וכו'), תוך התחשבות בהיררכיית סגנונות נתונה, כאשר המטרה תהיה להגיע לדיוק של לפחות 85 אחוז בסיווג בין שני סוגי מוסיקה, ולפחות 60 אחוז דיוק בסיווג בין כל סגנונות המוסיקה. אנסה לנתח את תוכן האודיו ברמת הפיצ'רים, לרבות התוכן הריתמי, גבהי צליל ופיצ'רים נוספים על מנת להגיע ליכולת לסווג את קטע האודיו בדיוק מוצלח מספיק.

חלקי הפרוייקט:

הפרוייקט יחולק לשלושה חלקים.

- <u>משוונים</u>: בפרק זה, אצלול לתוך השיטות השונות לאיזון ספקטרלי של אותות אודיו. כחלק מפרק זה, איישם מימוש של משוון 7-Band (בעל חלוקה ל7 יחידות עיבוד מקביליות). אתחיל מלהסביר ולפתח את הרכיבים ממנו בנוי המשוון – המסננים השונים.

משוונים שונים ניתנים למימוש ע"י אסופה של מסננים שונים. משפחות המסננים מתחלקות למסנני peak, shelving, notch, וניתן לבצע סינונים ע"י אלגוריתמים מתקדמים יותר. אם נרצה, לצורך המחשה, להגביר תכולת תדר סביב תדר fc כלשהו , נוכל לממש מערכת המורכבת מ all pass במקביל עם BP פילטר, והמערכת המשותפת תיקרא peak filter. המערכת תעביר את כל הסיגנל במלואו, ובנוסף תגביר (או תנחית) את תחום התדר הרצוי. בעזרת פילטר זה, ובשיטות של חישוב מקדמי המסננים כתלות בפרמטרים של המערכת (תדר קטעון, רוחב סרט לתחום ההגבר \ הנחת וכו') נוכל לממש את המשוון המלא (למשל: פונקציה שתקבל את ההגבר הרצוי בDB, תדר FC סביבו מגבירים ורוחב סרט BW, ותדע לייצר את מקדמי המסנן PEAK\SHELVING הרצוי).

לאחר פיתוח סוגי המסננים השונים, אדגים שילוב שלהם לכדי יחידת EQ אחת גדולה.

- <u>סימולציית חדר:</u> בפרק זה, אדגים שיטה לעיבוד אודיו, המאפשרת לנו לערוך קטע אודיו "יבש" (מוקלט בתנאים אופטימליים) לכדי אודיו מעובד, שמטרתו תהיה להישמע כאילו הקטע הוקלט (או נוגן) בחדר בעל הד.

בשיטה זו, ניתן לקחת את האות היבש ולהעביר אותו בקונבולוציה יחד עם ה"תגובה לתדר" של החדר שאותו אנו מנסים לתאר. כל חדר מתאפיין ביכולת מסויימת להחזיר (או לבלוע) את התדרים המתפשטים בו, כתלות בחומר ממנו עשויים כל ה"מכשולים" בחדר שגורמים לגלים חוזרים. ע"פ זאת, אם נצליח לערוך באופן דיגיטלי את האות, כך שייתקבל ייצוג לא רק של האות המוקלט, אלא של האות המוקלט הכולל בתוכו את החזרי הגלים הפוטנציאלים מהחדר, נוכל לייצר את האשליה שהאות הוקלט (או הושמע) בחדר הנ"ל. התיאור המתמטי של החזרי הגלים המתקבלים מהחדר ייקרא "התגובה לתדר" של החדר.

בפרק זה אדגים שני אלגוריתמים, בשם SCHRODER ו-MOORER, המיישמים את הלוגיקה הזו לעיבוד הסאונד.

- <u>למידת מכונה:</u> בפרק זה, אנסה להעמיק בשיטות השונות לערב למידת מכונה עבור עיבוד אודיו. למידת מכונה תבוא לידי ביטוי בניתוחי ספקטוגרמות, אלגוריתמים שונים לזיהוי פיצ'רים ונושאים מתקדמים נוספים. בפרוייקט זה, אתמקד בשיטות לסיווג סגנון מוסיקלי.

לצורך פרק זה, ישנו data set גדול, המורכב מקטעי אודיו המסווגים לסגנונות השונים, ביניהם, קלאסי, רוק, ג'ז וכולי, כמו גם תמונות של ספקטוגרמות התואמות את הסגנונות השונים, מהן יהיה ניתן לנתח את התמונה האופיינית המתקבלת עבור סגנון ספציפי. ישנם גם קבצי EXCEL המכילים תיוגים על מידע אופייני לסוגי המוזיקה השונים (טמפו, תכולות תדר, RMS וכו'). אנסה לייצר מערכת שתגיע לכל הפחות ל60 אחוז דיוק בסיווג בין כלל הסגנונות השונים, ו85 אחוז דיוק בקביעת סיווג בין שני סגנונות אפשריים.

פרק 1 - משוונים:

איזון ספקטרלי של אותות אודיו הינה מתודה חשובה ושימושית מאוד להמון אפליקציות מודרניות. החל מהרדיו שיושב ברכב, ועד לאולפני הפקת סאונד (מוסיקה, פודקאסטים וכו'..) – בכולם ניתן למצוא שימוש במשוווני אודיו.

בפרק זה אציג את המהלך השלם למימוש משוון פרמטרי בעל 7 יחידות עיבוד (7-Band Parametric EQ), החל מאבני הבניין הקטנה ביותר במערכת – המסנן (פילטר).

הגדרות ומושגי יסוד:

סינון אותות אודיו מתבצע ע"י אחד (או כמה) מסוגי המשוונים הבאים:

1. LP/HP – מסננים אלו מוגדרים ע"י "תדר קטעון" (מסומן: f_c). תדר זה מסמל ירידה במגניטודת האות בdb 3 ואחריו (או לפניו) מגיע תחום הקטעון (LP/HP בהתאמה).

2. BP/BS – מסנני band pass\stop אלו מסננים המקיימים מגוון תחומי מעבר וקטעון לאות בודד. מוגדים ע"י שני תדרי קטעון f_l, f_u אשר מסמלים את נק' ההנחת בdb 3. בין תחום תדרים זה מתקיים פס ה"מעבר" (בBS: קטעון). ניתן לתאר את תחום המעבר (או קטעון): $fb = f_u - f_l$.

3. Shelving Filters – פילטרים דומים בתכונותיהם לLP/HP, אך ההבדל הוא שלפילטר אין "תחום קטעון". במקום – הפילטר יודע לייצר הגבר \ הנחת לתחום תדר מסויים, **בעוד ששאר התחומים נשארים ללא עיבוד**.

4. Peak / Notch Filters – פילטרים דומים בתכונותיהם לBP/BS, אך, כמו במקרה הקודם, ההבדל הוא שלפילטר אין "תחום קטעון". במקום – הפילטר יודע לייצר הגבר \ הנחת לתחום תדר מסויים, **בעוד ששאר התחומים נשארים ללא עיבוד**.

פילטרים רקורסיביים:

הקדמה:

למימוש פילטרים ישנן שתי גישות. האחת: מימוש הפילטר תוך שימוש ב"אפסים" בלבד, במישור s (או z) – פילטר חסר משוב (לא רקורסיבי). השניה: מימוש הפילטר תוך שימוש ב"אפסים" וב"קטבים" (המעידים על היותו פילטר רקורסיבי, בפרספקטיבה כללית). לשתי הגישות ישנן משמעויות ותכונות שונות, בעוד שהנקודה המרכזית שניתן לגעת בה כרגע היא זמן החישוב של המערכת. בעזרת פילטר רקורסיבי (בעל קטבים), נוכל לממש מסנן מסדר נמוך בהרבה יותר מכך של מסנן בעל אפסים בלבד, על מנת לקרב פתרון של בעיה זהה. מכך, נסיק שפילטר רקורסיבי יוכל לבצע את החישוב בזמן קצר יותר (סדר נמוך יותר של מערכת).

את המסננים שלנו נתכנן במישור S (לפלס), המתאים לצורך המחשת הדיון לכרגע. נזכור, כי בפועל, התמרת לפלס תתאר משוואות דיפרנציאליות של פונקציות **רציפות**, בעוד, הייצוג הדיגיטלי של המידע מכריח אותנו לחשוב על עיבוד המידע שיש לנו כעיבוד של מידע **בדיד**. בהמשך, נדון בטרנספורמציה למישור Z המטפל בדיוק בנושא זה.

:HP/LP/BP

על מנת לעבד מידע המיוצג בתחום התדר, נוכל לעשות מימוש במסנני LP, BP ו – HP. נוכל לייצר תחומי הגבר וקיטעון לכל תדר ורוחב סרט במרחב האודיו שלנו, שבאופן טיפוסי נע בין 0-20 קילו הרץ (האוזן האנושית לא כך כך שומעת מעבר ל20 קילו הרץ).

ייצוג במישור לפלס של מסננים אלו (בתלות בתדר הקטעון):

$$H_{HP}(s) = \frac{s^2}{s^2 + \frac{\omega_c}{Q_\infty}s + \omega_c^2} \quad H_{LP}(s) = \frac{\omega_c^2}{s^2 + \frac{\omega_c}{Q_\infty}s + \omega_c^2},$$

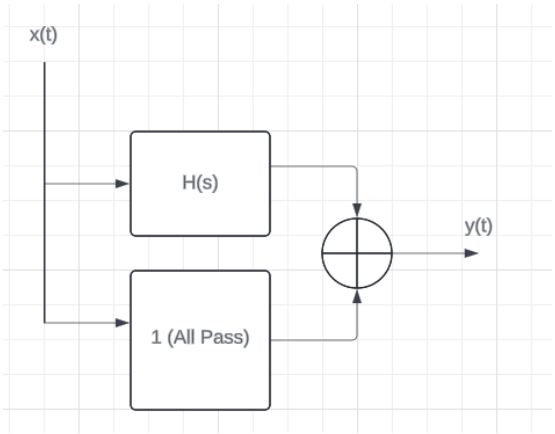
$$H_{BS}(s) = \frac{s^2 - \omega_c^2}{s^2 + \frac{\omega_c}{Q_\infty}s + \omega_c^2}. \quad H_{BP}(s) = \frac{\frac{\omega_c}{Q_\infty}s}{s^2 + \frac{\omega_c}{Q_\infty}s + \omega_c^2},$$

כאשר רוחב הסרט של המסננים ייקבע ע"פ גורם הטיב (Q), אשר מייצג את היחס בין רוחב הסרט לתדר הקטעון.

:Shelving Filters

בעיה מסויימת צצה כאשר אנחנו רוצים לעבד תחום תדר, ולהשאיר את שאר התחומים בלי עיבוד. המסננים שהצגתי עד כה אמנם מעבדים את התחום המתבקש אך מייצרים קטעון לשאר תחום התדר.

לשם כך, נתאר את העיבוד הרצוי כמערכת שמשלבת העברת אות ללא עיבוד, ובמקביל העברת אות ליחידת עיבוד. כלומר:



במוצא נקבל את האות המעובד בתחום הרצוי, בעוד בשאר התחומים האות יעבור כמו שהוא ללא עיבוד.

תיאור מתמטי של מסננים אלו בעבור LP\HP:

$$H(s) = 1 + H_{HP}(s) \quad H(s) = 1 + H_{LP}(s)$$

ופונקציית התמסורת למערכת מסדר שני התלויה בפרמטר הגבר כלשהו v0:

עבור LP:

$$H(s) = \frac{s^2 + \sqrt{2V_0}s + V_0}{s^2 + \sqrt{2}s + 1}$$

עבור HP:

$$H(s) = \frac{V_0 s^2 + \sqrt{2V_0}s + 1}{s^2 + \sqrt{2}s + 1}$$

(ניתן לשים לב לאופי התגובה כאשר לוקחים את s לאינסוף ולאפס).

דוגמא לתגובות התדר של מסננים אלו כתלות בערכי v0 שונים:

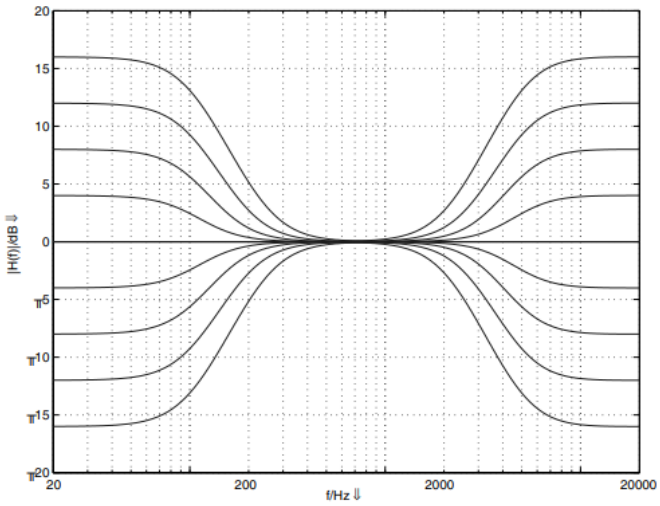


Figure 5.14 Frequency responses of second-order low-/high-frequency shelving filters – low-frequency shelving filter $f_c = 100$ Hz (parameter V_0), high-frequency shelving filter $f_c = 5000$ Hz (parameter V_0).

Peak Filters:

סוג נוסף של מסנן המתוכנן לתת הגבר או הנחת לתחומי תדר ספציפיים, בעוד שאר התדרים נשארים בלא עיבוד. בניגוד למסננים בסעיף הקודם, הנותנים מענה לתחום "כללי" התואם לאיזורי תדר "גבוהים" או "נמוכים", מסנן זה יותר דומה לBP, ולכן BP יהיה התשתית שעליה נעבוד. בדומה למסננים מסעיף קודם, גם את מסנן זה ניישם ע"י שליחה של האות דרך נתיב ALL PASS, ובמקביל שליחה דרך נתיב BP, ונסכום את המוצא. כלומר:

$$\boxed{H(s) = 1 + H_{BP}(s).}$$

דוגמא לתגובות התדר של מסננים אלו בעבור ערכי v0 שונים:

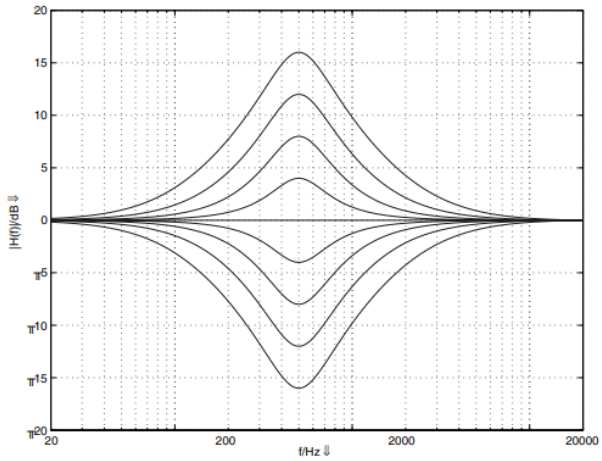


Figure 5.16 Frequency response of a peak filter – $f_c = 500$ Hz, $Q_\infty = 1.25$, cut parameter V_0.

טרנספורמציית Z (מעבר מתכנון אנלוגי לספרתי):

באמור, לאחר שדנים בתכנון מערכת כלשהי במישור לפלס, יש לזכור כי במישור זה אנו משערכים פתרונות למשוואות **דיפרנציאליות ואינטגרליות** של פונקציות **רציפות**. כאשר אנו דנים בעיבוד **ספרתי** של אותות, יש לייצר את השערוך הנ"ל בעולם ה**בדיד** (לדוג' – במחשב לא קיימת שום סוג של "רציפות". המידע הוא **אותות** בדידים המיוצגים על **מרחב** בדיד).

על מנת ליישם את העיבוד בעולם הבדיד, ניתן לבצע מעבר ממישור לפלס למישור Z, אשר מייצג לנו את התכנון של מערכת בדידה, עבור אותות בדידים.

המטרה תהיה, למפות את הקטבים והאפסים, אשר מגדירים את המערכת במישור לפלס, למישור Z.

מיפוי זה הולך להתקבל ע"פ הטרנספורמציה הבאה:

$$w_d = \tan\left(w_c * \frac{T}{2}\right) \qquad s = \frac{2}{T}\frac{z-1}{z+1}.$$

תתקבל לנו פונקצית תמסורת במישור Z. נתאר אותה באופן כללי בעזרת המקדמים שלה:

$$H(z) = \frac{a_0 + a_1 z^{-1} + a_2 z^{-2}}{1 + b_1 z^{-1} + b_2 z^{-2}},$$

ניתן להראות, בהצבה של משתנה עז $K = \tan\left(w_c * \frac{T}{2}\right)$ שאפשר לקבל טבלה המייצגת את כל ערכי המקדמים הנדרשים, לכל פונקציית תמסורת מהפונקציות שנדרשות לנו למימוש הפילטרים שציינו למעלה (טבלה בעמוד הבא).

טבלת המקדמים כתלות בפרמטרים wc,v0:

Table 5.3 Peak filter design.

Peak (boost $V_0 = 10^{G/20}$)				
a_0	a_1	a_2	b_1	b_2
$\frac{1+\frac{V_0}{Q_\infty}K+K^2}{1+\frac{1}{Q_\infty}K+K^2}$	$\frac{2(K^2-1)}{1+\frac{1}{Q_\infty}K+K^2}$	$\frac{1-\frac{V_0}{Q_\infty}K+K^2}{1+\frac{1}{Q_\infty}K+K^2}$	$\frac{2(K^2-1)}{1+\frac{1}{Q_\infty}K+K^2}$	$\frac{1-\frac{1}{Q_\infty}K+K^2}{1+\frac{1}{Q_\infty}K+K^2}$
Peak (cut $V_0 = 10^{-G/20}$)				
a_0	a_1	a_2	b_1	b_2
$\frac{1+\frac{1}{Q_\infty}K+K^2}{1+\frac{V_0}{Q_\infty}K+K^2}$	$\frac{2(K^2-1)}{1+\frac{V_0}{Q_\infty}K+K^2}$	$\frac{1-\frac{1}{Q_\infty}K+K^2}{1+\frac{V_0}{Q_\infty}K+K^2}$	$\frac{2(K^2-1)}{1+\frac{V_0}{Q_\infty}K+K^2}$	$\frac{1-\frac{V_0}{Q_\infty}K+K^2}{1+\frac{V_0}{Q_\infty}K+K^2}$

Table 5.4 Low-frequency shelving filter design.

Low-frequency shelving (boost $V_0 = 10^{G/20}$)				
a_0	a_1	a_2	b_1	b_2
$\frac{1+\sqrt{2V_0}K+V_0K^2}{1+\sqrt{2}K+K^2}$	$\frac{2(V_0K^2-1)}{1+\sqrt{2}K+K^2}$	$\frac{1-\sqrt{2V_0}K+V_0K^2}{1+\sqrt{2}K+K^2}$	$\frac{2(K^2-1)}{1+\sqrt{2}K+K^2}$	$\frac{1-\sqrt{2}K+K^2}{1+\sqrt{2}K+K^2}$
Low-frequency shelving (cut $V_0 = 10^{-G/20}$)				
a_0	a_1	a_2	b_1	b_2
$\frac{1+\sqrt{2}K+K^2}{1+\sqrt{2V_0}K+V_0K^2}$	$\frac{2(K^2-1)}{1+\sqrt{2V_0}K+V_0K^2}$	$\frac{1-\sqrt{2}K+K^2}{1+\sqrt{2V_0}K+V_0K^2}$	$\frac{2(V_0K^2-1)}{1+\sqrt{2V_0}K+V_0K^2}$	$\frac{1-\sqrt{2V_0}K+V_0K^2}{1+\sqrt{2V_0}K+V_0K^2}$

Table 5.5 High-frequency shelving filter design.

High-frequency shelving (boost $V_0 = 10^{G/20}$)				
a_0	a_1	a_2	b_1	b_2
$\frac{V_0+\sqrt{2V_0}K+K^2}{1+\sqrt{2}K+K^2}$	$\frac{2(K^2-V_0)}{1+\sqrt{2}K+K^2}$	$\frac{V_0-\sqrt{2V_0}K+K^2}{1+\sqrt{2}K+K^2}$	$\frac{2(K^2-1)}{1+\sqrt{2}K+K^2}$	$\frac{1-\sqrt{2}K+K^2}{1+\sqrt{2}K+K^2}$
High-frequency shelving (cut $V_0 = 10^{-G/20}$)				
a_0	a_1	a_2	b_1	b_2
$\frac{1+\sqrt{2}K+K^2}{V_0+\sqrt{2V_0}K+K^2}$	$\frac{2(K^2-1)}{V_0+\sqrt{2V_0}K+K^2}$	$\frac{1-\sqrt{2}K+K^2}{V_0+\sqrt{2V_0}K+K^2}$	$\frac{2(K^2/V_0-1)}{1+\sqrt{2/V_0}K+K^2/V_0}$	$\frac{1-\sqrt{2/V_0}K+K^2/V_0}{1+\sqrt{2/V_0}K+K^2/V_0}$

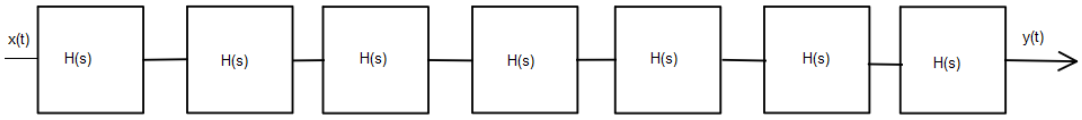
בטבלה זו נשתמש בפועל, על מנת לחשב את המקדמים לכל אחד מהפילטרים הנדרשים למימוש המשוון.

משוון פרמטרי בעל 7 יחידות עיבוד (7 Band Parametric EQ):

כעת אוכל לממש את המערכת הנדרשת לעיבוד האות הדיגיטלי. לבחירה לממש את המשוון בעזרת 7 יחידות עיבוד יש משמעויות נרחבות, שמפעת גבולותיו הסופיים של הפרוייקט לא נכנסתי לחקור אותם (לעת עתה). הבחירה ב7 יחידות הינה (פסאדו) שרירותית, מכיוון שמהיכרות עם מוצרים שקיימים בשוק, רוב המשוונים שראיתי עד כה היו בעלי 7 יחידות.

אחלק את המשוון ל2 יחידות בקצוות הספקטרום, Low Shelving ו – High Shelving, ו – 5 יחידות במרכז הספקטרום: Peak Filters. לכל יחידה יש שליטה בפרמטרים fc,G כאשר fc הוא תדר הקיטעון בהרצים, וG הוא ההגבר הנדרש בdB.

בעזרת הטבלאות הפרמטריות, נחשב את המקדמים של כל אחת משבעת פונקציות התמסורת, לאחר מכן נעביר את האות עיבוד בקו הכולל את שבעת המסננים (כאשר בעבור מסנן במצב BYPASS, ללא עיבוד, נקבל במוצא את אות הכניסה).



משוונים - מימוש:

את המימוש בפרק זה אראה בעזרת פונקציות מטלב, ואיעזר בגרפים להמחשת העיבוד.

ישנן 3 פונקציות מטלב (מצורפות להגשה) שמטרתן לחשב, ולהחזיר את מקדמי המסננים. שלושת הפונקציות מקבלות כקלט את התדר המרכזי fc, ההגבר Gdb ותדר הדגימה של המערכת fs. חישוב מקדמי הPEAK דורש גם את גורם הטיב ולכן הפונקציה מקבלת גם את רוחב הסרט כפרמטר.

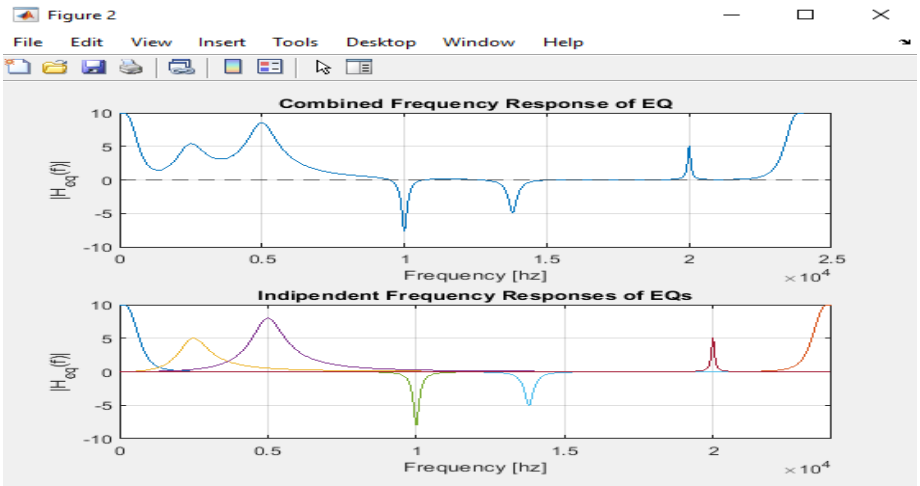
חתימות הפונקציות:

```
function [b,a] = calc_bp_coeffs(fc,Gdb,BW, fs)
function [b,a] = calc_hp_coeffs(fc,Gdb, fs)
function [b,a] = calc_lp_coeffs(fc,Gdb, fs)
```

כל אחת משבעת יחידות העיבוד מקבלת את המקדמים שלה, התדר המרכזי ביחס אליו היא מעבדת, ההגבר ותדר הדגימה (כאשר הגבר 0 אומר שהמערכת לא מעבדת והיא בBYPASS).
לדוגמא:

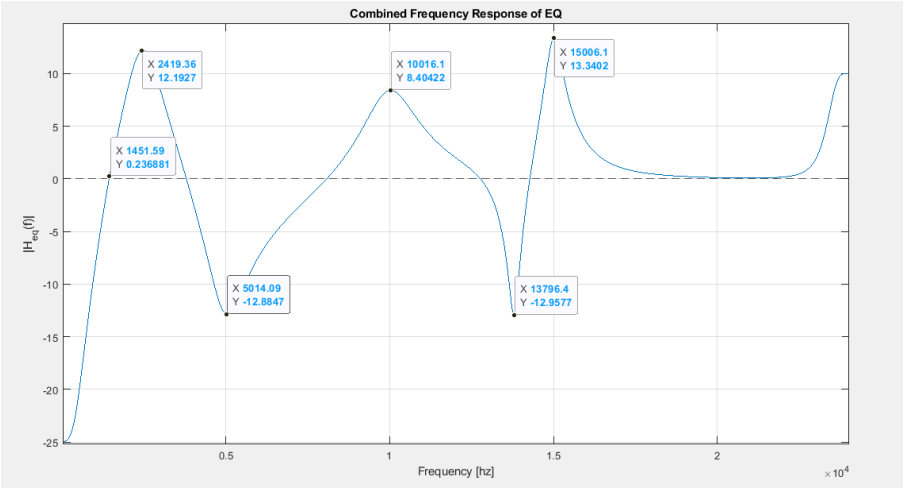
```
% calc coeffs
h1.Gdb = 10; h1.fc = 500;
[h1.b,h1.a] = calc_lp_coeffs(h1.fc,h1.Gdb ,fs);
```

הדגמה של תגובת התדר הכוללת של המשוון המורכב מ-7 יחידות עיבוד:

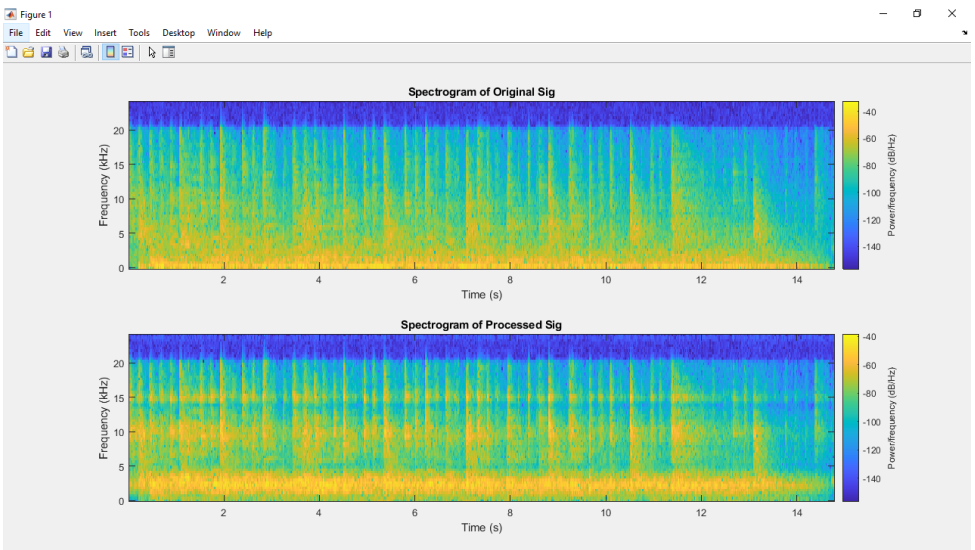


הדגמה בעבור אות שמע קצר: ספקטרוגרמת המקור, קונפיגורציית המשוון והאות המעובד:

משוון:



ספקטרוגרמת מקור ואות מעובד:



IIR ל-FIR:

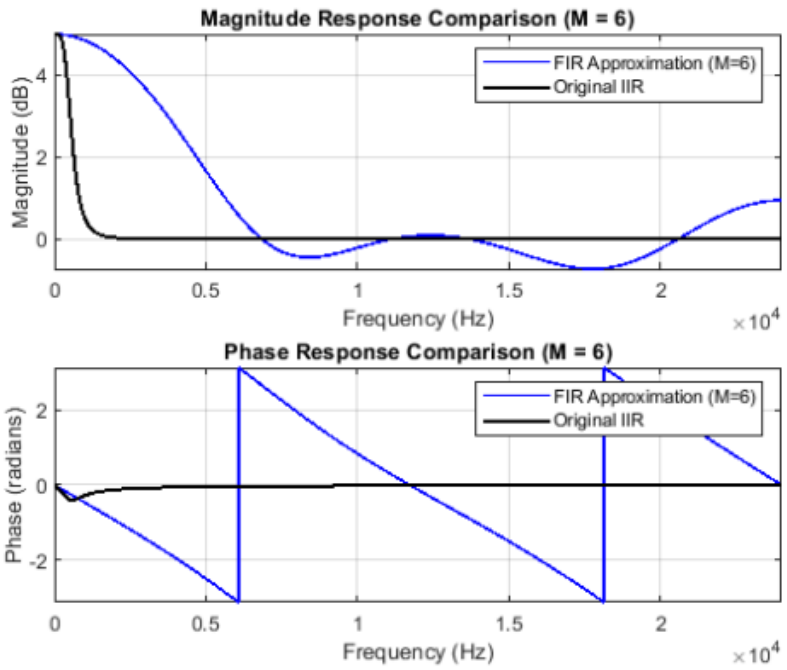
המימוש של המשוון בעזרת מסנני IIR מציג בעיה אחת עיקרית: עיוותי הפאזה שנוצרים עקב פאזה לא לינארית. ישנו דיון גדול בעולם האודיו האם העיוותים הללו, ברמת שימוש סטנדרטית, בכלל ניתנים להבחנה באוזן האנושית (כל הנושא נקרא Frequency Smearing).

מפאת סדר גודל הפרוייקט, אני לא נכנס למימושים מפורטים של משוון פאזה לינארית. מה שכן ארצה לעשות, זה לגעת בנקודה – מדוע מסנני FIR ומשוני פאזה לינארית לא פופולרים כל כך בעולם עיבוד האודיו (אדגיש שהם אכן קיימים, וקיימים אלגוריתמים למימושים שונים ומתוחכמים של משוונים אלו, אך לא נפוצים כל כך).

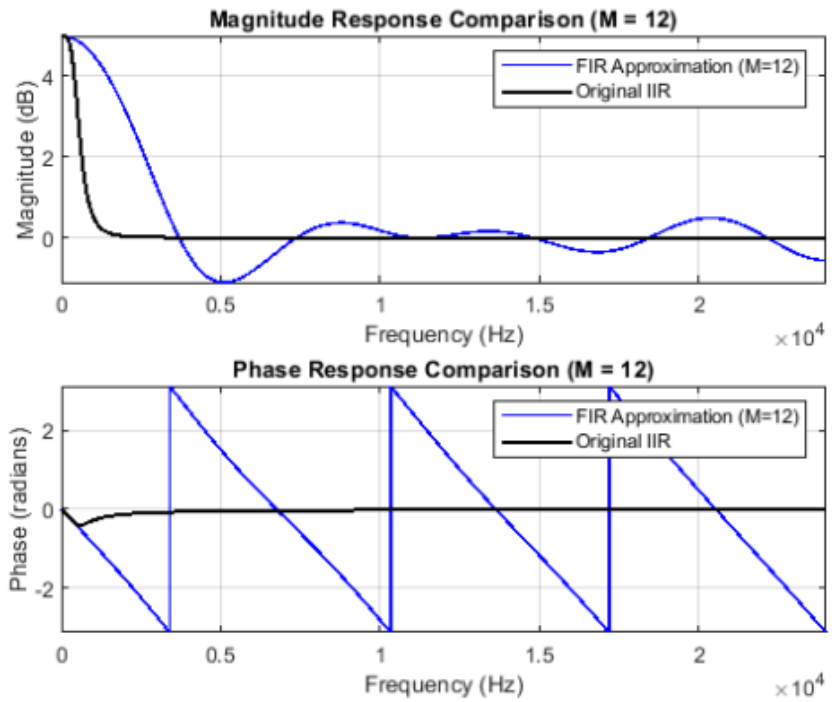
בעיה אחת עיקרית שנולדת מהרצון להשתמש במסנן FIR, היא השהיית החבורה הארוכה שמסנן כזה מוליד איתו. מסנן FIR יידרוש סדר גבוה בהרבה (מאוד!) על מנת לשערך את אותה תגובת תדר. סדר גבוה כזה ייגרור עיכוב גדול בעיבוד, מה שיכול להיות קריטי (מאוד!) לשימושי אודיו, ויהרוס את חוויית השמע במוצא המשוון (באפליקציות לשימוש בזמן אמת בעיקר).

בחלק קצר זה, אראה השוואה בין תגובות התדר המשוערכות, עבור סדרים שונים של מסנני FIR. מכיוון שיש לנו כבר את תגובת התדר הרצויה, מתכנוני המסננים הקודמים (IIR), אשתמש בטכניקת Frequency Sampeling על מנת לשערך את תגובת התדר של מסנן FIR מתוך התגובה שתכננתי בעזרת מסנן IIR.

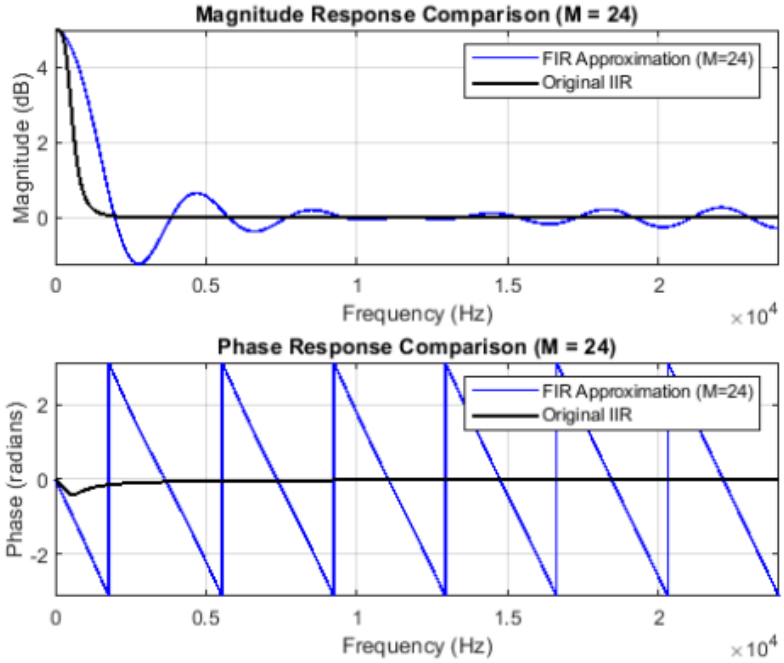
סדר 6:


Magnitude Response Comparison (M = 6)
FIR Approximation (M=6)
Original IIR
Magnitude (dB)
Frequency (Hz)
×10^4
Phase Response Comparison (M = 6)
FIR Approximation (M=6)
Original IIR
Phase (radians)
Frequency (Hz)
×10^4

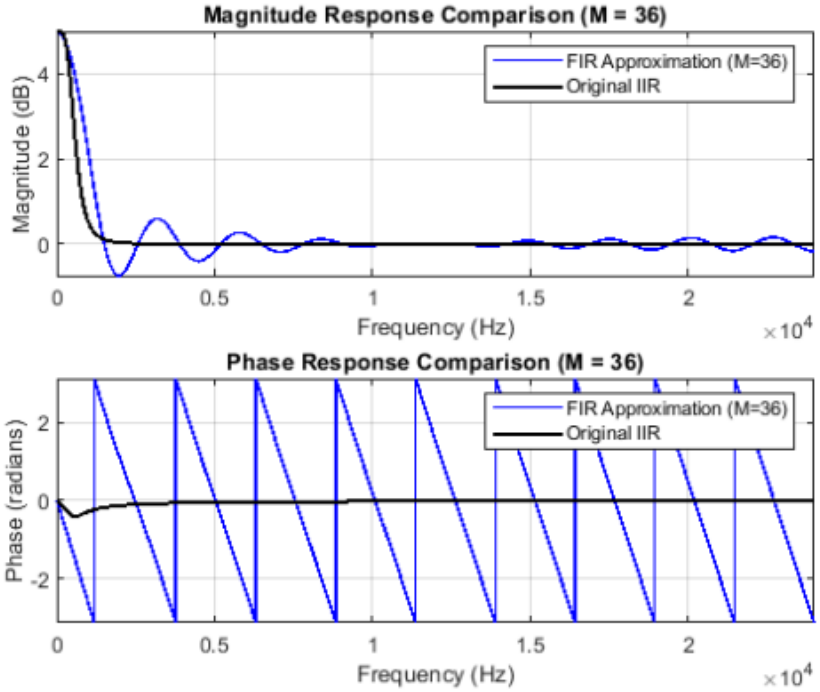
סדר 12:


Magnitude Response Comparison (M = 12)
FIR Approximation (M=12)
Original IIR
Magnitude (dB)
Frequency (Hz)
×10^4
Phase Response Comparison (M = 12)
FIR Approximation (M=12)
Original IIR
Phase (radians)
Frequency (Hz)
×10^4

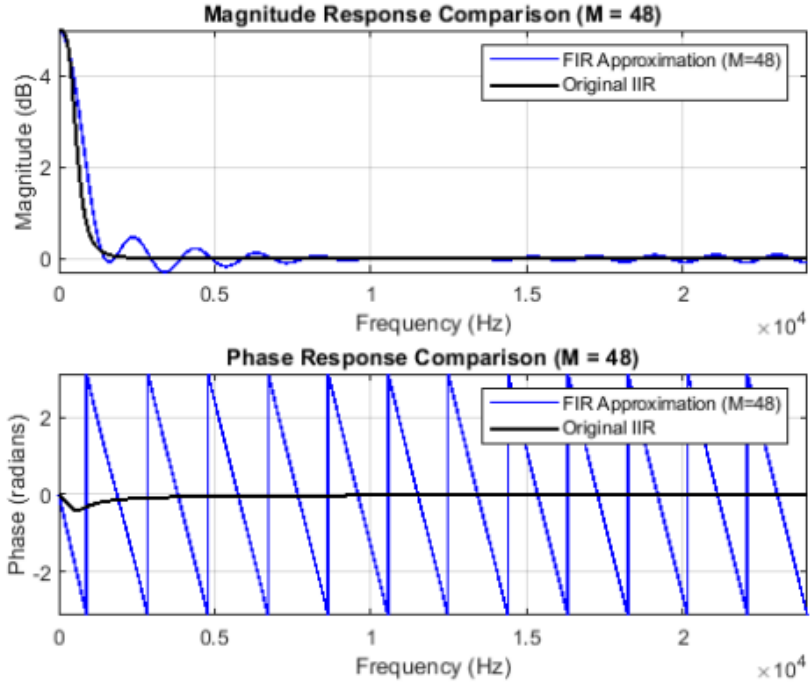
סדר 24:


Magnitude Response Comparison (M = 24)
FIR Approximation (M=24)
Original IIR
Magnitude (dB)
Frequency (Hz)
×10^4
Phase Response Comparison (M = 24)
FIR Approximation (M=24)
Original IIR
Phase (radians)
Frequency (Hz)
×10^4

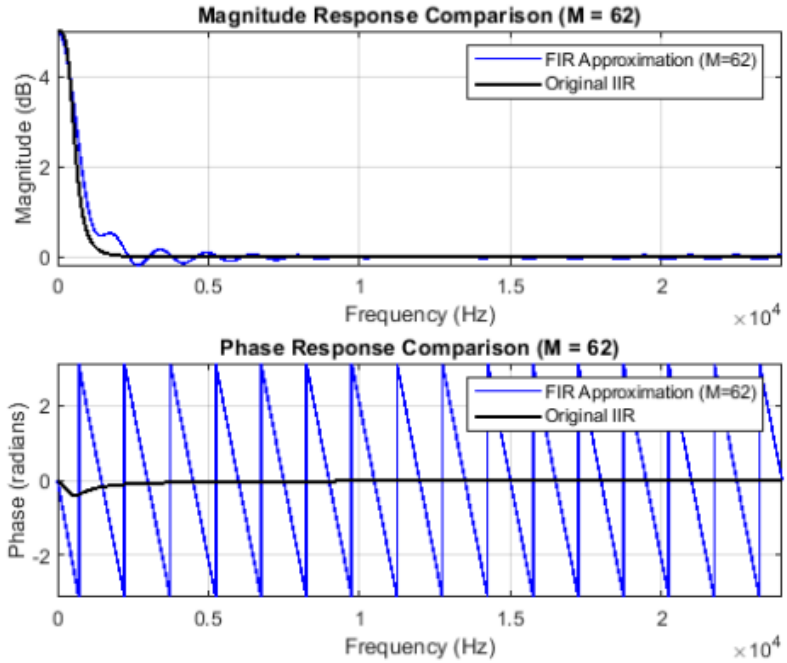
סדר 36:


Magnitude Response Comparison (M = 36)
FIR Approximation (M=36)
Original IIR
Magnitude (dB)
Frequency (Hz)
×10^4
Phase Response Comparison (M = 36)
FIR Approximation (M=36)
Original IIR
Phase (radians)
Frequency (Hz)
×10^4

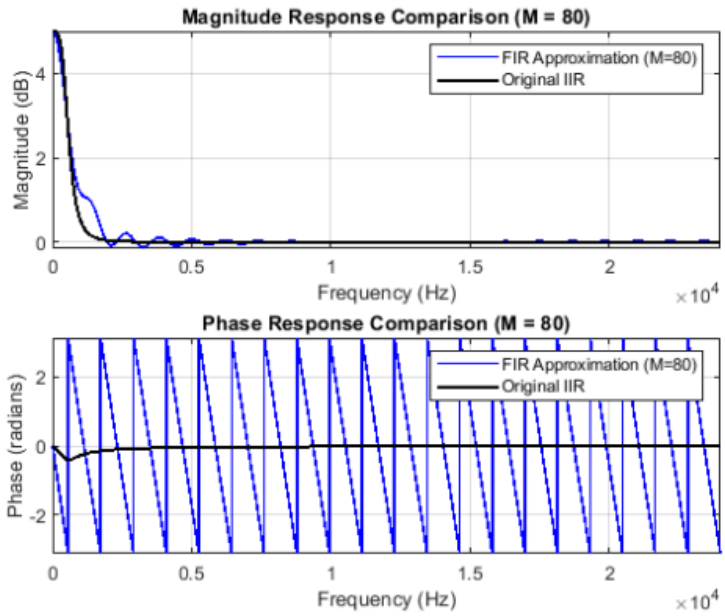
סדר 48:


Magnitude Response Comparison (M = 48)
FIR Approximation (M=48)
Original IIR
Magnitude (dB)
Frequency (Hz)
×10^4
Phase Response Comparison (M = 48)
FIR Approximation (M=48)
Original IIR
Phase (radians)
Frequency (Hz)
×10^4

סדר 62:


Magnitude Response Comparison (M = 62)
FIR Approximation (M=62)
Original IIR
Magnitude (dB)
Frequency (Hz)
×10^4
Phase Response Comparison (M = 62)
FIR Approximation (M=62)
Original IIR
Phase (radians)
Frequency (Hz)
×10^4

סדר 80:



כפי שניתן לראות, רק מסנן מסדר 80 מתחיל לקרב את תגובת התדר באופן ראוי.

השימושים הנרחבים בשוק עדיין נשארו בידי מסנני IIR בעלי פאזה מינימלית.

משוונים - מסקנות:

1. שימוש במסנני IIR הוא האופטימלי ביותר למימוש מערכת זו.
2. ע"י פרמטריזציה של המסננים הצלחנו לממש יחידות הגבר והנחת הנשלטות ע"י תדר קטעון והגבר.
3. ע"י שרשור של יחידות הגבר אלו בטור, ניתן לייצר יחידה גדולה היכולה לשלוט על כמה תחומי תדר ביחד.
4. ניתן לממש מגוון של משוונים ע"י משחק עם כמות יחידות ההגבר הקיימות במערכת.

פרק 2 – סימולציית חדר:

גלי קול הינם גלי לחץ אוויר הנעים ומתפשטים במרחב בו הם חיים. כמו כל גל המתפשט, הגל יכול לעבור תופעות שונות ומשונות במפגש עם תכונות תווך מסויימות.

גל קול אשר נולד בחדר סגור, יתפשט עד שייפגע בעצמים שונים בתווך, או בקירות התווך. פגיעה זו, תייצר אפקטים של החזרה ובליעה של הגל. ניתן להבין, שהגל שמגיע לנו לאוזן בחדר כזה הוא תרכובת של גלי "מקור" וגלים "חוזרים".

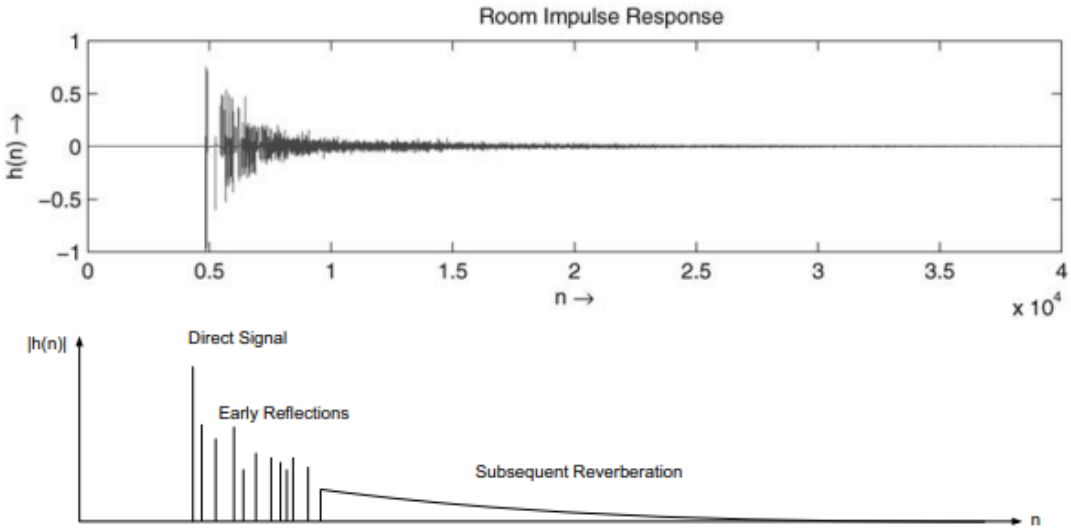
בעולם הדיגיטלי, ניתן לעשות שימוש בשיטות עיבוד על מנת לדמות תופעות אשר קורות מהחזרים שונים שלא קיימים בסיגנל המקורי, ה"יבש", ביניהם לדוגמא נמצא הREVERB (הדהוד). רוורב הוא אפקט אשר מדמה את החזרי גלי הקול ע"פ אופי החדר אותו מנסים לדמות. בפרק זה אכנס לשיטות העיבוד בהן משתמשים על מנת לייצר את האפקט.

הרעיון עובד באופן הבא: אם נוכל לשערך איך נראית תגובת החדר להלם, ידוע לנו כי כעת נוכל לקחת כל סיגנל, להעביר אותו קונבולוציה עם התגובה להלם של החדר ובכך נקבל את הסיגנל המקורי "מעובה" באפקט ההדהוד של החדר. אראה אלגוריתמים שונים המתעסקים בסימולציות חדר, כמו MOORER, SCHROEDER. בעזרת אלגוריתמים אלו נוכל להגיע לאפקט החדר הרצוי.

אקוסטיקת חדר:

ניתן להרחיב המון על התפשטות של גלים בחדר כתלות בגיאומטרית החדר, מקדמי החזר ובליעה של חומרים בחדר ואלמנטים נוספים. אני אסתפק בלהציג את התהליך של התפשטות גל לחץ אוויר בחדר באופן פשוט על מנת לפתח את השפה המשותפת עליה ייבנה האלגוריתם הדיגיטלי שממדל את האפקט.

תגובת ההלם של נק' בחדר יכולה להיות ממודלת באופן הבא: האות המקורי, מלווה בהחזרים "מידיים" יותר (מהקירות וכדומה), כאשר מספר ההחזרים הולך וגדל עם הזמן, סכום ההחזרים מתווסף לכדי אות עם דעיכה אקספוננציאלית הנקרא "רוורב עקיבה" (subsequent reverb).



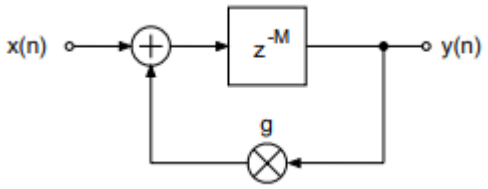
ניתן להבחין בשני היבטים חשובים:

- האחד, צפיפות ההחזרים, ההולכת וגדלה עם הזמן. ככל שהזמן עובר, יותר ויותר החזרים יגיעו מהנתיבים הפחות "ישירים" בחדר וייצרו צפיפות גדולה יותר של גלים חוזרים.
- השני, המרווחים בין ההחזרים, שלא מפולגים בצורה אחידה אלא בצורה יותר "אקראית". הדבר נובע גם מההבדל בין גלים בעלי נתיב ישיר להחזר, לבין גלים בעלי נתיבים יותר מורכבים.

שתי אבחנות אלו יעזרו לנו לבנות את האלגוריתם שיממש את הדימוי של החזרי החדר, אלגוריתם Schroder.

אלגוריתם Schroeder - מימוש:

אלגוריתם Schroeder ממדל את התופעות האקוסטיות שהובאו בפרק הקודם.
ליבת האלגוריתם – הינו פילטר רקורסיבי בעל הגבר משוב קטן מאחד, כך שתיווצר "סדרה" שדועכת כתלות בהגבר. לפילטר יש יחידת השהייה בנתיב הלוך (Forward Path) כך שגם המשוב המונחת וגם האות הנקי שמסתכם איתו, יעברו השהיה של M דגימות:



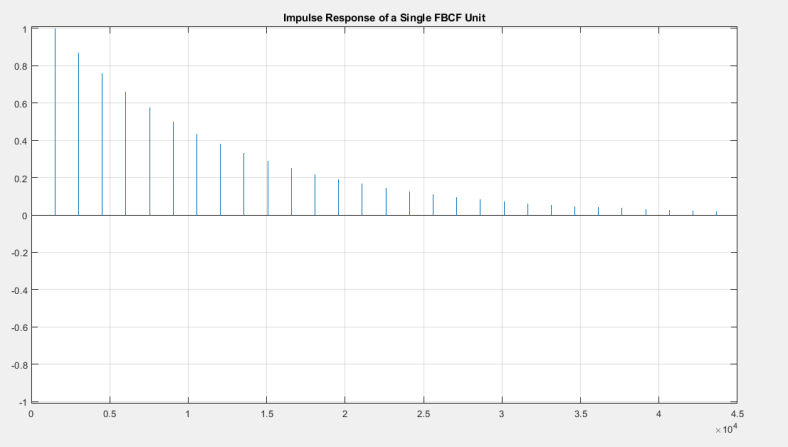
יחידה אחת כזו תמדל סדרה של החזרים בדעיכה אקספוננציאלית (g<1).

לפילטר בודד ישנה חתימת הפונקציה:

```
function [y,buff] = fbcomb(x,buff,n,d,G_linear)
```

כאשר, על מנת לעזור לבנות את המערכת השלמה, כל יחידה כזו מקבלת גם את הבאפר שהיא תמלא, גם את הדגימה שהיא מעבדת וגם את ההשהיה בזמן d.

בעבור פרמטרי התחלה מסויימים, תגובת ההלם של יחידה בודדת כזו תיראה כך:

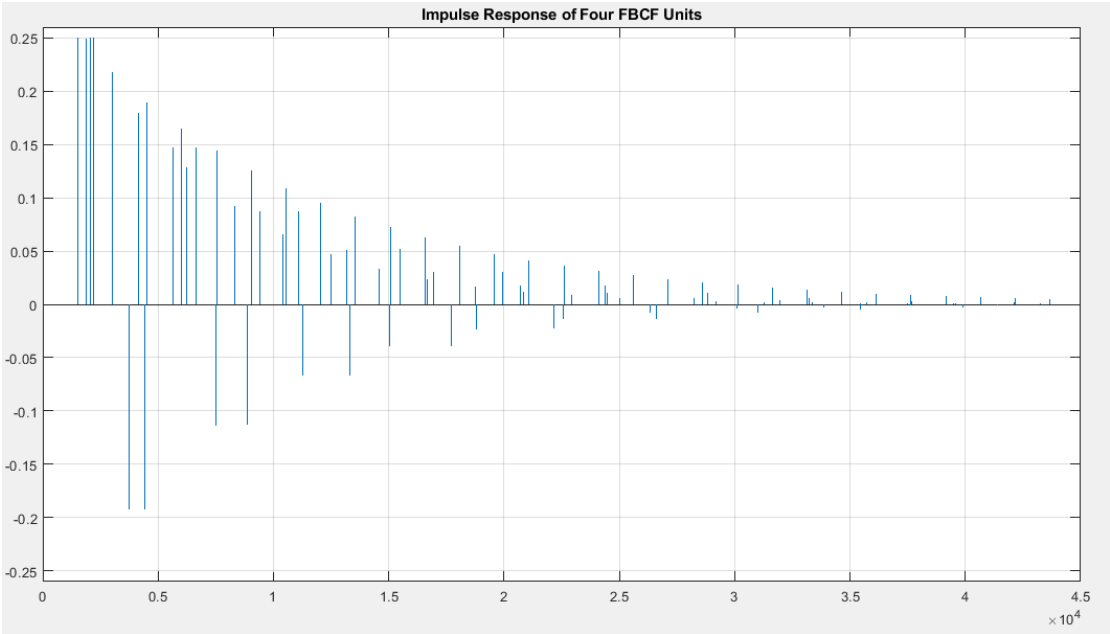


ניתן לראות כי ההחזרים דועכים כמו שציפינו, אך על מנת למדל באופן מלא את החדר, נצטרך לטפל בפילוג ההחזרים, שלא יהיה אחיד.

לצורך כך, נבנה יחידה המורכבת מארבעה פילטרים כאלו, בעלי זמני d שונים והגברים שונים, כך שהפילוג בין מופעים של גלים חוזרים יקבל אלמנט יותר "אקראי" (לא באמת אקראי במלו מובן המילה, אבל מפולג באופן יותר טוב מהפילוג האחיד של יחידה בודדת).

```
[w1,buffer1] = fbcomb(x(n,1),buffer1,n,d1,g1);
[w2,buffer2] = fbcomb(x(n,1),buffer2,n,d2,g2);
[w3,buffer3] = fbcomb(x(n,1),buffer3,n,d3,g3);
[w4,buffer4] = fbcomb(x(n,1),buffer4,n,d4,g4);
combPar = 0.25*(w1 + w2 + w3 + w4);
```

תגובת ההלם של ארבע יחידות מקבילות תיראה כך:



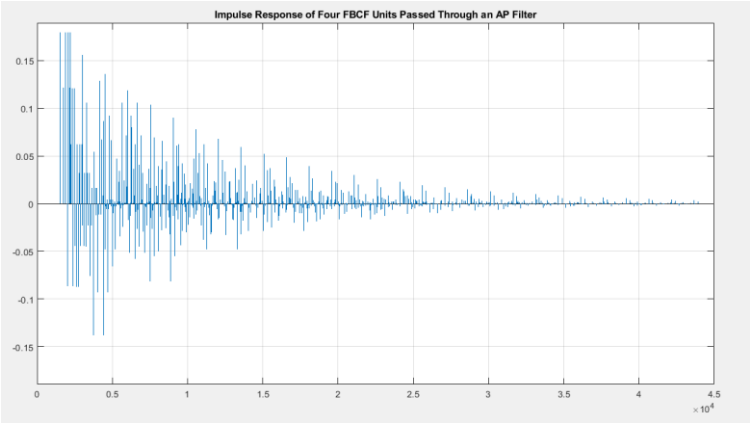
וקיבלנו תגובה מפולגת באופן קצת יותר מוצלח מהתגובה הקודמת.

עוד טכניקת עיבוד שתמומש כאן, היא מעבר ביחידת All Pass של האות המורכב מארבעת יחידות העיבוד שמומשו עד כה. המעבר במסנן AP "מעבה" את צפיפות ההחזרים, מה שעוזר לנו לייצר אפקט השהייה פסאדו אקראי נוסף.

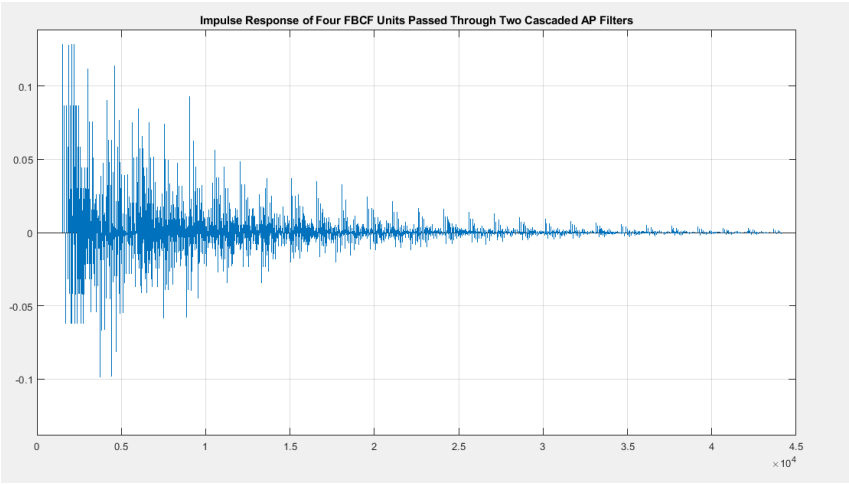
חתימת הAP פילטר בתכנית:

```
function [y,buff] = apfilt(x,buff,n,d,G_linear)
```

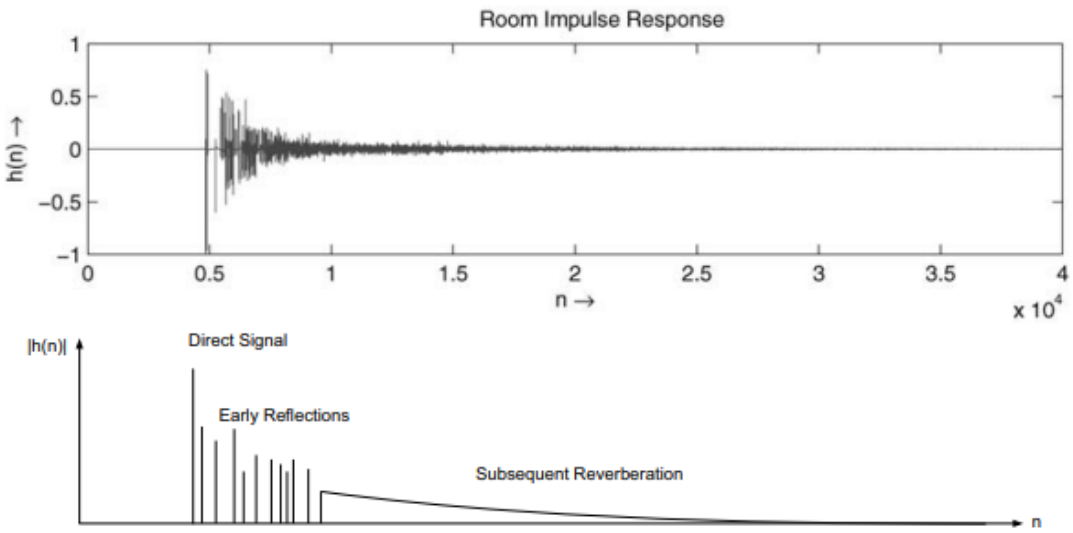
מעבר של מוצא היחידה המורכבת מארבעה FBCF בAP פילטר:



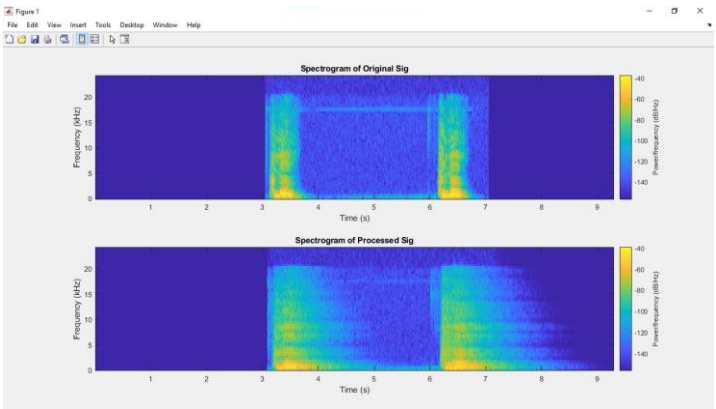
לאחר מעבר בAP נוסף (שני):



ולמעשה הצלחנו למדל באופן יחסית טוב את הגרף שתיאר את התנהגות החדר, שממנו התחיל הפרק:



הדגמה בעבור אות שמע קצר:



מוצגים: ספקטרוגרמת המקור והאות המעובד. ניתן לראות את "מריחת" ההחזרים בזמן.

סימולציית חדר - מסקנות:

1. ע"י מימוש אלגוריתם Schroder ניתן לקבל מגוון גדול של גרפים המייצגים "תגובות הלם" של חדרים בעלי אופי שונה.
2. הפרמטרים של המסננים שהשתמשנו בהם (השהייה והגבר) ימדלו את ההחזרים של גלי הקול בחדרים בעלי אופי ואקוסטיקה שונה.
3. ניתן לחקות תופעה אנלוגית בקירוב מאוד (מאוד!) טוב, ע"י אמצעים דיגיטליים בלבד, אם לוקחים בחשבון את התופעות הזניחות בעולם האנלוגי.

פרק 3: למידה עמוקה – סיווג סגנונות מוזיקליים של קטעי אודיו:

מבוא:

בפרק זה אחקור את בעיית הקלסיפיקציה של קטעי אודיו לכדי אחד מעשרה סגנונות מוסיקליים. הבסיס לפרוייקט זה הוא בסיס נתונים המכיל 1000 דגימות אודיו באורך 30 שניות ממגוון של עשרה סגנונות מוסיקליים שונים.

<u>שלבים:</u>

1. היכרות עם הדאטא: אציג את הדאטא, את רקע לחקר הפיצ'רים ואת תהליך בניית מרחב הפיצ'רים.
2. שימוש במודל בסיס הנבחר מתוך Kaggle ובו קיימת ארכיטקטורה בעלת ביצועים שאנסה לשפר.
3. חקר של מגוון של ארכיטקטורות נוספות שונות וביצועיהן.

היכרות עם הדאטא:

בסיס הנתונים בבעיה זו הוא טבלה, המכילה פיצ'רים מרכזיים שעובדו ממאגר נתונים של 1000 דוגמיות סאונד באורך 30 שניות ותיוגן.

הפיצר'ים המרכזיים:

- תוחלת ושונות של כרומוגרמה (כרומוגרמה היא תמונה המכילה את התו המתנגן לכל יחידת זמן בממוצע, כמו ספקטרוגרמה רק שציר התדר מתחלק ע"פ החלוקה לתווים)
- תוחלת ושונות של אנרגיה הממוצעת האגורה באות (RMS)
- תוחלת ושונות של התדר שמתחתיו מתרכזת 85% מהאנרגיה האגורה באות (או יותר נכון, בפריים נתון מתוך האות. Roll Off Frequency)
- תוחלת ושונות לרוחב פס של האות (Spectral Bandwidth)
- תוחלת ושונות של "מרכז המסה" של פיזור האנרגיה בתדר (Spectral Centroid)
- MFCC's – הסבר עליהם יכול להיות מצורף בספר פרוייקט משל עצמו. בקצרה – הMFCC היא ספקטרוגרמה של ספקטרוגרמה (כלומר, מדד למחזוריות של התמרת פורייה של האות).

יחד עם הטבלה, מגיע גם מאגר הדוגמיות שממנה יצרו את הטבלה. באחת הארכיטקטורות שמימשתי גם ביצעתי עיבוד למאגר המידע הזה בעצמי (על מנת להתאים אותו לכניסת הארכיטקטורה) בעזרת ספריית Librosa הנועדה לעיבוד אודיו.

הדאטא עצמו מגיע מאוזן, לכן לא היו הרבה בעיות של עיבוד מקדים. עבודה שחוזרת על עצמה הרבה באינטרנט היא חלוקת כל דוגמית של 30 שניות ל10 דוגמיות של 3 שניות וחילוץ הפיצ'רים משם, על מנת לעבות את מאגר המידע. לאחר שראיתי שהעבודה המקדימה הזו חוזרת על עצמה הרבה, מימשתי אותה גם.

הערה: המידע עצמו מגיע כמידע טבלאי המציג את כל הפיצ'רים הנדרשים על מנת לסווג את הקטעים. כחלק מחקר וסקרנות אישית, העמקתי לתוך ויזואליזציה של הדאטא והוצאת מאפייני קטעי אודיו, למרות שאין צורך בכך (שוב, המידע מגיע כמידע טבלאי מחולק ומסודר). על מנת לנסות לשמור על מסמך זה תמציתי לגבי הפרוייקט, אני מדלג פה על הצגת הדאטא עצמו. להרחבה, ניתן להיכנס למחברות Feature Generation,Data Visualisation בו צירפתי גרפים של כל אחד מהפיצ'רים עצמו.

ארכיטקטורת בסיס:

ארכיטקטורת הבסיס היא רשת הנוירונים המובאת בקישור:

https://www.kaggle.com/code/aasimahmed04/music-genre-classifier

מבנה הרשת:

-

```
In [12]: #Creating a Neural Network
         model = Sequential()

         model.add(Flatten(input_shape=(58,)))
         model.add(Dense(256, activation='relu'))
         model.add(BatchNormalization())
         model.add(Dense(128, activation='relu'))
         model.add(Dropout(0.3))
         model.add(Dense(10, activation='softmax'))

         model.summary()
```

חלוקת הנתונים לסדרת אימון, ולידציה ומבחן:

- הדאטא מחולק ל680 דגימות לאימון, 170 דגימות לולידציה ו150 דגימות למבחן.

Split the data to train and test:

```
In [10]: X_train, X_test, y_train, y_test = train_test_split(scaled_data, labels, test_size=0.15,shuffle=True, random_state=1)
         X_train, X_val, y_train, y_val = train_test_split(X_train, y_train, test_size=0.2,shuffle=True, random_state=1)

In [11]: print("Train data size: ", X_train.shape);
         print("Validation data size: ",X_val.shape);
         print("Test data size: ", X_test.shape);

         Train data size:  (680, 58)
         Validation data size:  (170, 58)
         Test data size:  (150, 58)
```

קימפול והרצת הלמידה:

Compiling and fitting:

```
In [13]: #Compiling & Fitting the Model
         adam = keras.optimizers.Adam(learning_rate=1e-4)
         model.compile(optimizer=adam,
                       loss='sparse_categorical_crossentropy',
                       metrics=['accuracy'])
         hist = model.fit(X_train, y_train, validation_data= (X_test, y_test),
                          epochs=100,
                          batch_size=32)
```

תוצאות ריצת האימון:

```
Epoch 100/100
22/22 ━━━━━━━━━━━━━━━━━━━━ 0s 5ms/step - accuracy: 0.9075 - loss: 0.3250 - val_accuracy: 0.7667 - val_loss: 0.8645
```

בחינת הרשת על סט המבחן:

```
In [14]: y_predict = np.argmax(model.predict(X_test),axis=1)

         5/5 ━━━━━━━━━━━━━━━━━━━━ 0s 17ms/step
```

```
In [15]: #Test Accuracy
         acc = np.mean(y_predict == y_test)*100
         print("test accuracy = %.2f"%acc, "%");

         test accuracy = 72.00 %
```

ניתן לראות כי הרשת מגיעה לדיוק של 72 אחוז מול 10 מחלקות שונות (לא רע בכלל!).

שיפור ארכיטקטורת הבסיס (ע"י ניסיון למזער התאמת יתר):

בחלק זה אני בוחן מספר אפשרויות למלחמה בהתאמת יתר. לבסוף אסכם ואציג את הרשת המוצלחת ביותר.

<u>המחברת:</u>

Simple Neural Network - Based on Kaggle's aasimahmed04 - L1 Regularization

במחברת זו הוספתי רגולריזציה מסוג L1. שיטה זו היא אחת משתיים לרגולריזציה על קצב שינוי הפרמטרים של הרשת. ההבדל העיקרי בשיטה זו היא ששיטה זו יכול להגיע למצב שהיא "משתקת" נוירון (מגיעה לאפס). הרשת:

Improving The Network:

```
In [17]: #Creating a Neural Network
         model = Sequential()

         model.add(Flatten(input_shape=(58,)))
         model.add(Dense(256,kernel_regularizer=regularizers.l1(0.0005), activation='relu'))
         model.add(BatchNormalization())
         model.add(Dense(128,kernel_regularizer=regularizers.l1(0.0005), activation='relu'))
         model.add(Dropout(0.3))
         model.add(Dense(10, activation='softmax'))

         model.summary()
```

דיוק האימון:

```
Epoch 100/100
22/22 ━━━━━━━━━━━━━━━━━━━━ 0s 4ms/step - accuracy: 0.9299 - loss: 1.5153 - val_accuracy: 0.7667 - val_loss: 1.9878
```

דיוק המבחן:

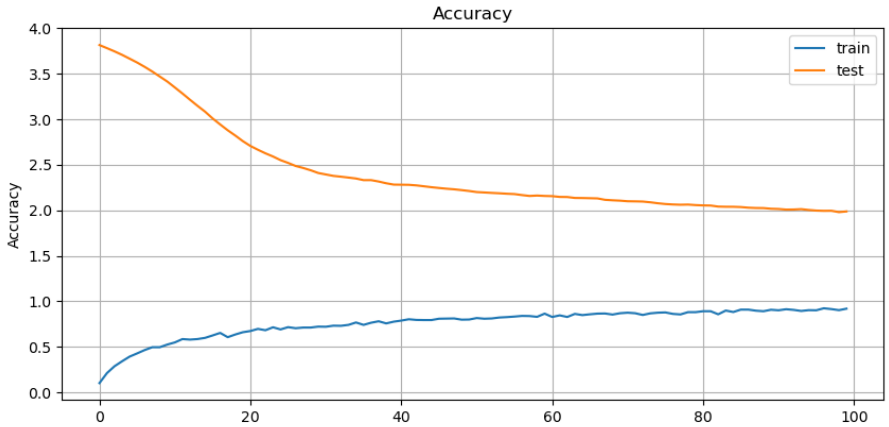
```
In [19]: y_predict = np.argmax(model.predict(X_test),axis=1)

         5/5 ━━━━━━━━━━━━━━━━━━━━ 0s 17ms/step
```

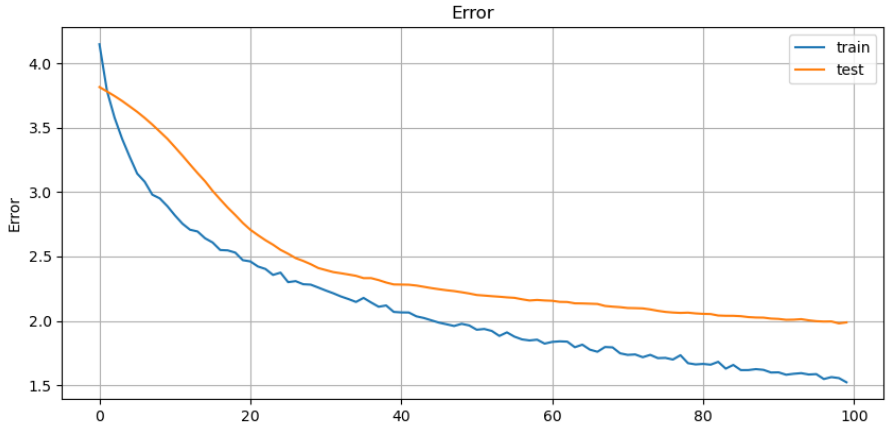
```
In [20]: #Test Accuracy
         acc = np.mean(y_predict == y_test)*100
         print("test accuracy = %.2f"%acc, "%");

         test accuracy = 76.67 %
```

התקבל שיפור של כ4.6 אחוז!

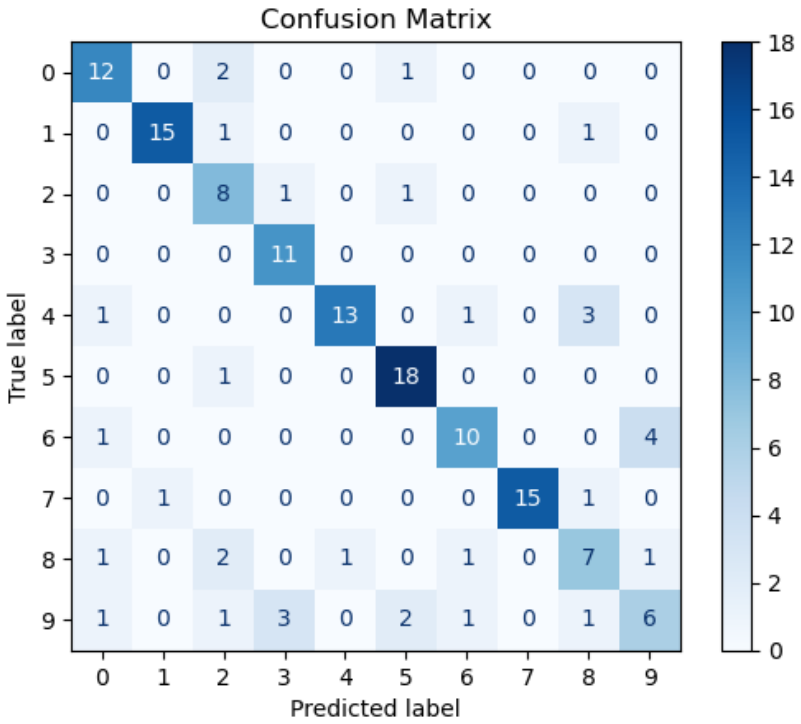
גרפים:

דיוק מבחן לעומת אימון:





מטריצת ערבול:



ניתן לראות ממטריצת הערבול כי אין מחלקה מסויימת שממופה באופן מסיבי למחלקה ספציפית אחרת. השגיאות מפוזרות לאורך המחלקות השונות.

<u>סיכום:</u> התקבל שיפור משמעותי ביותר בביצועי הרשת. הרשת עצמה עדיין לא מצליחה לשבור את מחסום ה-80 אחוז דיוק ולכן המשכתי לבחון ארכיטקטורות אפשריות נוספות.

<u>המחברת:</u>

Simple Neural Network - Based on Kaggle's aasimahmed04 - L2 Regularization

רגולריזציה זו לא יכולה "לשתק" אף נוירון. הרשת:

Improving The Network:

In [19]:
```
#Creating a Neural Network
model = Sequential()

model.add(Flatten(input_shape=(58,)))
model.add(Dense(256,kernel_regularizer=regularizers.l2(0.0005), activation='relu'))
model.add(BatchNormalization())
model.add(Dense(128,kernel_regularizer=regularizers.l2(0.0005), activation='relu'))
model.add(Dropout(0.3))
model.add(Dense(10, activation='softmax'))

model.summary()
```

הרצת האימון:

```
Epoch 100/100
22/22 ━━━━━━━━━━━━━━━━━━━━ 0s 4ms/step - accuracy: 0.9391 - loss: 0.4095 - val_accuracy: 0.7400 - val_loss: 0.9812
```

הרצת המבחן:

Test set:

In [21]:
```
y_predict = np.argmax(model.predict(X_test),axis=1)
```

```
5/5 ━━━━━━━━━━━━━━━━━━━━ 0s 17ms/step
```

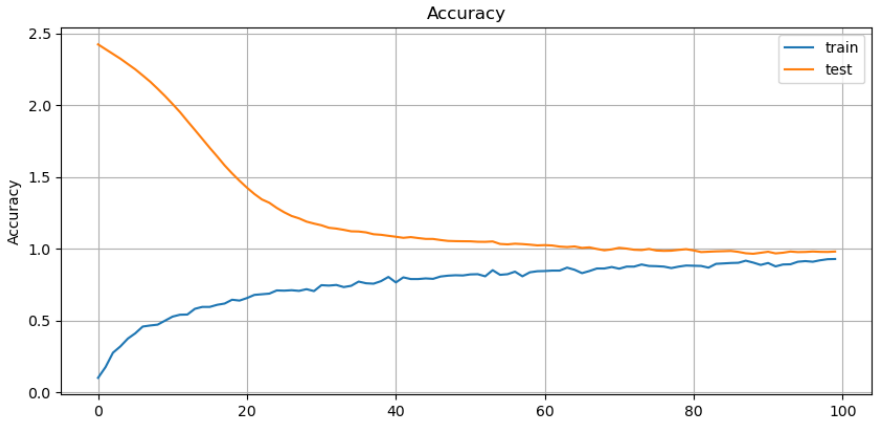
In [22]:
```
acc = np.mean(y_predict == y_test)*100
print("test accuracy = %.2f"%acc, "%");
```

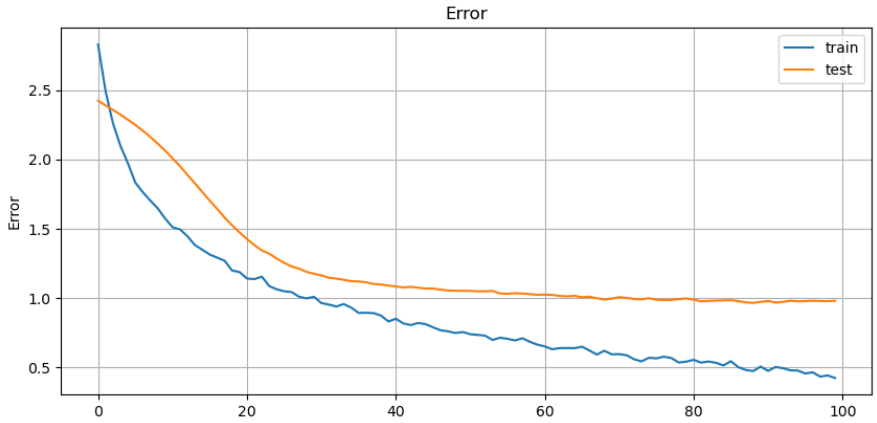
```
test accuracy = 74.00 %
```

קיבלתי דיוק של 74 אחוז. ביצועים מעט פחות טובים מהשיפור הקודם שעשיתי, אך עדיין שיפור ביחס למודל הבסיס.

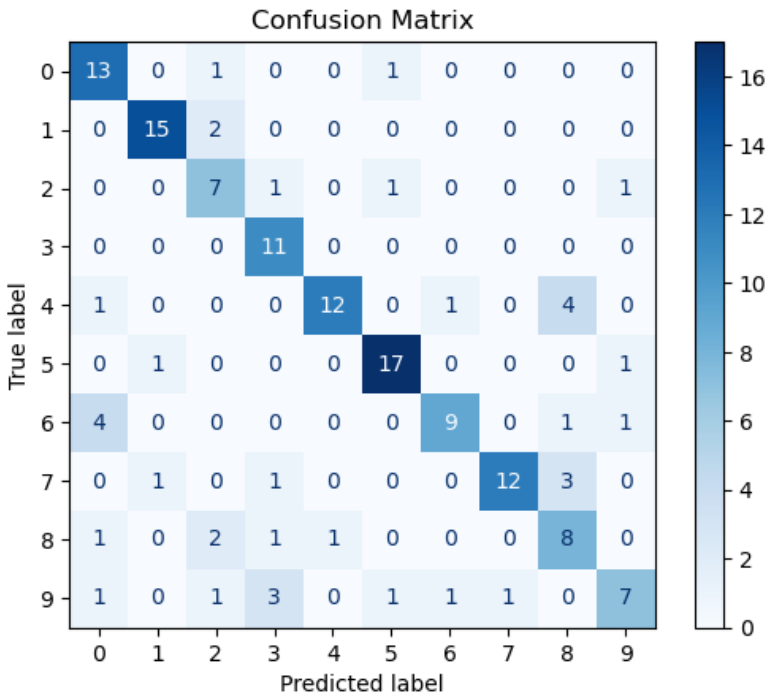
גרפים:

דיוק מבחן לעומת אימון:





מטריצת ערבול:



ניתן לראות באותו האופן, כי הטעויות מתפזרות על מספר יחסית גדול של מחלקות ולא ממחלקות ספציפיות למחלקות ספציפיות.

המחברת:

Simple Neural Network - Based on Kaggle's aasimahmed04 - L2 Regularization SECOND STAGE ONLY

במחברת זו, מימשתי את הרגולריזציה על השכבה השניה בלבד. מתוך קו המחשבה ש**הדאטא המוצג במחברת Data Visualization אינו מייצר עקומי החלטה מורכבים מידי במישור הדאטא.** לכן, ארצה לייצר רשת זהירה יותר בשלב השני לעומת השלב הראשון.

הרשת:

Improving The Network:

```
In [19]: #Creating a Neural Network
         model = Sequential()

         model.add(Flatten(input_shape=(58,)))
         model.add(Dense(256, activation='relu'))
         model.add(BatchNormalization())
         model.add(Dense(128,kernel_regularizer=regularizers.l2(0.0005), activation='relu'))
         model.add(Dropout(0.3))
         model.add(Dense(10, activation='softmax'))

         model.summary()
```

אימון הרשת:

```
Epoch 100/100
22/22 ━━━━━━━━━━━━━━━━━━━━ 0s 4ms/step - accuracy: 0.9139 - loss: 0.4019 - val_accuracy: 0.7471 - val_loss: 0.7761
```

סדרת המבחן:

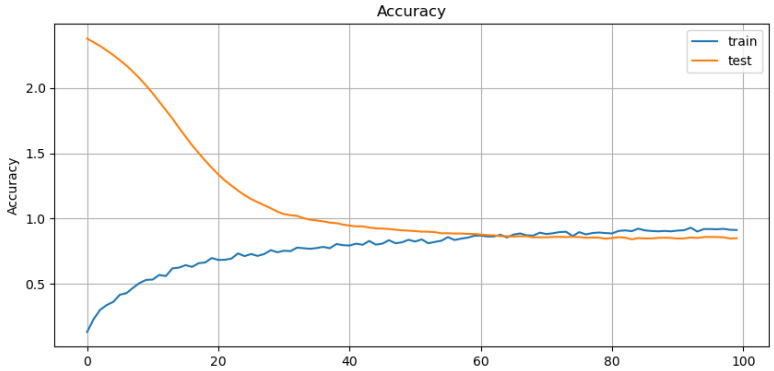
Test set:

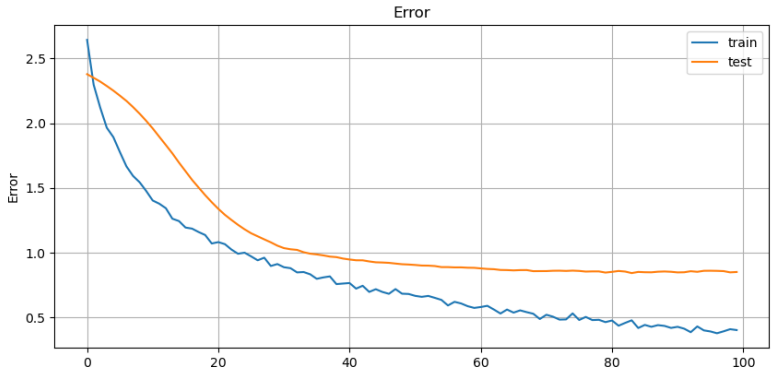
```
In [21]: y_predict = np.argmax(model.predict(X_test),axis=1)

         5/5 ━━━━━━━━━━━━━━━━━━━━ 0s 17ms/step

In [22]: #Test Accuracy
         acc = np.mean(y_predict == y_test)*100
         print("test accuracy = %.2f"%acc, "%");

         test accuracy = 75.33 %
```

קיבלתי דיוק של 75.3 אחוז בסדרת המבחן. דיוק טוב יותר משל הרשת בעלת רגולריזציה מסוג L2 לשתי השכבות, אך פחות טוב משל הרשת בעלת רגולריזציה מסוג L1.

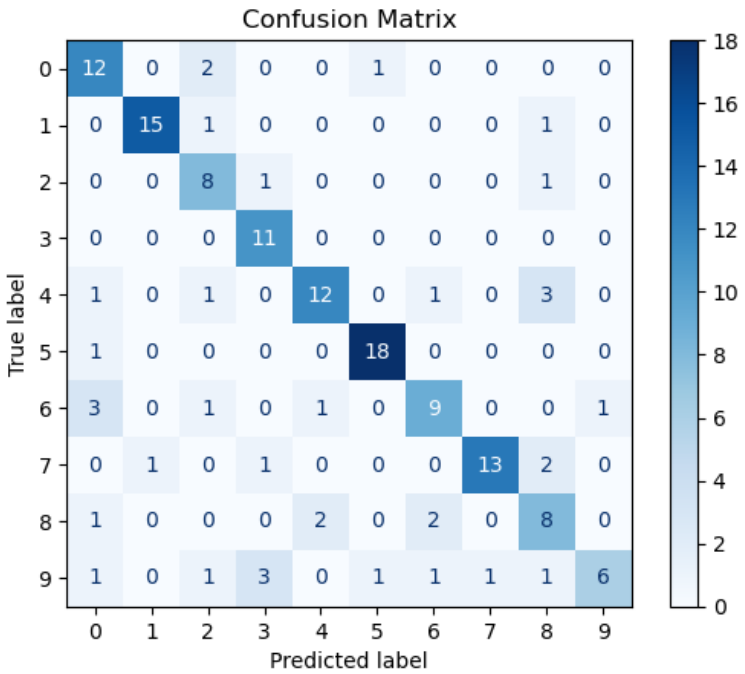
גרפים:

דיוק אימון לעומת מבחן:





מטריצת ערבול:



המסקנות מאוד זהות. השגיאות מפוזרות בין מגוון של מחלקות.

למידת העברה - VGGish:

אחת הבעיות העיקריות בפרוייקט זה היה הרצון לרתום למשימה גם רשת בארכיטקטורת CNN. בעיה שהתעוררה היא העובדה שהמידע המעובד מקטעי האודיו הוא לא תמונה או מידע דו מימדי כלשהו. המידע לכל דוגמת אודיו מוחזק בצורת וקטור (של ערכי ממוצע, שונויות וערכים נוספים).

בהמשך, הראיתי גם ביצוע של רשת CNN על המידע (ע"י סידור צורת הכניסה לרשת ומימוש רשת המבצעת קונבולוציות חד ממדיות) – כל זאת בהמשך.

תוך כדי קריאה וחשיבה על מודל VGG שלמדנו עליו, הסתבר לי שגוגל התמודדו עם הבעיה הזו בעבר. גוגל ייצרו מודל בשם VGGish, שנוצר על מנת לתת את המענה לרשת קונבולוציה שיודעת להתמודד עם המידע שמוחזק עבור קטעי אודיו.

רשת זו מצפה לכניסה ברזולוציה של 16 קילו-הרץ ובאורך של שניה, כלומר, 16 אלף דגימות. מוצא הרשת הוא וקטור באורך 128. במימוש שלי, מוצא המערכת של גוגל ייכנס לClassifier משלי (כלומר, אשתמש בעיבוד המקדים של רשת הקונבולוציה כסוג של Embedding לקטע האודיו).

הורדת המודל והעברת קטעי האודיו בVGGish מובאת במחברת:

Transfer Learning - VGGish (Google Model) - Preprocessing And Arranging the Data (Notebook 1 of 2)

Loading the architecture:

In [3]:
```
# Load VGGish feature extractor from TensorFlow Hub
vggish = hub.load("https://tfhub.dev/google/vggish/1")
```

מוצא הVggish נשמר כמידע טבלאי שיוזן למסווג שמעליו.

Create the .csv file out of the data outside the VGGish network:

This step is in order to not have to process the data through the VGGish model each run (the process is very slow)

In [8]:
```
# Convert to DataFrame
df_features = pd.DataFrame(X)
df_labels = pd.DataFrame(y, columns=['label'])

# Combine features and labels into one DataFrame
df = pd.concat([df_features, df_labels], axis=1)

# Save to CSV
csv_file_path = 'vggish_features_labels.csv'
df.to_csv(csv_file_path, index=False)

print(f"Dataset saved to {csv_file_path}")
```

```
Dataset saved to vggish_features_labels.csv
```

המחברת (של המסווג):

Transfer Learning - VGGish (Google Model) - Preprocessing And Arranging the Data (Notebook 2 of 2)

במחברת זו מימשתי את המסווג מעל רשת הקונבולוציה:

```
In [6]: # Build a simple classifier on top of VGGish features
        model = Sequential()

        model.add(Flatten(input_shape=(X_train.shape[1],)))
        model.add(Dense(128,kernel_regularizer=regularizers.l2(0.0005), activation='relu'))
        model.add(BatchNormalization())
        model.add(Dense(64,kernel_regularizer=regularizers.l2(0.0005), activation='relu'))
        model.add(Dropout(0.5))
        model.add(Dense(16,kernel_regularizer=regularizers.l2(0.0005), activation='relu'))
        model.add(Dropout(0.5))
        model.add(Dense(10, activation='softmax'))

        model.summary()
```

ריצת האימון:

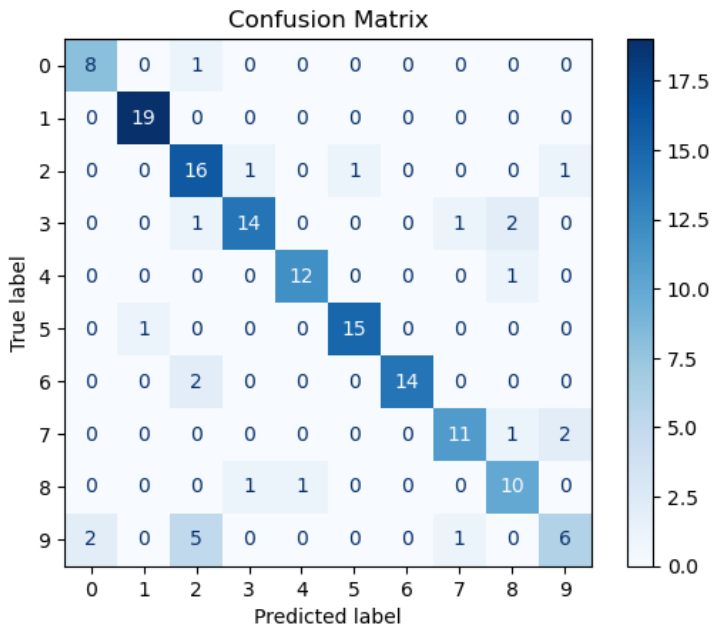
```
Epoch 100/100
22/22 ━━━━━━━━━━━━━━━━━━━━ 0s 4ms/step - accuracy: 0.7958 - loss: 0.7165 - val_accuracy: 0.7941 - val_loss: 0.7923
```

ריצת המבחן:

```
In [8]: y_predict = np.argmax(model.predict(X_test),axis=1)

        5/5 ━━━━━━━━━━━━━━━━━━━━ 0s 24ms/step

In [9]: acc = np.mean(y_predict == y_test)*100
        print("test accuracy = %.2f"%acc, "%");

        test accuracy = 83.33 %
```

הגעתי ל83.3 אחוז דיוק! שיפור של 7 אחוז מהביצוע הטוב ביותר בחלק הקודם.

מטריצת ערבול:



השגיאה של תיוג 2 לעומת תווית אמיתית 9 קצת מעצבנת אבל כל השאר מפוזרות יחסית.

בנקודה זו חשבתי איך אפשר לרתום את נושא הגדלת בסיס הנתונים. אין הרבה משמעות למיסוך תדר לספקטרוגרמות או כרומוגרמות מכיוון שהערך שנכנס לטבלת הנתונים הוא ממוצע או שונות של תדר לאורך כל הקטע (איפוס עמודה לא ישנה בצורה דרסטית את הממוצע הזה).

כחלק מהמידע הטבלאי המוגש לפרוייקט, קיימת טבלה שנבנתה על חלוקת קטעי האודיו של 30 שניות ל10 קטעים של 3 שניות. חקר הביצועים על הדאטא הזה מניב תוצאות מעניינות.

<u>המחברת:</u>

Transfer Learning - VGGish (Google Model) - Preprocessing And Arranging EXTENDED DATASET (Notebook 1 of 2)

- במחברת זו חזרתי על אותו העיבוד והעברת המידע ברשת VGGish אך חילקתי כל קטע אודיו ל-10 קטעים של 3 שניות.

<u>המחברת:</u>

Transfer Learning - VGGish (Google Model) - Classifier EXTENDED DATASET (Notebook 2 of 2)

המסווג היושב מעל הרשת VGGish הוא בדיוק אותו המסווג שישב על הרשת הקודמת.

הרצת האימון:

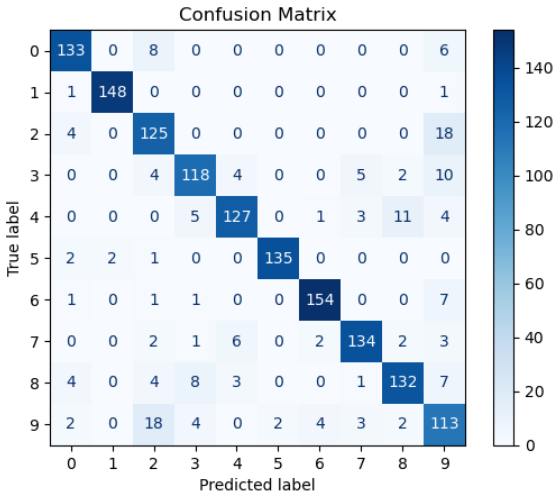
```
Epoch 100/100
213/213 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step - accuracy: 0.9141 - loss: 0.3466 - val_accuracy: 0.8629 - val_loss: 0.5197
```

הרצת המבחן:

```
In [8]: y_predict = np.argmax(model.predict(X_test),axis=1)

        47/47 ━━━━━━━━━━━━━━━━━━━━ 0s 3ms/step

In [9]: acc = np.mean(y_predict == y_test)*100
        print("test accuracy = %.2f"%acc, "%");

        test accuracy = 87.99 %
```

הצלחתי להגיע לסיווג של 88 אחוז! הביצוע הכי טוב (בפער) עד כה.

מטריצת ערבול:



השגיאה של פרדיקציה של 2 לעומת תווית 9 חוזרת על עצמה שוב פה. מסתמן שבעבור ארכיטקטורה זו הבלבול בין המחלקות האלו ספציפית הוא מאוד בעייתי (גם פרדיקציה של 9 לעומת תווית 2).

קיימת מחברת נוספת מצורפת להגשה, המציגה מימוש של רשת CNN, אך במקום שכבות קונבולוציה דו מימדית אני ממש שכבות חד מימדיות. הרשת הזאת הניבה ביצועים נוראיים ואני לא מצרף סקירה שלה פה, ניתן לפתוח את המחברת להריץ ולראות.

Music Genre Classifier - CNN Model (1D , Based on Network from HW)

מסקנות:

4. **שימוש ברשת CNN מסוג VGGish (בחלק של למידת העברה) הניב את התוצאות הכי טובות, גם עבור בסיס נתונים רגיל וגם עבור בסיס הנתונים המורחב.** ניתן לחשוב על הרשת הזו כסוג של "מקודדת" את המאפיינים השונים באודיו לכדי וקטור מאפיינים באורך 128, ולהשתמש במסווג על וקטור מאפיינים זה. (Embedding).
5. **שימוש ברשתות קונבולוציה** צריך לעבור אדפטציה משמעותית לניתוח קטעי אודיו. לעתיד ניתן יהיה לבדוק מה יהיו ביצועים של רשת קונבולוציה על תמונות (של ממש) המייצגות אודיו (כמו ספקטרוגרמה, או ספסטרום).
6. **שימוש ברשת רגילה** נותן ביצועים טובים בארכיטקטורה בעלת רגולריזציה מסוג L1. במעבר חופשי והשמעה אקראית של דגימות מתוך בסיס הנתונים, אני לא בטוח שאדם אקראי היה מצליח לזהות סגנונות באחוזים יותר טובים (למשל, במחברת Data Visualization מוצגים שני קטעי אודיו. אני לא בטוח בכלל שאדם אקראי שישמע את השיר המובא ב Hip Hop Sample לא יחשוב שזו דגימת רוק בכלל..)

מקורות לVGGish:

1. CNN ARCHITECTURES FOR LARGE-SCALE AUDIO CLASSIFICATION – מצורף להגשה (PDF)
2. https://www.mathworks.com/help/audio/ref/vggishfeatures.html
3. https://colab.research.google.com/drive/1TbX92UL9sYWbdwdGE0rJ9owmezB-Rl1C

רשימת שאר המקורות:

[4] Digital Audio Signal Processing, Udo Zolzer

[5] **Audio & Speech Processing with Matlab**, Paul R. Hill.

[6] Hack Audio, An Introduction to Computer Programming and Digital Signal Processing in MATLAB, Eric Tarr

[7] Theory and Applications of Digital Speech Processing, Lawrence R Rabiner, Ronal W. Schafer.

[8] Introduction to Digital Speech Processing, Lawrence R. Rabiner, Ronal W. Schafer.

[9] Speech and Audio Processing - a MATLAB Based Approach, Ian Vince McLoughlin

[10] All About Audio Equalization, Vesa Välimäki and Joshua D. Reiss, Applied Science

[11] Speech Recognition Using Articula