הפקולטה למדעים מדויקים
ע״ש ריימונד ובברלי סאקלר
אוניברסיטת תל אביב

יזכור
מלחמת חרבות ברזל

החוג למדעי המחשב (0368)
עיבוד ספרתי של אותות (3464)
(גרסה ארוכה)

מרצה: יעקב שטיין

תשפ"ד, סמסטר א' (2024)

מסכם: רועי מעין

The Raymond and
Beverly Sackler Faculty
of Exact Sciences
Tel Aviv University

פרק 1 – אותות ומערכות

פרק 2 – אלגוריתמים ויישומים

1 – אותות ומערכות

אותות

מוטיבציה והגדרת אות

<u>מה זה בכלל DSP:</u>

- **Digital** (Signal Processing) – עיבוד אותות באמצעות מנגנונים ספרתיים – עיבוד ספרתי של אותות.
- (Digital Signal) **Processing** – עיבוד של אותות ספרתיים. הקורס מוקדש להגדרה זו. נדבר על אותות ספרתיים (digital signals) ואיך לעבד אותם. בניגוד לקורס של מהנדסים, רוב הקורס ידבר על אותות ספרתיים (לא אנלוגיים).

<u>מה זה אות/Signal:</u> אין דבר כזה "אות" כלשהו, יש אות אנלוגי (analog) ואות ספרתי (digital).

- <u>אות אנלוגי</u> – **פונקציה ממשית** $s(t)$ של משתנה יחיד רציף t שמייצג **זמן**. הפונקציה צריכה לעמוד במספר תנאים.
- <u>אות ספרתי/דיגיטלי</u> – **סדרה ממשית** s_n. בניגוד לפונקציה, יש לסדרה **מספר בן-מניה של ערכים שונים**. גם כאן הסדרה צריכה לעמוד במספר תנאים שגורמים לכך שיש קשר בין האות הספרתי לבין דברים שפוגשים בעולם האמיתי.

<u>מה זה לא Signal:</u>

- **פונקציה מרוכבת** היא אינה אות (בהמשך הקורס נשתמש, ונסביר למה מותר לנו וזה לא מפר משהו עמוק בתיאוריה, אלא עבור הפשטות המתמטית בחלק מסוים).
- **פונקציות במספר משתנים** כמו תמונות $I(x,y)$, וידאו $v(x,y,t)$, גלים $w(x,t)$ שהן פונקציה של מרחב וזמן, לא רק זמן.
- **אינפורמציה** אינה אות, יש תהליך שנקרא אפנון שמאפשר להעביר אינפורמציה על האות.

<u>למה דיגיטלי ולא אנלוגי:</u> תחום האלקטרוניקה מעבד אותות אנלוגיים, אך אנחנו רוצים לעבד אותות דיגיטליים באמצעות מחשב. למחשב ספרתי, הקלט והפלט לא יכולים להיות רציפים, אלא איברים שנוכל לשים בוקטור/מטריצה. יש דרך לקחת אות אנלוגי ולהפוך אותו לאות דיגיטלי עם אותה אינפורמציה. למה שנרצה לעשות את זה?

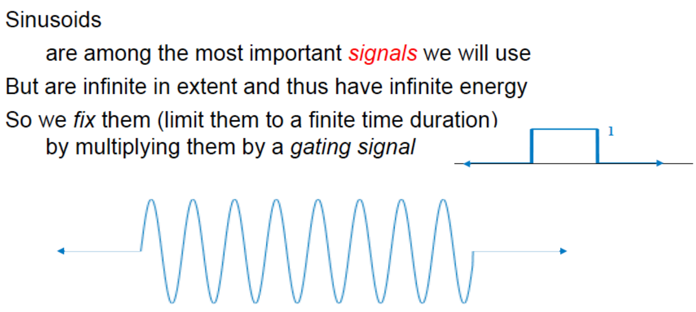
- גמישות – scalability, כאשר יש להרחיב/לעדכן את האלגוריתם ניתן לבצע זאת בקלות בתוכנה, לעומת מעגל אלקטרוני.
- דיוק – מחשב אנלוגי לעיתים לא מדייק בתוצאות החישוב, לשם כך נרצה לעבוד באופן דיגיטלי.
- יציבות – מחשב אנלוגי יכול לתת תוצאות שונות בזמנים שונים עבור אותו קלט, מחשב דיגיטלי לא יעשה זאת.
- מימושים מורכבים – יותר פשוט לממש במחשב דיגיטלי.

	אות אנלוגי	אות דיגיטלי
הצגה	$s(t) - continuous\ time$ $-\infty < t < +\infty$	$s_n - discrete\ time$ $n = -\infty, \dots, +\infty$
דרישות פיזיקליות	• ערכי s הם ממשיים • ערכי s מוגדרים לכל t • **אנרגיה סופית** – כמה "גדול" האות • **רוחב סרט סופי** – כמה "מהיר" האות	
שימוש מתמטי	• ערכי s יכולים להיות מרוכבים • אנרגיה אינסופית אפשרית • רוחב סרט אינסופי אפשרי	

<u>אנרגיה (energy):</u> נרצה לכמת את הגודל של האות. מקובל להגדיר באופן הבא:

- עבור אות אנלוגי: $E = \int_{-\infty}^{\infty} |s(t)|^2 dt$.
- עבור אות דיגיטלי: $E = \sum_{n=-\infty}^{\infty} |s_n|^2$.

לפי הגדרה זו האנרגיה היא אי-שלילית, וככל שהסיגנל יותר גדול האנרגיה יותר גדולה.

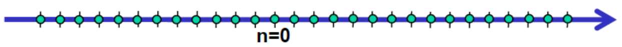
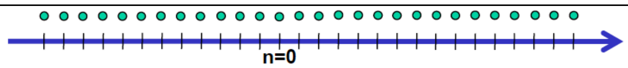
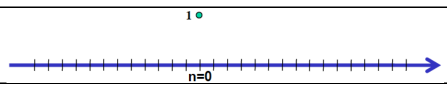
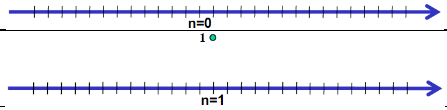
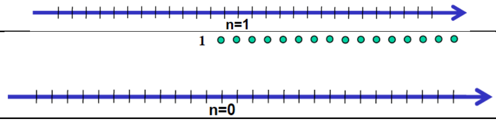


במקרה של sin/cos (שנעבוד איתם הרבה), יש להם אנרגיה אינסופית. לכן, נשתמש ב-**gating**: כל אות שהאנרגיה שלו אינסופית, נוכל להכפיל אותו באות gate (שהוא 0 או 1), וניצור אות שאינו ה-sin/cos המקורי, אלא שהוא מאופס בפרקן זמן מסוימים. כך נכשיר את הסיגנל.

נעבור על מספר דוגמאות ונבחן האם מדובר באות:

דוגמה	אות	ניתוח
1. גובה הר אוורסט	**אנלוגי** – בכל זמן הגובה מוגדר והוא מספר ממשי, אך לא בהכרח קבוע. נגדיר שלפני שההר נוצר הגובה הוא 0, וגם אחרי שהוא יהרס גובהו 0.	אנרגיה – משום שאיפסנו אותו בהתחלה ובסוף, יש להר אנרגיה סופית. רוחב סרט – גם אם הגובה משתנה עם הזמן, הוא לא משתנה מאוד מהר, ולכן גם רוחב הסרט הוא סופי.
2. $\left(e^{it}+e^{-it}\right)$	**אנלוגי** – נשים לב שערך הביטוי הוא $2\cos(t)$. לכן מדובר בפונקציה ממשית.	אנרגיה – אינסופית. רוחב הסרט – אפס, לא משתנה. אפשר לנבא אותו X זמן קדימה.
3. מחיר משולש פיצה	**אנלוגי** – זו פונקציה שתלויה בזמן. המחירים בדידים.	נגדיר היטב היכן אנחנו קונים משולש פיצה ומאיזה סוג בדיוק.
4. $sinc=\frac{\sin(t)}{t}$	**אנלוגי** – פונקציה ממשית מוגדרת בכל זמן.	מוגדר ב-0 להיות 1. אנרגיה – סופית. רוחב סרט – סופי.
5. $\phi(n)$ – מספר השלמים החיוביים הקטנים מ-n ללא מחלקים משותפים עם n		
6. רמת המים באסלה	**אנלוגי** – לרוב קבוע, אבל בזמנים רנדומלים יכול ליפול ל-0 מהר מאוד.	
7. $\lfloor t\rfloor$		
8. מיקום קצה הכנף של יתוש	**אנלוגי** – מיקום במרחב	אם נתמקד בציר אחד זה סיגנל כשר
9. $\sqrt{t}$	**אנלוגי** – פונקציה ממשית מוגדרת בכל זמן.	נדבר על שורש של מספר חיובי, כלומר עבור ערכים שליליים נקבע 0. אנרגיה – אינסופית, לא חסומה. גדלה לאט לכן לא תהיה בעיה בפועל.
10. ממוצע מדד ה-Dow Jones	**דיגיטלי** – מוגדר בסגירת הבורסה כל יום.	
11. $\sin\left(\frac{1}{t}\right)$	**לא**	לא מוגדר ב-0 $t=0$ וגם לא מתנהג יפה שם. יש שם תדירות אינסופית. אין שימושים פיזיקליים בעולם האמיתי של $\sin\left(\frac{1}{t}\right)$.
12. גודל טיפות המים מברז מטפטף	**דיגיטלי** – סדרת הטיפות היא גדלים קבועים שחוזרים על עצמם.	ניתן לתאר את זה על ידי "טיפה, טיפה, אין טיפה"
13. סדרת הערכים $x_n\in[0,1]$ המוגדרת: $x_{n+1}=\lambda x_n(1-x_n), 0\leq\lambda\leq 4$		

דוגמאות לאותות:

אות	נוסחה	המחשה
אות **האפס** (היחיד שיש לו אנרגיה אפס)	$s_n=0$	
אות **קבוע** – אנרגיה אינסופית, אבל נחסום ונכשיר אותו להיות סופי	$s_n=k$	
UI - אות מתקף היחידה / הלם (unit impulse)	$s_n=\delta_{n,0}$	
SUI - אות מתקף היחידה עם הזזה	$s_n=\delta_{n,m}$	
אות **המדרגה** – במובן מסוים האינטגרל של ה-UI	$s_n=\theta_n$	

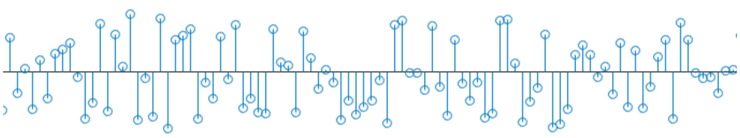
סוגי אותות ואופרטורים

אות כאובייקט: אות הוא יותר מאוסף של ערכים (ייצוג של האות), נתייחס לאות בתור **אובייקט** שניתן לבצע עליו פעולות:

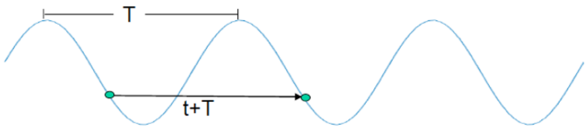
- הגברה (gain/attenuation): ניקח את האות x ונגביר אותו במספר g ונקבל את האות y: $y = gx$. זה אומר שלכל n מתקיים $y_n = gx_n$, ועבור המקרה האנלוגי לכל t מתקיים $y(t) = gx(t)$.
  - עבור $g = -1$ מדובר ב-inversion.
  - אם $|g| < 1$ מדובר בהפחתה.
- חיבור: עבור שני סיגנלים x, y נגדיר את החיבור $w = x + y$. במקרה הדיגיטלי $w_n = x_n + y_n$ ובמקרה האנלוגי $w(t) = x(t) + y(t)$.
- חיסור: $w = x - y = x + (-y)$.
- סכום משוקלל: עבור אותות x, y ומספרים a, b אפשר להגדיר $w = ax + by$.

סוגי אותות:

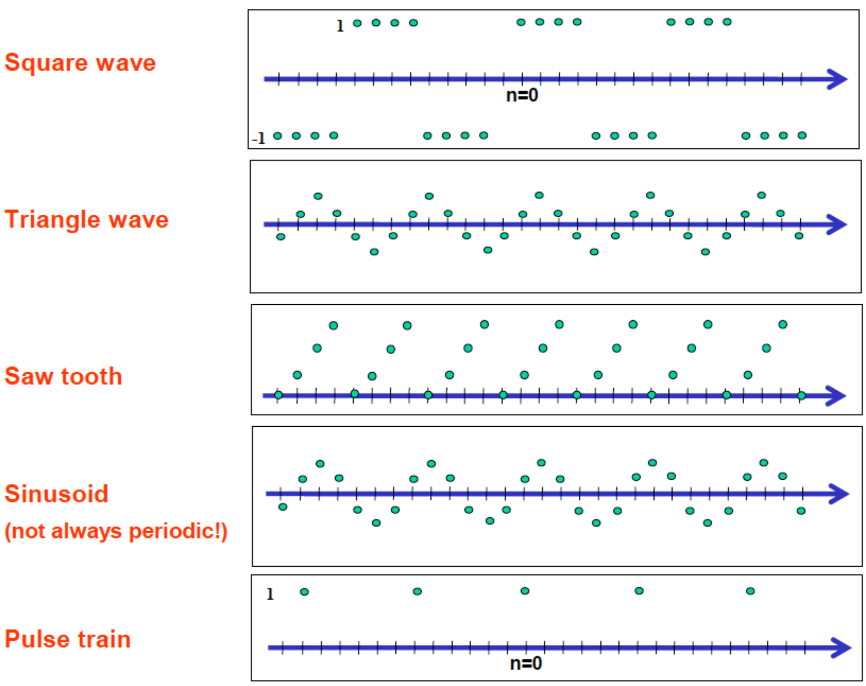
- אות **דטרמיניסטי** – יש אלגוריתם שמאפשר לנו לנבא את הערך של האות בכל זמן.
- אות **סטוכסטי (מאופנן)** – אין אלגוריתם שיכול לנבא במדויק את הערך של האות בכל זמן. האות הוא רנדומלי במובן מסוים. דוגמה מובהקת לאות סטוכסטי הוא רעש לבן. גם אם נסתכל על האות מזמן $-\infty$ עד n, לא נדע להגיד שום דבר על $n + 1$. יש גם סטוכסטי שאינו רעש לבן.



- אות **מחזורי** – אות שחוזר על עצמו אחרי פרק זמן מסוים.
  - עבור אות אנלוגי $p(t)$, אם קיים $T > 0$ כך שלכל t מתקיים $p(t + T) = p(t)$. ה-T הקטן ביותר שמקיים את זה נקרא ה**מחזור**.
  - עבור אות דיגיטלי p_n באותו אופן קיים $N > 0$ כך שלכל n מתקיים $p_{n+N} = p_n$.
  - **אות מחזורי חייב להיות דטרמיניסטי** - ברגע שיודעים מחזור אחד שלו, אפשר לנבא אותו לכל הזמנים. הוא לא יכול להיות סטוכסטי.
  - האות הדיגיטלי $s_n = A\sin(\omega n)$ הוא תמיד מחזורי למרות שהמחזור שלו יכול להיות שונה מהמחזור של הסינוס האנלוגי המקורי.



דוגמאות לאותות מחזוריים:



אופרטור $\hat{z}$ (הגברה): לאותות **דיגיטליים** אנחנו מגדירים:

- אופרטור advance (קידום בזמן): $y = \hat{z}x$ משמע $y_n = x_{n+1}$. זה אופרטור לא סיבתי (noncasual), רק לדטרמיניסטיים.
- אופרטור delay (השהייה בזמן): $y = \hat{z}^{-1}x$ משמע $y_n = x_{n-1}$. זה אופרטור סיבתי (casual), אפשר לממש אותו לכל אות. בסה"כ צריך נקודת זיכרון אחת. נשים לב שאפשר לקבל את אופרטור היחידה באמצעות $\hat{z}\hat{z}^{-1} = \hat{1}$.
- **האופרטורים האלה רלוונטיים רק לאותות דיגיטליים**. לגבי אותות אנלוגיים אין יחידת זמן אחת שניתן לעבוד איתה.

<u>אופרטור $\hat{\Delta}$ (הפרש סופי):</u>

- ההפרש הסופי הראשון. $y = \hat{\Delta}x$ משמע $y_n = x_n - x_{n-1}$. נשים לב כי $\hat{\boldsymbol{\Delta}} = \mathbf{1} - \hat{\mathbf{z}}^{-1}$.
- ההפרש הסופי השני הוא $\hat{\Delta}^2$. יש הפרשים מסדר גבוה יותר $y = \hat{\Delta}^m x$.
- אם יש אות שהוא פולינומיאלי ב-n, אז יש חזקה m כלשהי שעבורה $\hat{\Delta}^m x = 0$.

n		-2	-1	0	1	2	…
x	…	x_{-2}	x_{-1}	x_0	x_1	x_2	…
$\hat{\Delta}$ x	…	$x_{-2} - x_{-3}$	$x_{-1} - x_{-2}$	$x_0 - x_{-1}$	$x_1 - x_0$	$x_2 - x_1$	…
$\hat{\Delta}^2$x	…	…	…	$(x_0 - x_{-1}) - (x_{-1} - x_{-2}) = x_0 - 2x_{-1} + x_{-2}$	…	…	…

<u>אופרטור $\hat{\gamma}$ (סוכם):</u>

- סוכם (accumulator). $y = \hat{\gamma}x$ משמע $y_n = \sum_{m=-\infty}^{n} x_m = y_{n-1} + x_n$.
- הסוכם הוא ההופכי של ההפרש הסופי: $\hat{\gamma}\hat{\Delta} = \hat{\Delta}\hat{\gamma} = \hat{1}$.

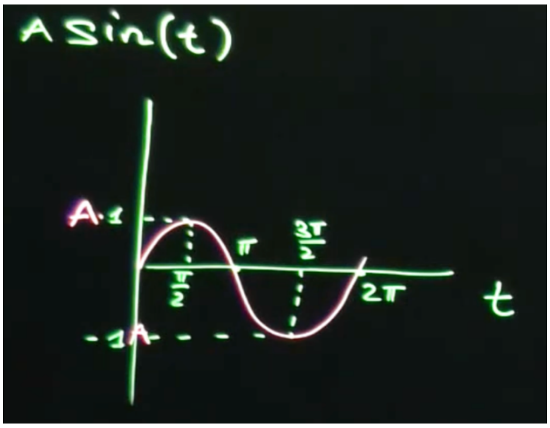
<u>אופרטור Rev (היפוך זמן):</u>

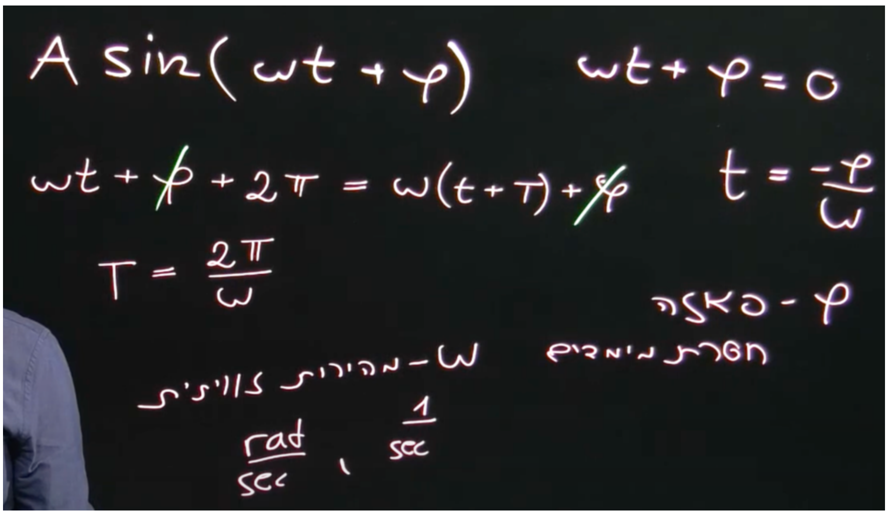
- $y = Rev(x)$ משמע $y_n = x_{-n}$.
- במקום שהזמן ילך מעכשיו עד אינסוף, הוא ילך אחורה מעכשיו עד מינוס אינסוף.
- אפשר לממש אופרטור כזה בסיגנלים **דטרמיניסטיים**. אם זה סיגנל סטוכסטי, גם אם אנחנו יודעים כמה הוא היה, לא נוכל לדעת כמה הוא יהיה.

<u>פעולות נוספות:</u>

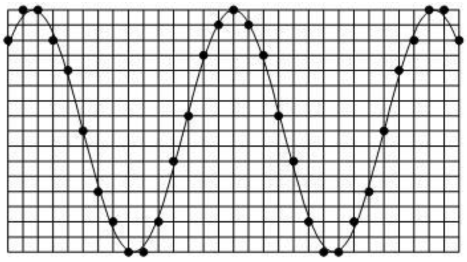
- השוואה בין סיגנלים (דמיון): $C_{xy}(m) = \sum_{n=-\infty}^{\infty} x_n y_{n-m}$.
- התמרת הילברט $\hat{H}$.

<u>רענון בנוגע לפונקציית סינוס – תיאור כללי של פונקציות מחזוריות:</u>





אותות כמרחב וקטורי



דגימה (בהמשך נגדיר באופן מדויק): הרבה פעמים אנחנו מקבלים אותות ספרתיים מתוך אותות אנלוגיים. זאת משום שבעולם האמיתי, האותות הם אנלוגיים. נרצה לקחת אות אנלוגי ולעבד אותו (בעזרת מחשב). לכן, נצטרך להפוך אותו לאות ספרתי. לתהליך הזה קוראים **דגימה (sampling)** ולפעמים (analog to digital) **a2d**. יש תנאים שבהם הדגימה לא מאבדת מידע – אפשר לדגום ולמרות שהורדנו ממספר איברים של $\aleph_1$ למספר של $\aleph_0$, עדיין אנחנו תופסים את כל מה שיש באות.

אותות ספרתיים דומים לוקטורים בהרבה מובנים, ועומדים בתנאים של מרחב וקטורי:

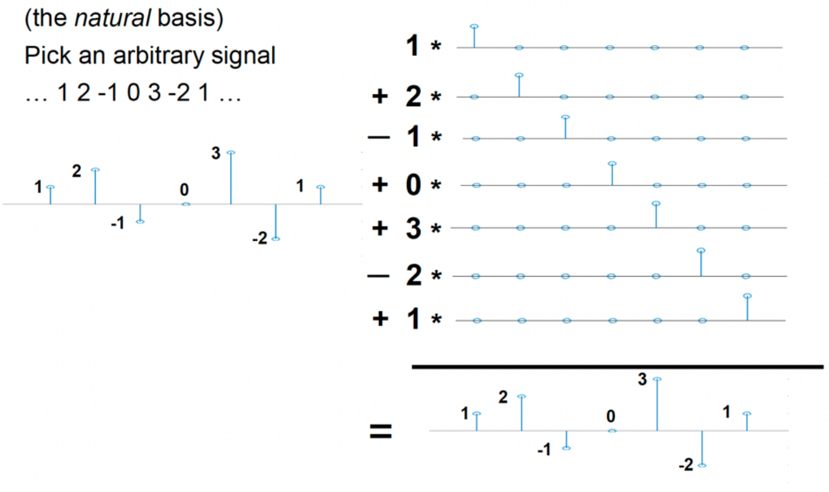
- אות האפס: $0_n = 0$ לכל n. זה דומה לכך שבמרחב וקטורי יש וקטור מיוחד שנקרא וקטור האפס.
- חיבור אותות: שני אותות יכולים לעבור חיבור וליצור אות חדש $z = x + y$. זה קורה גם בוקטורים.
- הגברת אות: כל אות ניתן להכפיל במספר ממשי (להגביר), כמו וקטור שאנחנו מכפילים בסקלר.
- היפוך אות: לכל אות s הגדרנו אות הופכי $-s$ כך שמתקיים $s + (-s) = 0$. כך גם בוקטורים.
- גודל אות: לכל אות הגדרנו מספר ממשי שמודד את הגודל שלו – אנרגיה.

אוסף כל האותות הספרתיים, מהווים מרחב וקטורי ליניארי (linear vector space). אמנם, אותות הם לא רק וקטורים, יש להם גם תכונה של זמן. סדר הרכיבים הוא לא שרירותי, כיוון שהזמן מתקדם רק בכיוון אחד מסוים. בליניארית למדנו שלכל מרחב וקטורי יש בסיס (לפחות אחד). **לנו יהיו חשובים 2 בסיסים במרחב האותות: הבסיס שמשרה את הזמן, הבסיס שמשרה את התדר.**

בסיס (תזכורת): קבוצה של וקטורים $b_1, \dots, b_d$ שעונה על שני תנאים:

1. הקבוצה פורשת את המרחב –אפשר לכתוב כל וקטור במרחב בתור צירוף ליניארי של וקטורי הבסיס. כלומר לכל וקטור x נוכל לכתוב: $x = a_1 b_2 + \dots + a_d b_d$ כאשר $a_1, \dots, a_d$ הם מקדמים.
2. הקבוצה היא בת"ל – אם יש צ"ל שלהם שנותן אפס, בהכרח כל ההגברים אינם אפס. דרך נוספת, אם אנחנו מוצאים צ"ל מסוים של אות באמצעות איברי הבסיס, הוא ייחודי, אין שתי דרכים שונות לייצג אותו.

בסיס SUI: ראינו את אות מתקף היחידה, ומתקף היחידה עם הזזה: $s_n = \delta_{n,m}$. האותות הוא תמיד אפס פרט לנקודה $n = m$ שבו הוא מקבל ערך 1. **אוסף כל אותות ה-SUI מהווים מרחב לבסיס כל האותות הספרתיים** – זה הבסיס הטבעי.



ניקח לדוגמה אות כלשהו $x = x_1, \dots, x_n$, ונכתוב אותו באמצעות איברי הבסיס SUI. ניקח את האיבר b_i ונכפיל אותו ב-m עבור $x_i = m$. כך נקבל $x_i = kb_i$.

יש גם SUI לגבי אותות אנלוגיים.

מימד (תזכורת): מספר איברי הבסיס. המימד של מרחב כל האותות הספרתיים הוא $\aleph_0$, ושל מרחב האותות האנלוגיים הוא $\aleph_1$.

נרצה לעבוד עם עוד בסיס חשוב שעוסק בתדר. בעבודה עם אותות ספרתיים נעבור בין ציר הזמן לציר התדר לא מעט, ולכן נזדקק לטרנספורמציה שמעבירה אותנו ביניהם – זו תהיה התמרת פורייה שנראה בהמשך (FFT, DFT).

ציר התדר

סדרת פורייה (Fourier Series):

בהדגמה ראינו בעצם שהרבה אותות אנלוגיים יכולים להיכתב כסכום של HRS (Harmonically Related Sinusoids): סינוסים שהמחזור שלהם הוא $\frac{1}{n}T$ כאשר T הוא המחזור של הפונקציה sin המקורית.

לגבי אות מחזורי אי-זוגי, ניתן לפתח עם סינוסים שהם HRS: אם המחזור הוא T, התדר $f = \frac{1}{T}$ והתדירות הזוויתית $\omega = 2\pi f = \frac{2\pi}{T}$, אז: $s(t) = a_1 \sin(\omega t) + a_2 \sin(2\omega t) + \cdots$.
באופן דומה, לגבי אות מחזורי זוגי, אפשר לעשות את זה עם קוסינוסים (cos): $s(t) = b_0 + b_1 \cos(\omega t) + b_2 \cos(2\omega t) + \cdots$.

<u>טענה כללית:</u> כל אות אנלוגי מחזורי כללי, אפשר לכתוב בתור סכום של אות זוגי ועוד אות אי-זוגי.
$s(t) = e(t) + o(t)$ כאשר $e(t) = \frac{s(t)+s(-t)}{2}$, $o(t) = \frac{s(t)-s(-t)}{2}$.

<u>משפט פורייה:</u> כל **אות אנלוגי מחזורי**, אפשר לכתוב באופן הבא: $s(t) = a_1 \sin(\omega t) + \cdots b_0 + b_1 \cos(\omega t) + \cdots$. **יש כאן $\aleph_0$ איברי בסיס (בן-מניה)**. אמרנו שכל האותות האנלוגיים הוא ממימד $\aleph_1$, איך זה הגיוני? מדובר בתת-מרחב של מרחב האותות האנלוגיים.

אם כן, מצאנו כרגע בנוסף ל-SUI כי: **אוסף כל האותות האנלוגיים המחזורים במחזור מסוים T, מהווים תת-מרחב וקטורי** שיש לו בסיס של סינוסים וקוסינוסים מופעלים על מכפלה מהצורה $n\omega t$. אנחנו שואפים לבסיס למרחב כל האותות הספרתיים בהמשך.

<u>הערות:</u>

- לגראנז' טען שלא נוכל לקבל מסכום של פונקציות רציפות (סינוסים), פונקציה אי-רציפה (כמו גל המשולש). זה אכן נכון עבור סכום **סופי** של פונקציות רציפות/חלקות (נגזרת רציפה), אבל **סכום אינסופי יכול להיות עם תוצאה אי-רציפה**.
- למה לא נוכל להשתמש בפיתוח של פולינומים (טיילור), אלא רק עם סינוסים וקוסינוסים? כל פונקציה רציפה וגזירה אינסוף פעמים, אם נסתכל על אזור מספיק קטן הוא נראה כמו קו ישר. אם נרצה לדעת אזור קצת יותר גדול, זה כבר לא קו ישר. אנחנו מפתחים **מסביב לנקודה מסוימת**. ב-DSP אנחנו מתעניינים **באות לכל אורך פרק הזמן שלו** (ברמת האינסוף), אין זמן מסוים שמעניין אותנו. טור פורייה מתכנס בצורה אחרת, לא מסביב לנקודה, אלא בכל שאנחנו מוסיפים עוד סינוסים וקוסינוסים הוא **נראה יותר קרוב לאות המקורי באותה מידה בכל מקום**. **השגיאה קטנה** בכל הזמנים באותה מידה.
- נשתמש במושג **ספקטרום** בהקשר של אות, ונתכוון לייצוג שלו בבסיס החדש שאנחנו הולכים לבנות – **בסיס התדר**.

<u>נפריד בין זמן ותדר:</u> יש לנו את בסיס ה-SUI שקראנו לו ציר הזמן. אנחנו רוצים למצוא את ציר התדר.
קונבנציה בקורס: **אות קטנה תשמש אותנו לייצוג בציר הזמן - $s(t), s_n$. נכתוב באות גדולה לייצוג בציר התדר - $S(\omega), S_k$**.
זו אותה אות (letter) כי מדובר באותו אות (signal), רק ייצוגים שונים שלו. כדי לעבור בין הציר הזמן לתדר:

- אותות אנלוגיים – FT.
- אותות ספרתיים – DFT.

Time domain (axis)

s(t) s_n

Basis - **S**hifted **U**nit **I**mpulses

Frequency domain (axis)

S(ω) S_k

Basis - **sinusoids**

We use the same letter *capitalized* to stress that these are the same signal, just different representations

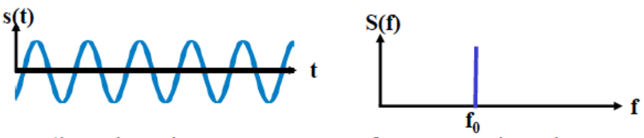
To go between the representations :
analog signals - Fourier Transform FT/iFT
digital signals - Discrete Fourier Transform DFT/iDFT

There is a *fast* algorithm for the DFT/iDFT called the FFT

<u>ציר התדר:</u>

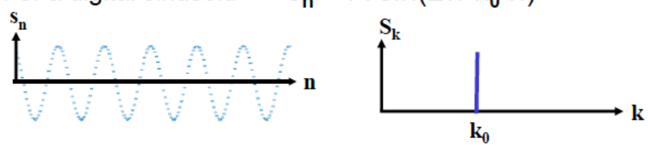
נסתכל תחילה על אות סינוס אנלוגי: $s(t) = A\sin(\omega_0 t)$. **יש לו תדר אחד מוגדר היטב – כמו מתקף.** אותו דבר בסינוס ספרתי: $s_n = A\sin(\omega_0 n)$, יהיה לו אינדקס אחד בציר התדר שלו. בתרשים מדובר על ייצוג בתדר רגיל, ניתן גם לרשום באמצעות התדירות הזוויתית.

For an analog sinusoid s(t) = A sin(2π f_0 t)



time domain frequency domain

For a digital sinusoid s_n = A sin(2π k_0 n)

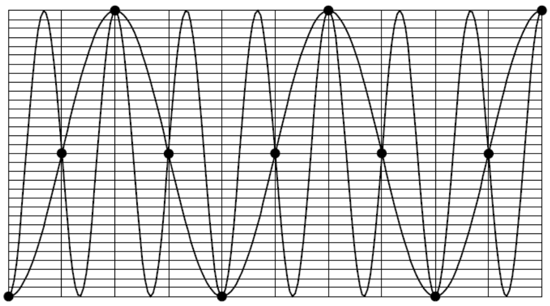


time domain frequency domain

כעת נסתכל על סינוס ועוד קבוע: $s(t) = 1 + \sin(\omega_0 t)$. יש עוד קו בתדר 0. יש לאות הזה שני תדרים. נקרא לתדר הקבוע כאשר $\omega = 0$ **רכיב ה-DC**.

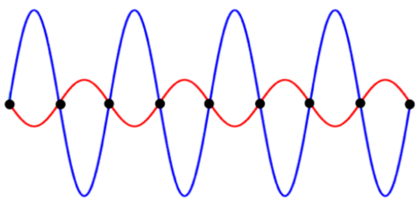
- <u>בציר הזמן</u> - אם נעשה **ממוצע** של האות על זמן מספיק ארוך, בדיוק מה שיש באמצע של הסינוס. לכן כדי למצוא את רכיב ה-DC של אות צריך למצע על $s(t), s_n$.
- <u>בציר התדר</u> – ניקח את **הרכיב הראשון בתדר 0**: $S(0), S_0$.

<u>רוחב סרט (bandwidth):</u> דיברנו על כך שאות צריך לקיים רוחב סרט סופי. נגדיר רוחב סרט כהפרש בין התדר הגבוה ביותר לתדר הנמוך ביותר בציר התדר: $BW = f_{max} - f_{min}$. רוחב סרט סופי, זה אומר שהתדר הגבוה ביותר בייצוג **בציר התדר** הוא סופי. אם מסתכלים בציר התדר, יש לנו תדר גבוה ביותר שהוא לא אפס, ואחרי זה הוא אפס – כלומר הוא מוגדר רק בתחום מסוים בציר התדר.

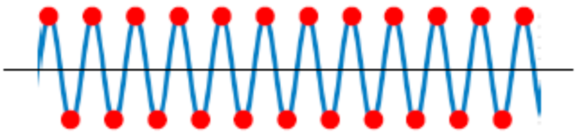


משפט הדגימה של Nyquist: אם דוגמים אות אנלוגי חוקי (עם רוחב סרט סופי) ואין לו תדרים בספקטרום מעל $f_{Nyquist}$, אז אם **דוגמים בקצב יותר מפי 2 בתדר הגבוה ביותר**: $2f_{Nyquist}+\varepsilon$, אז אין איבוד מידע. כלומר, ניתן לחזור אחורה מתוך האות הספרתי לאות האנלוגי המקורי.
בדיוק פי 2 לא תמיד עובד, לכן צריך לדגום בקצב קצת יותר מפי 2.

אם נותנים לנו f_s של דגימה מסוימת. ניתן לומר מה התדר הגבוה ביותר שמותר שיופיע באות, והוא $f_{Nyquist}=\frac{f_s}{2}$. אם דוגמים בקצב נמוך מידי, מבצעים **aliasing**. ברגע שעוברים את $f_{Nyquist}$ כבר לא עומדים בתנאים של משפט הדגימה. אנחנו צריכים לוודא שאין רכיבים בתדר מעל $f_{Nyquist}$, אחרת יהיו עיוותים. ניתן לבצע את זה באמצעות anti-aliasing filter.



יש לנו אות אנלוגי $s(t)$. כאשר אנחנו דוגמים אותו, אנחנו מסתכלים על $t=nt_s$. נניח כאן $t_s=1ms$, אז מסתכלים על $0,1ms,2ms,\dots$ ואז נותנן לנו s_n. מתקיים $\boldsymbol{n}=\frac{\boldsymbol{t}}{\boldsymbol{t_s}}$ ויחידות השנייה מצטמצמות. אם עובדים ביחידות של t_s אז התדר הגבוה ביותר הוא f_N, פי 2 ממנו זה הדגימה f_s.



- התדר **הספרתי** יוגדר $\boldsymbol{k}=\frac{\boldsymbol{f}}{\boldsymbol{f_s}}$ וגם הוא יהיה ללא יחידות כי ההרץ מצטמצם. נקבל כי k **מקסימלי** הוא $k=\frac{f_N}{f_s}=\frac{0.5f_s}{f_s}=\frac{\mathbf{1}}{\mathbf{2}}$.
- התדר **הזוויתי** המקסימלי: $\omega_{max}=2\pi f_{max}=\boldsymbol{\pi}$.

איך נראה אות Nyquist? זה אות שנדגום אותו בדיוק פעמיים ב-1 ו-1.
יש נוסחה לשחזור הסיגנל האנלוגי – לא ניכנס לזה.

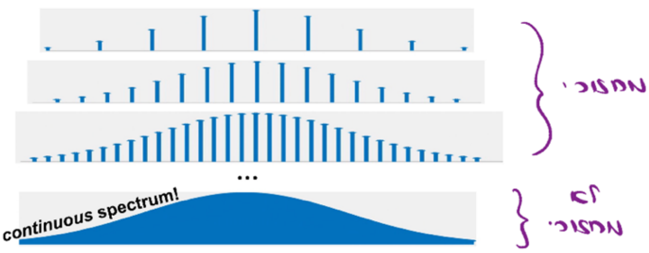
התמרות

נרצה לטעון שטור פורייה הוא קצת "מכוער". כאשר מדברים על מרחבים וקטוריים, אנחנו אוהבים בסיסיים שבהם כל האיברים הם מאותו סוג. כאן יש לנו איברי בסיס מסוג $\sin(k\omega t)$ וחלק מסוג $\cos(k\omega t)$. האם אפשר לבנות בסיס יפה יותר עם סוג אחד? ניעזר בזהות הבאה: $\sin(k\omega t)=\frac{1}{2i}\left(e^{ik\omega t}-e^{-ik\omega t}\right)$ וגם $\cos(k\omega t)=\frac{1}{2}\left(e^{ik\omega t}+e^{-ik\omega t}\right)$. במקום לפתח עם sin, cos נעשה פיתוח של סדרת פורייה במונחים של e: $\boldsymbol{s(t)=\sum_{k=-\infty}^{\infty}S_k e^{ik\omega t}}$.

יש כמה בעיות, אך אנחנו מוצאים להן פתרון:

1. הפונקציה $e^{ik\omega t}$ היא מרוכבת, והגדרנו שאות לא יכול להיות כזה! ניקח בסוף החישוב את **החלק הממשי** $\boldsymbol{Re(e^{ik\omega t})}$.
2. התדרים $k\omega$ יכולים להיות שליליים. מה זה אומר לבצע -2 מחזורים בשנייה? **התדרים אותו הדבר**.

בהדגמה ראינו כיצד תדירות חיובית $e^{\omega t}$ (ω חיובית) מניבה לנו את פונקציית sin. בתדירות שלילית $e^{-\omega t}$, נקבל גם sin. מה ההבדל בין תדירות חיובית ותדירות שלילית? מהצד זה נראה אותו דבר. בסוף החישוב עם הפונקציה המרוכבת, ניקח את החלק הממשי, כי אין הבדל בפועל בין החיובי לשלילי. זה מפשט את החישוב מתמטית מאוד.



איך נהפוך אות מחזורי ללא-מחזורי? נשאיף את זמן המחזור T לאינסוף, דהיינו האות אינו מחזורי. בהתאמה, התדירות שואפת ל-0. טור מבצע סכום ומחזיר לנו ערך בדיד. הפעולה שאנו מבצעים נקראת התמרה (transform), המרווח בין התדרים מצטמצם ואנו מקבלים ערכים **רציפים**: אין תדר בסיסי שכל התדירויות הן כפולות שלו.

התמרת פורייה (FT): כשיש לנו מספר לא בן-מניה בסכום, אנחנו מקבלים **אינטגרל**. באותות **אנלוגיים**:

מעביר אותנו מציר הזמן לציר התדר:

$$S(w)=FT\big(s(t)\big)=\int_{t=-\infty}^{\infty}s(t)\,e^{-iwt}dt$$

בכיוון ההפוך – מציר התדר לציר הזמן:

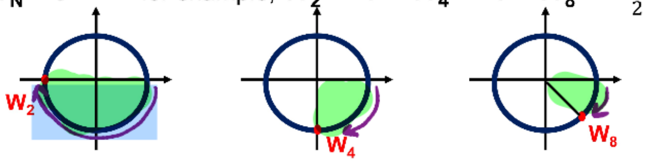
$$s(t)=FT^{-1}\big(S(w)\big)=\frac{1}{2\pi}\int_{w=-\infty}^{\infty}S(w)\,e^{iwt}dw$$

בהינתן אות **ספרתי כללי** שאינו בהכרח מחזורי: נתחיל עם FT ונדגום את האות בציר הזמן, ואת האות בציר התדר, ונקבל **DFT** (discrete). האינטגרל הופך להיות סכום. **נסתכל מזמן 0 עד זמן N כלשהו, כלומר N דגימות**:

$$S_K = DFT(s_n) = \sum_{n=0}^{N-1} e^{-i\frac{2\pi nk}{N}} s_n \qquad s_n = DFT^{-1}(S_k) = \frac{1}{N}\sum_{k=0}^{N-1} e^{i\frac{2\pi nk}{N}} S_k$$

<u>המטריצה W:</u> על מנת לעבור מהבסיס של SUI לבסיס במרוכבים, נשתמש במטריצה שמייצגת את ההעתקה, המטריצה W. היא אוניטרית (מרוכבים). אם נצייר מעגל יחידה, **לכל W_N מתאים ערך – שורש היחידה ממעלה N**. חותכים את מעגל היחידה ל-N חלקים, החלק הראשון הוא שורש היחידה ממעלה N. מתקיים $W_N^N = 1$.

$W_N = e^{-i\frac{2\pi}{N}}$ for example, $W_2 = -1$ $\quad W_4 = -i \quad W_8 = -\frac{\sqrt{2}}{2}(1+i)$



האינדקס הראשון במטריצה הוא n והשני הוא k. בדוגמה עבור $n = 2$ כל מה שהוא עם $n = 0, k = 0$ נקבל 1. רק במקרה של $W_2^1 = -1$.

נשים לב לטעות ב-W_8 זה $(1 - i)$.

נזכיר:

$$S_k = \sum_{n=0}^{N-1} e^{-i\frac{2\pi nk}{N}} s_n \quad k = 0 \ldots N-1 \qquad \longleftrightarrow \qquad S_k = \sum_{n=0}^{N-1} W_N^{nk} s_n \quad k = 0 \ldots N-1$$

$$s_n = \frac{1}{N}\sum_{k=0}^{N-1} e^{i\frac{2\pi nk}{N}} S_k \quad n = 0 \ldots N-1 \qquad \qquad s_n = \frac{1}{N}\sum_{k=0}^{N-1} W_N^{-nk} S_k \quad n = 0 \ldots N-1$$

באופן כללי – בשורה הראשונה נקבל תמיד 1, וכך גם בעמודה הראשונה.

- רכיב ה-**DC** הוא סכום כל הערכים בשורה הראשונה.
- תדר **Nyquist** הוא סכום הערכים בשורה האחרונה.

דוגמא עבור N=2:

$$\begin{pmatrix} S_0 \\ S_1 \\ \vdots \\ S_{N-1} \end{pmatrix} = \begin{pmatrix} W_{0,0} & W_{0,1} & \cdots & W_{0,N-1} \\ W_{1,0} & W_{1,1} & \cdots & W_{1,N-1} \\ \cdots & \cdots & \cdots & \cdots \\ W_{N-1,0} & W_{N-1,1} & \cdots & W_{N-1,N-1} \end{pmatrix} \begin{pmatrix} s_0 \\ s_1 \\ \vdots \\ s_{N-1} \end{pmatrix}$$

For N=2 we have 2 signal values in the time domain s_0 s_1

$W_2 = e^{-i\frac{2\pi}{2}} = e^{-i\pi} = -1$

and in the frequency domain $\quad S_0 = s_0 + s_1 \quad S_1 = s_0 - s_1$

$$\begin{pmatrix} S_0 \\ S_1 \end{pmatrix} = \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \begin{pmatrix} s_0 \\ s_1 \end{pmatrix} \qquad \begin{pmatrix} s_0 \\ s_1 \end{pmatrix} = \frac{1}{2} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \begin{pmatrix} S_0 \\ S_1 \end{pmatrix}$$

$$\begin{array}{c|c} & \begin{array}{cccccc} k=0 & 1 & 2 & 3 & & n-1 \end{array} \\ \begin{array}{c} n=0 \\ 1 \\ 2 \\ 3 \\ \\ \\ N-1 \end{array} & \begin{pmatrix} 1 & 1 & 1 & 1 & \cdots & 1 \\ 1 & W_N & W_N^2 & W_N^3 & \cdots & W_N^{N-1} \\ 1 & W_N^2 & W_N^4 & W_N^6 & \cdots & W_N^{2(N-1)} \\ 1 & W_N^3 & W_N^6 & W_N^9 & \cdots & W_N^{3(N-1)} \\ 1 & -1 & 1 & -1 & \cdots & \\ 1 & \cdots & \cdots & \cdots & \cdots & \cdots \\ 1 & W_N^{-1} & W_N^{-2} & W_N^{-3} & \cdots & W_N^{1-N} \end{pmatrix} \end{array}$$

דוגמה עבור $n = 4$:

Let's try another one

$$\begin{pmatrix} S_0 \\ S_1 \\ \vdots \\ S_{N-1} \end{pmatrix} = \begin{pmatrix} W_{0,0} & W_{0,1} & \cdots & W_{0,N-1} \\ W_{1,0} & W_{1,1} & \cdots & W_{1,N-1} \\ \cdots & \cdots & \cdots & \cdots \\ W_{N-1,0} & W_{N-1,1} & \cdots & W_{N-1,N-1} \end{pmatrix} \begin{pmatrix} s_0 \\ s_1 \\ \vdots \\ s_{N-1} \end{pmatrix}$$

For N=4 we have 4 signal values in the time domain s_0 s_1 s_2 s_3

$W_4 = e^{i\frac{2\pi}{4}} = e^{i\pi/4} = -i$

$$\underline{\underline{W_4}} = \begin{pmatrix} W_4^0 & W_4^0 & W_4^0 & W_4^0 \\ W_4^0 & W_4^1 & W_4^2 & W_4^3 \\ W_4^0 & W_4^2 & W_4^4 & W_4^6 \\ W_4^0 & W_4^3 & W_4^6 & W_4^9 \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & W_4 & W_4^2 & W_4^3 \\ 1 & W_4^2 & W_4^0 & W_4^2 \\ 1 & W_4^3 & W_4^2 & W_4^1 \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & -i & -1 & i \\ 1 & -1 & 1 & -1 \\ 1 & i & -1 & -i \end{pmatrix}$$

נשים לב שרק לסינוס יש תדר אחד מוגדר. לכל סיגנל אחר יש ספקטרום של תדרים, בין אם מחזורי או לא.

Frequency is well defined for sinusoids

$s(t) = A\sin(\omega t + \phi)$ $\quad s_n = A\sin(\omega n + \phi)$

A amplitude
ω (angular) frequency
Φ phase

אפנון: נוכל לגרום לסיגנל לשאת אינפורמציה ממקום למקום, נשנה מקומית את התדר של הסיגנל.

- אפנון AM (אמפליטודה) – משנים אמפליטודה כדי להעביר אינפורמציה.
- אפמנון FM (תדר) – נשנה את התדר של הסיגנל כדי שהוא יעביר אינפורמציה.

AM: $s(t) = A(t)\sin(\omega t + \phi)$

FM: $s(t) = A\sin(\omega(t)t + \phi)$

AM+FM: $s(t) = A(t)\sin(\omega(t)t + \phi)$

נעשה תמיד phase modulation כי זה תמיד גורם גם ל-frequency modulation.

התמרת הילברט (Hilbert Transform): נניח שיש לנו סיגנל שממלא שני תנאים:

1. נטול רכיב DC (רכיב ה-DC בטרנספורמציית פורייה של הסיגנל הוא 0, כלומר הממוצע הוא 0).
2. רוחב סרט סופי (לא חייב אנרגיה סופית).

נוכל להגדיר לו **תדר תלוי זמן**. ההתמרה מקבלת $\cos(\phi(t))$ ומחזירה $\sin(\phi(t))$. בהינתן סיגנל $x(t)$ נקבל את $y(t)$. מתוך כל האופציות לבחור $A(t), \phi(t)$, HT מביא לנו את ה"חלקים" ביותר.

The Hilbert transform is a 90 degree *phase shifter*

$\hat{H}\cos(\Phi(t)) = \sin(\Phi(t))$

Hence

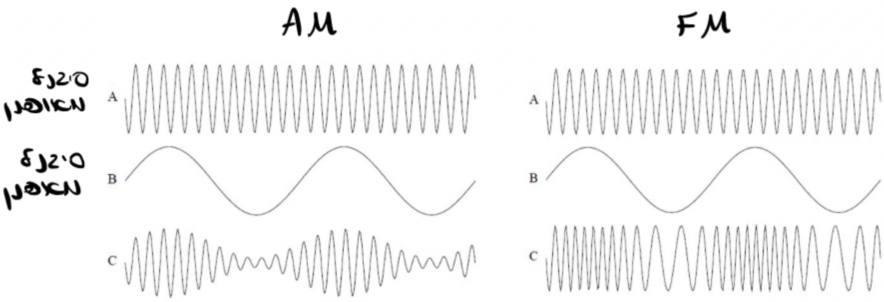
- $x(t) = A(t)\cos(\Phi(t))$
- $y(t) = \hat{H}x(t) = A(t)\sin(\Phi(t)) \to$ A(t) יצא כקבוע
- $A(t) = \sqrt{x^2(t) + y^2(t)}$
- $\Phi(t) = \arctan_4\left(y(t)/x(t)\right)$

כרגע שתלוי רגע משתנה, אם הפאזה משתנה ולהיפך

this is equivalent to shifting every spectral component separately by 90°

The instantaneous frequency is the derivative of the instantaneous frequency

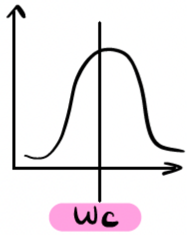
דוגמא



The instantaneous (analytical) representation

- $x(t) = A(t)\cos(\Phi(t)) = A(t)\cos(\omega_c t + \phi(t))$
- A(t) is the *instantaneous amplitude*
- $\phi(t)$ is the *instantaneous phase*

ω_c center frequency
carrier frequency



This is used in information transmission

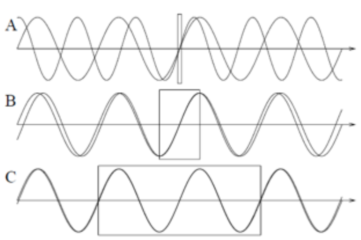
- **A**mplitude **M**odulation $\quad x(t) = A(t)\cos(\omega_c t)$
- **P**hase **M**odulation $\quad x(t) = A\cos(\omega_c t + \phi(t))$
- **F**requency **M**odulation $\quad x(t) \neq A\cos(\omega(t)\, t)$

Why not?

Frequency is the derivative of phase

- $\Phi(t) = \omega_0 t$ then $\omega(t) = \omega_0$
- $\Phi(t) = \omega_c t + \phi(t)$ then $\omega(t) = \omega_c + \frac{d}{dt}\phi(t)$

עקרון אי הוודאות (uncertainty theorem): ככל שמסתכלים על סיגנל בטווח זמן Δt יותר גדול, כאשר מנסים לשער מה התדר שלו, הדיוק שלנו יותר גדול והשגיאה בתדירות Δw יותר קטנה ולהיפך. כלומר: $\Delta w \cdot \Delta t > constant$.



התמרת Z

התמרה זו מוגדרת רק ב-DSP ולא בסיגנלים אנלוגיים. זה לא באמת טרנספורמציה.

פונקציה יוצרת: בהינתן סדרה s_n, נגדיר את הפונקציה היוצרת להיות טור החזקות הבא: $s(x)=\sum_{n=0}^{\infty} s_n x^n$. נראה דוגמה לטריקים עם פונקציות יוצרות שישמשו אותנו עבור התמרת Z. זה כלי שמאפשר לנו לנתח סדרה באמצעות מעבר לפונקציה.

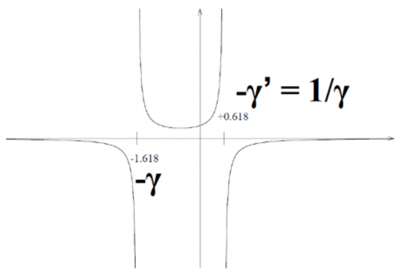
נדגים על סדרת פיבונאצ'י: $f_0=f_1=1, f_n=f_{n-1}+f_{n-2}$. הפונקציה היוצרת של הסדרה היא:

$$f(x)=\sum_{n=0}^{\infty} f_n x^n=1+x+2x^2+3x^3+5x^4+8x^5+\cdots$$

נסכום מ-2 כי חסרים שני איברים בטור: $\sum_{n=2}^{\infty} f_n x^n=\sum_{n=2}^{\infty} f_{n-1} x^n+\sum_{n=2}^{\infty} f_{n-2} x^n$. כעת נשים לב:

- $\sum_{n=2}^{\infty} f_n x^n=f(x)-1-x$ כי חסרים כאן שני האיברים הראשונים $1, x$.
- $\sum_{n=2}^{\infty} f_{n-1} x^n=x\cdot\sum_{n=2}^{\infty} f_{n-1} x^{n-1} \underset{\substack{m=n-1\\ n=m+1}}{=} x\cdot\sum_{m=1}^{\infty} f_m x^m=x\cdot\sum_{n=1}^{\infty} f_n x^n=x\cdot(f(x)-1)$. קיבלנו בסוף כי חסר האיבר הראשון (1).
- באופן דומה נקבל $\sum_{n=2}^{\infty} f_{n-2} x^n=x^2(f(x))$ כי לא חסרים אף איברים מהטור.

Using the techniques learned in elementary calculus we can sketch it
note: x² + x – 1 = 0 has roots -γ' = 1/γ and -γ



סה"כ קיבלנו:

$$f(x)-1-x=x\cdot(f(x)-1)+x^2 f(x)$$
$$\Leftrightarrow f(x)=\frac{1}{1-x-x^2}$$

על ידי חקירת פונקציה נקבל:

Using a *partial fraction expansion*

$$f(x)=\frac{1}{1-x-x^2}=\frac{A}{x+\gamma'}+\frac{B}{x+\gamma}$$

so

$$1=A(x+\gamma)+B(x+\gamma')$$

substituting the roots we find

$$A=-B=\frac{1}{\sqrt{5}}$$

and so

$$f(x)=\frac{1}{\sqrt{5}}\left(\frac{1}{x+\gamma'}-\frac{1}{x+\gamma}\right)=\frac{1}{\sqrt{5}}\left(\frac{\frac{1}{\gamma'}}{1+\frac{x}{\gamma'}}-\frac{\frac{1}{\gamma}}{1+\frac{x}{\gamma}}\right)$$

and comparing to the sum of a geometric series we can find :

$$f_n=\frac{1}{\sqrt{5}}\left(\gamma^{n+1}-(\gamma')^{n+1}\right)$$

התמרת Z: סיגנלים ספרתיים מזכירים פונקציות יוצרות. נגדיר באופן הבא:

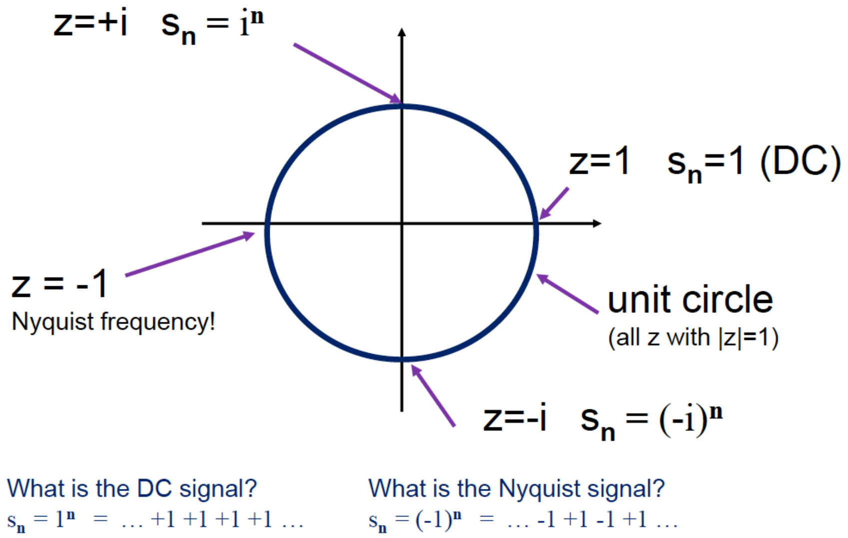
$$S(z) = zT(s_n) = \sum_{-\infty}^{\infty} s_n z^{-n}$$

הערות:

1. הסכום מ-$(-\infty)$ ועד ∞ במקום 0 עד ∞ בפונקציות יוצרות.
2. z^{-n} במקום z^n אבל זו קונבנציה כי n מתחיל ב-$(-\infty)$.
3. משתמשים ב-z ולא x כי זה מרמז שמדברים על ערכים **מרוכבים**. אין מישור תדר, יש מישור z. הרעיון יהיה להפוך סיגנל ספרתי לפונקציה מרוכבת. נראה שצריך בסיס חדש למרוכבים.
4. התמרת Z לוקחת סיגנל ומחזירה פונקציה מרוכבת, ולכן זו לא טרנספורמציה!

מישור Z:

כל נקודה במישור z מייצגת סיגנל $s_n = z^n$. נקודה על מעגל היחידה היא מספר שהערך המוחלט שלו הוא 1. כל המספרים המרוכבים הם $z = re^{i\theta}$, כלומר $r = 1, z = e^{i\theta}$. אם נקרא לזווית הזו w נקבל $z^n = e^{iwn}$, כלומר סינוסים!



עבור נקודה בתוך מעגל היחידה $r < 1$. כלומר ככל ש-n גדל, r^n קטן. לכן נקבל סינוס שתדירותו לא קבועה, אלא הוא דועך. מחוץ למעגל היחידה זה סינוס עם תדירות גדולה. הנקודות מחוץ למעגל היחידה לא יהיו סיגנל כשר, מכיוון שהאנרגיה אינסופית.

נביט במישור המרוכב:

- ככל שמתקרבים לראשית התדרים יותר נמוכים – במרכז נמצא אות האפס!
- $z = -1$ זה ה-Nyquist כי מעל תדירות דגימה זו נקבל כבר תדרים שליליים בגלל aliasing.

נראה תכונה חשובה של התמרת Z:

נניח שחישבנו התמרת Z עבור סיגנל $S(z) = zT(s)$. נרצה באמצעותה לחשב את $zT(\hat{z}^{-1}s)$. אותו סיגנל שהיה לנו, אבל הפעלנו עליו את אופרטור ההשהיה. נקבל:

$$zT(\hat{z}^{-1}s) = \sum_{-\infty}^{\infty} s_{n-1}z^{-n} = z^{-1} \cdot \sum_{-\infty}^{\infty} s_{n-1}z^{-(n-1)} = z^{-1}\sum_{-\infty}^{\infty} s_n z^{-n} = z^{-1} \cdot zT(s)$$

סה"כ קיבלנו את המשפט היסודי של התמרת Z: $zT(\hat{z}^{-1}x) = z^{-1}zT(x)$.
באופן דומה קיים עבור אופרטור הקידום בזמן: $zT(\hat{z}x) = z \cdot zT(x)$

<u>הקשר בין התמרת Z ו-DFT:</u> נקודות על מעגל היחידה הם סינוסים וקוסינוסים במישור המרוכב. DFT זה בסיס של אקספוננטים מרוכבים. עבור אות אינסופי – כל נקודה על מעגל היחידה מתאימה לאקספוננט מרוכב עם תדר מסוים בין 0 ל-i, ואחר כך שליליים בחזרה ל-0. כלומר כל 2π התדרים האפשריים הספרתיים.

- אם נבצע **התמרת Z רק על מעגל היחידה** בעצם חישבנו עם איברי הבסיס של התמרת פורייה, **זה התמרת פורייה בדיוק**.
- התמרת Z הכללית, היא גם התמרת פורייה אבל כולל איברים מחוץ למעגל, לכאורה יותר אינפורמציה.

מסקנות:

- התמרת Z היא **הרחבה של התמרת פורייה**, ומאפשרת דברים שהתמרת פורייה לא עובדת בהם.
- RoC (אזור התכנסות): התמרת Z אינה תלויה בזווית, אלא רק במרחק מהראשית (הערך המוחלט של z), כלומר תמיד **מתכנס על טבעת**. אם מעגל היחידה נמצא בתוך אזור ההתכנסות, ניתן להשתמש ב-DFT. אחרת, טור פורייה לא עובד, אבל ניתן למצוא תחום בו התמרת Z כן תעבוד. הרדיוסים הם גבולות אזור ההתכנסות.

מערכות

מערכות עיבוד אותות

<u>מערכת לעיבוד אותות:</u> מערכת שמקבלת בקלט 0 או יותר אותות, ומחזירה בתור פלט 1 או יותר אותות.

- למה נאפשר קלט ריק? מערכת יכולה להיות **מחולל** של אותות.
- מערכת עם קלט אחד ופלט אחד – **הגברה** למשל.

<u>תכונות של מערכות:</u>

- **<u>סיבתית</u>** <u>(casual):</u> הפלט y_n תלוי בקלטים מהעבר x_{n-m} אבל לא בקלטים מהעתיד x_{n+m} עבור $m \geq 0$.
- **<u>לינארית</u>** <u>(linear):</u> זה אומר שאם יש מערכת שעבור x_1 הייתה פולטת y_1, ועבור x_2 הייתה פולטת y_2, אז עבור $\boldsymbol{ax_1} + \boldsymbol{bx_2}$ חייבים לקבל $\boldsymbol{ay_1} + \boldsymbol{by_2}$. כלומר עבור $2x$ נצפה לקבל $2y$.
  - לא אומר שהפלט הוא לינארי בקלט: $y_n = ax_n + b$ (בגלל הגורם b, אז אם $b = 0$ זה לינארי)!
- **<u>אינווריאנטית לזמן/TI</u>** <u>(time invariant):</u> מערכת שאין לה שעון פנימי. כלומר אם מתקיים $x \to y$, אז גם $\hat{z}^n x \to \hat{z}^n y$.
  - $y_n = x_{-n}$ לא אינווריאנטית לזמן, מערכת כזאת צריכה לדעת מתי $n = 0$.
  - $y_n = \sin(\omega n)$ גם לא אינווריאנטית לזמן, תלויה באופן מפורש ב-n.
- **<u>מסנן/LTI</u>** <u>(filter):</u> מערכת שהיא **גם לינארית וגם אינווריאנטית לזמן** נקראת מסנן.

<u>נעבור על מספר דוגמאות ונבחן האם מדובר במערכת:</u>

דוגמה	מערכת?	ניתוח
1. יחס הזהות $y = x$	כן	המערכת לאו דווקא צריכה לעשות לוגיקה מורכבת על הקלט. זה אפילו מסנן.
2. $y = k$ שמתעלמת מהקלט x	כן	לא מחייבים שהפלט יהיה תלוי בקלט. כן TI, אבל לא L (אם נכפיל את הקלט ב-2 הפלט לא מוכפל ב-2, אלא אם $k = 0$ ואז זה כן לינארי).
3. $y = \pm\sqrt{x}$	כן	שקול מערכת עם 2 פלטים. אין סיגנל מוגדר כי זה גם פלוס וגם מינוס, אם $x = -1$ נקבל שורש מדומה ואמרנו שסיגנלים צריכים להיות ממשיים בלבד, אפשר לשים ערך מוחלט ולקחת ערך ממשי של הסיגנל.
4. מכשיר שמקבל פיצה ומחזיר את רשימת המרכיבים שלה	לא	אותיות זה לא סיגנל
5. $y = \sin\left(\frac{1}{t}\right)$	לא	אמרנו שזה לא סיגנל באשר כי יש לו רוחב סרט אינסופי
6. $y(t) = \int_{-\infty}^{t} x(t)$	כן	אופרטור הצובר לסיגנל אנלוגי
7. התמרת פורייה	לא	התמרת פורייה מעבירה בין ייצוגים של אותו סיגנל, וזה לא סיגנל
8. טלוויזיה	כן	מקבלת סיגנל ומחזירה סיגנל
9. ממיר D/A	כן	לא הגדרנו מה הסיגנל צריך להיות

<u>דוגמאות למערכות:</u>

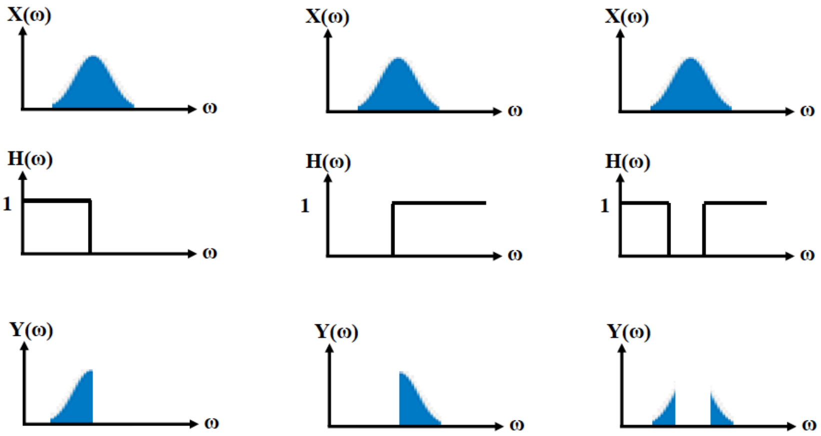
מערכת	פירוט
$y_n = x_n$	**הזהות**: מערכת סיבתית, לינארית, TI, לכן היא אפילו **מסנן**.
$y_n = g \cdot x_n$	נקרא **Amplifier**: מערכת סיבתית, לינארית, TI כל עוד g קבוע, ולכן **זה מסנן**.
$sign(x_n)$	נקרא **Saturator**: עבור סינוס נקבל גל ריבועי. זו מערכת אבל לא לינארית, כן TI אבל **זה לא מסנן**.
$y_n = f(x_n)$	מערכת טריוויאלית מאוד, פונקציה במשתנה אחד: אין שימוש בזיכרון, תלויים בקלט בזמן אחד.
$y = \hat{z}^{-1}x \Rightarrow y_n = x_{n-1}$	יש תלות ב-x אחרים, יש השהייה ולכן שימוש בזיכרון.
$y = \hat{\Delta}x$	תלוי בשני זמנים, x_n, x_{n-1}.
$y_n = \frac{1}{4}x_{n-1} + \frac{1}{2}x_n + \frac{1}{4}x_{n+1}$	נקרא **מחליק**: לא סיבתי כי משתמש ב-x_{n+1} אבל כן לינארי. זה TI ולכן זה **מסנן** לא סיבתי. מאפשר להיפטר ממעברים חדים.

מסננים

מסנן: מעביר/מפלטר דברים שקיימים בקלט, אבל לא יכול להמציא תדר שלא היה קיים. **מה שלא היה בקלט לא יכול להיות בפלט.**

- במקרה הספרתי: $Y_k = H_k X_k$.
- במקרה האנלוגי: $Y(\omega) = H(\omega)X(\omega)$.

אם תדר $X_k = 0$ אז בהכרח $Y_k = 0$. H_k **נקרא התגובה לתדר מסוים**. לכן גם מגבר הוא מסנן – אם נכניס סינוס בתדר מסוים, נקבל את אותו תדר סינוס מוגבר.



חוק המסנן: מתקיים $Y_k = H_k X_k$, לאחר הכפלה עם המסנן. למעלה נראה את הספקטרום של הסיגנל בכניסה, זה לא סינוס טהור. זה סיגנל אנלוגי שיש לו ספקטרום מלא, הרבה רכיבי סינוס (לאחר התמרת פורייה).

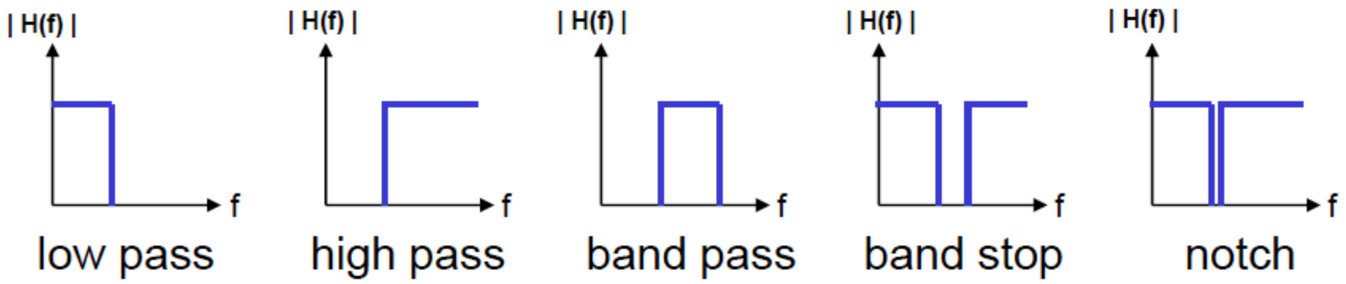
נשים לב שכדי לעבור לציר הזמן משתמשים בהתמרת פורייה ההפוכה, ונקבל בציר הזמן סיגנל לכאורה שונה לגמרי. לכן יש פעולות שיותר נוח לעשות בציר התדר ולא בציר הזמן.

התגובה לתדר: יכול להיות מספר מרוכב, שלאחר שמכפילים בו הוא מגביר כל תדר ומזיז אותו בפאזה:

$$H(\omega)e^{i\omega t} \cdot |H(\omega)|e^{i\angle H(\omega)} = |H(\omega)|e^{i(\omega t + \angle H(\omega))}$$

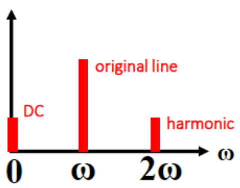
סוגי מסננים:

- low pass – מעביר רק תדרים נמוכים.
- high pass – מעביר רק תדרים גבוהים.
- band pass – מעביר תחום תדרים נבחר.
- band stop – מעביר תדרים פרט לטווח מסוים.
- notch – מוחקים רק תדר מסוים.



אלו הם מסננים אידיאליים, שהם 0 או 1 בכל מקום, אבל לא ניתן לממש אותם באופן מדויק.

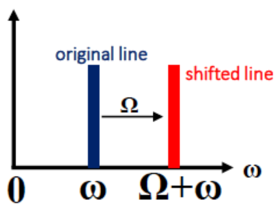
מערכות שאינן מסננים:



- $y_n = x_n + \varepsilon x_n^2$: **אינה ליניארית**. אם נכפיל את X ב-2 ה-Y לא יהיה מוכפל ב-2. אם נכניס $\sin(\omega n)$ עם תדר אחד, המערכת מוציאה שני תדרים שלא היו קיימים בקלט. לכן המערכת אינה מסנן.

For example, $y_n = x_n + \epsilon x_n^2$ $\sin^2(\phi) = ½ - ½ \cos(2\phi)$

if $x_n = \sin(\omega n)$ then $y_n = \sin(\omega n) + \epsilon/2 - \epsilon/2 \cos(2\omega n)$



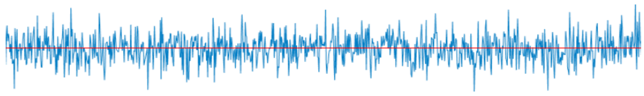
- $y(t) = e^{i\Omega t} \cdot x(t)$: **אינה TI**. אנחנו מכפילים בגורם שהוא תלוי ב-t, מסתמכים על הזמן. המערכת מוציאה תדרים חדשים ולכן זה לא מסנן.

if $x(t) = e^{i\omega t}$ then $y(t) = e^{i\Omega t} e^{i\omega t} = e^{i(\Omega+\omega)t}$

מסנן MA וקונבולוציה

<u>כדי ללמוד על מסננים נחקור את הבעיות הבאות:</u>

1) יש לנו סיגנל שאנחנו יודעים שהוא DC מסוים ($s_n = k$) בתוספת רעש כלשהו, $x_n = s_n + v_n$. הרעש עצמו v_n נטול-DC. איך נגלה את k? נבצע ממוצע על טווח זמן כמה שיותר גדול. ראינו שאם אין רכיב DC זה אומר שהממוצע של סיגנל הוא 0. לכן **אם נחשב ממוצע של הסיגנל המורעש נקבל את k**. הסימן <> הוא של תוחלת $\mathbb{E}$.



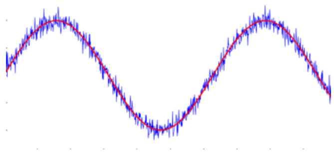
We **average** over as much time as we can

$$k = \langle x_n \rangle = \langle s_n + v_n \rangle = \langle s_n \rangle + \langle v_n \rangle = \langle s_n \rangle + 0$$

In practice, we take N samples

$$k = \frac{1}{N}\sum_{n=0}^{N-1} x_n$$

2) הפעם הסיגנל אינו קבוע, אבל בעל תדירויות נמוכות בלבד. נרצה שוב להשתמש בממוצע אבל על פני חלון זמן מסוים. לכן ניעזר בדבר שנקרא **Moving Average**.



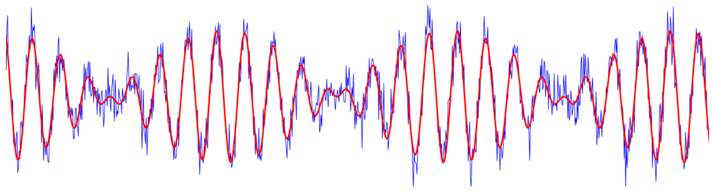
נמצא מקום ביניים כך שהרעש יתמצע ל-0 אבל לא יהרוס את הסיגנל המקורי s.

And then we move on to the next window

This is called **M**oving **A**verage

$$y_n = \frac{1}{L}\sum_{l=-L/2}^{+L/2} x_{n-l} \quad \text{(if L is odd then 1/(L+1))}$$

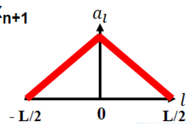
3) הסיגנל משתנה מהר, עדיין הרבה יותר איטי מהרעש. איך נבחר את L? נשתמש בממוצע ממושקל, אבל מעתה נקרא לזה פשוט Moving Average. נזיז את החלון קרוב יותר.



We perform a (generalized) **M**oving **A**verage but with non-equal coefficients

$$y_n = \sum_{l=n-L/2}^{n+L/2} a_l\, x_{n-l} \quad \text{where } \sum a_l = 1$$

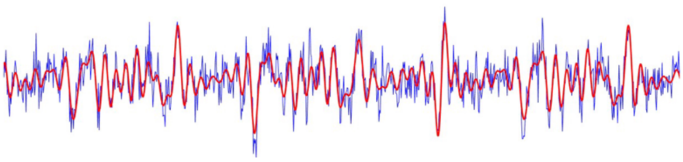
For example, the *smoother* $y_n = \frac{1}{4}x_{n-1} + \frac{1}{2}x_n + \frac{1}{4}x_{n+1}$



What coefficients return us to the original MA ?

Why do we often use *triangular* coefficients ?

4) מה נעשה במקרה הכללי בו הסיגנל s משתנה עוד יותר מהר? נצטרך להשתמש במסננים. נראה איפה נמצאים התדרים של הסיגנל, נתאים band pass filter ובכן נצליח לסנן את רוב הרעש. עדיין MA יכול לעבוד, אבל עם מקדמים פחות פשוטים.



<u>MA הוא תמיד מסנן</u> נוכיח ש-MA הוא ליניארי ו-TI (כי a_l הוא קבוע ולא תלוי בזמן).

LINEARITY

If we multiply the input by a gain g

$$y_n' = \sum a_l\, g x_{n-l} = g \sum a_l\, x_{n-l} = g\, y_n$$

If we add two inputs u and v which give outputs x and y

$$\sum a_l (u+v)_{n-l} = +\sum a_l\; u_{n-l} + \sum a_l\, v_{n-l} = x_n + y_n$$

TIME INVARIANCE

If we shift the input signal by m times (m positive or negative)

and the coefficients don't change!

$$y_n' = \sum a_l\, x_{(n+m)-l} = \left(\sum a_l\, x_{j-l}\right)_{j=n+m} = y_{n+m}$$

איך ליצור מסנן דיגיטלי? לחפש "digital filter design software free download" ☺

קונבולוציה: סכום על מכפלות. אינדקס אחד עולה ואינדקס אחד יורד, וסכום האינדקסים חייב להיות קבוע n.

We saw that to filter out noise we used the signal processing system

$$y_n = \sum_{l=n-L/2}^{n+L/2} a_l \, x_{n-l}$$

This is not causal, a similar causal filter is

$$y_n = \sum_{l=0}^{L-1} a_l \, x_{n-l}$$

קורלציה: כמו קונבולוציה רק ששני האינדקסים עולים וסכומם אינו n.

נראה קונבולוציה בשני הקשרים נוספים:

1) אם צועקים במערה, שומעים הד (x_{n-l}) בנוסף לזמן הנוכחי, כי הקול שיוצא מגיע לקיר וחוזר. רק חלק חוזר, יותר חלש. RTT זה הזמן שלוקח להגיע לקיר ולחזור. אם יש הרבה קירות, נקבל הד מהרבה מקומות שונים. זה גם מסנן MA סיבתי.

where ℓ is the RTT

$$y_n = x_n + a\, x_{n-\ell}$$

But there can be *many* echoes

$$y_n = x_n + a_1 x_{n-1} + a_2 x_{n-2} + a_3 x_{n-3} + a_4 x_{n-4} + \ldots$$

If the longest possible echo returns after L times

$$y_n = \sum_{l=0}^{L-1} a_l \, x_{n-l} \quad (\text{where } a_0 = 1, \text{ all other } 0 \le |a_l| < 1)$$

2) **כפל של שני מספרים/פולינומים:**

LONG MULTIPLICATION

$$\begin{array}{rrrrrrr}
 & & & B_3 & B_2 & B_1 & B_0 \\
 & & * & A_3 & A_2 & A_1 & A_0 \\
\hline
 & & & A_0B_3 & A_0B_2 & A_0B_1 & A_0B_0 \\
 & & A_1B_3 & A_1B_2 & A_1B_1 & A_1B_0 & \\
 & A_2B_3 & A_2B_2 & A_2B_1 & A_2B_0 & & \\
A_3B_3 & A_3B_2 & A_3B_1 & A_3B_0 & & & \\
\hline
\end{array}$$

$$C_3 = A_0B_3 + A_1B_2 + A_2B_1 + A_3B_0$$

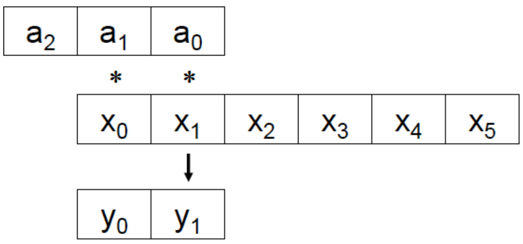
You learned about convolution
in grade school,
and then again in high school!

POLYNOMIAL MULTIPLICATION

$$(a_3 x^3 + a_2 x^2 + a_1 x + a_0)\,(b_3 x^3 + b_2 x^2 + b_1 x + b_0) =$$

$$a_3 b_3 x^6 + \ldots + (a_3 b_0 + a_2 b_1 + a_1 b_2 + a_0 b_3)\, x^3 + \ldots + a_0 b_0$$

איך מבצעים קונבולוציה בין סיגנל לבין מקדמים עם MA סיבתי? **אנחנו קודם הופכים את הכיוון של המסנן, ואז מקבלים קונבולוציה!**
בחישוב יש לולאה חיצונית של הזמנים, ולולאה פנימית של המקדמים של המסנן.



- הפעולה הבסיסית שעושים זה להכפיל ולצבור: Multiply and Accumulate, שיטה זו נקראת **MAC**. מעבד DSP הרבה יותר יעיל בחישוב MAC מאשר מעבד רגיל.
- אם יש קונבולוציה בציר הזמן, **בציר התדר צריכה להיות מכפלה רגילה**. כלומר בציר התדר זה ליניארי במספר האיברים. אבל תלוי כמה עולה לנו לעבור מציר הזמן לציר התדר. אם זה פחות מריבועי, כדאי לקחת את הסיגנלים מציר הזמן לציר התדר, להכפיל, ולחזור.

$$y_n = \sum_{l=0}^{L-1} a_l x_{n-l}$$

ראינו לגבי המסנן כי:

- אם יש בקלט X תדר מסוים 0, בפלט חייב להיות גם באותו תדר 0.
- אם כל הקלט הוא 0 חוץ מתדר מסוים (סינוס), הפלט חייב להיות סינוס באותו התדר.

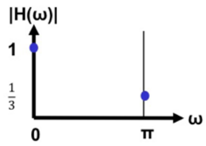
משפט המסנן: **סינוס הוא eigensignal (אות עצמי) של כל מסנן**. כלומר נוכל לקבל כפלט ממסנן את אותו סינוס רק מוגבר/מונחת/מוזז בפאזה (יכול להפוך לקוסינוס למשל).

נסתכל על הדוגמאות הבאות לחישוב $H(\omega)$.

דוגמה 1: $y_n = \frac{1}{3}(x_{n-1} + x_n + x_{n+1})$.

- לפני שמחשבים את $H(\omega)$ **נחשב מה המסנן עושה לסיגנל DC, התדר הנמוך ביותר**. למשל עבור ... 1,1,1, נקבל $y_n = \frac{1}{3}(1+1+1) = 1$. כלומר במקרה זה הוא לא מנחית ולא מגביר DC.
- כעת נשאל מה הוא **עושה לתדר הגבוה ביותר Nyquist**. בציר הזמן זה ... $1, -1, 1, -1,$. נחלק למקרים:
  - $y_n = \frac{1}{3}(1-1+1) = \frac{1}{3}$.
  - $y_n = \frac{1}{3}(-1+1-1) = -\frac{1}{3}$.

  סה"כ קיבלנו כי $y_n = \cdots, +\frac{1}{3}, -\frac{1}{3}, +\frac{1}{3}, -\frac{1}{3}, \ldots$. לכן סה"כ $H(\omega) = \frac{y_n}{x_n} = -\frac{1}{3}$ ועל כן $|H(\omega)| = \frac{1}{3}$.
- כעת נוכל להציג סיגנל בין 0 ל-Nyquist. נשתמש בסיגנל מרוכב שהוא $x_n = e^{i\omega n}$.
- זה מסנן low pass כי הוא מעביר יותר טוב תדרים נמוכים מאשר גבוהים. נרצה לקבל ב-Nyquist 0 בדיוק. בגלל שסינוס הוא eigensignal נקבל שמה שמוכפל בסינוס חייב להיות $H(\omega)$.



We have already found 2 points on the |H(ω)| plot

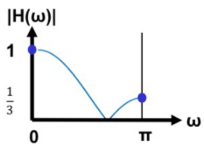
Now let's find the rest!

We substitute $x_n = e^{i\omega n}$ for arbitrary ω ($0 \le \omega \le \pi$)

$$
\begin{aligned}
y_n &= \frac{1}{3}\left(e^{i\omega(n-1)} + e^{i\omega n} + e^{i\omega(n+1)}\right) \\
&= \frac{1}{3}\left(e^{i\omega n}e^{-i\omega} + e^{i\omega n} + e^{i\omega n}e^{+i\omega}\right) \\
&= \frac{1}{3}\left(e^{-i\omega} + 1 + e^{+i\omega}\right)e^{i\omega n} \\
&= \frac{1}{3}\left(1 + 2\cos(\omega)\right)e^{i\omega n}
\end{aligned}
$$

So y_n is a constant times $e^{i\omega n}$ (of course - it had to be!)

and $H(\omega) = y_n / x_n = \frac{1}{3}\left(1 + 2\cos(\omega)\right)$

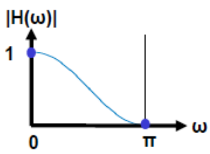


Why isn't this a nice enough low-pass filter?

דוגמה 2: המחליק $y_n = \frac{1}{4}x_{n-1} + \frac{1}{2}x_n + \frac{1}{4}x_{n+1}$.

- עבור DC: נקבל $y_n = \frac{1}{4} + \frac{1}{2} + \frac{1}{4} = 1$ אז $H(DC) = \frac{y_n}{x_n} = 1$.
- עבור Nyquist: נקבל 0 אז $H(Nyquist) = 0$.
- במקרה הכללי $x_n = e^{i\omega n}$.

$$
\begin{aligned}
y_n &= \tfrac{1}{4}e^{i\omega(n-1)} + \tfrac{1}{2}e^{i\omega n} + \tfrac{1}{4}e^{i\omega(n+1)} \\
&= \tfrac{1}{4}e^{i\omega n}e^{-i\omega} + \tfrac{1}{2}e^{i\omega n} + \tfrac{1}{4}e^{i\omega n}e^{+i\omega} \\
&= \left(\tfrac{1}{4}e^{-i\omega} + \tfrac{1}{2} + \tfrac{1}{4}e^{+i\omega}\right)e^{i\omega n} \\
&= \tfrac{1}{2}\left(1 + \cos(\omega)\right)e^{i\omega n}
\end{aligned}
$$



y is a sinusoid of the same frequency (of course – it has to be!)

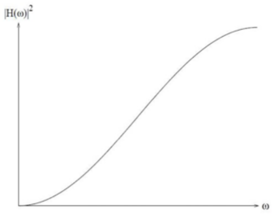
and $|H(\omega)| = ½ (1 + 2\cos(\omega))$ Why is this better?

<u>דוגמה 3:</u> $y=\widehat{\Delta} x$ כלומר $y_n = x_n - x_{n-1}$.

- זה מסנן MA סיבתי.
- עבור DC: נקבל 0.
- עבור Nyquist: נקבל 2.
- יש **פאזה כי זה סיבתי**, וצריכה להיות הזזה ב-$\frac{1}{2}$ יחידות זמן בביכול.
- ההפרש הסופי הראשון הוא כמו נגזרת, ולכן תדר גבוה זה נגזרת גדולה.

$H(\omega)\, e^{i\omega n} = e^{i\omega n} - e^{i\omega(n-1)}$ so $H(\omega) = 1 - e^{-i\omega} = e^{-i\omega/2}\,(e^{i\omega/2} - e^{-i\omega/2})$

So $|H(\omega)| = 2 \sin(\omega/2)$



This is a high-pass filter!

Why must it be high-pass?

Why is this complex (i.e., why does it have a phase shift)?

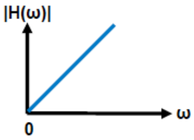
<u>דוגמה 4:</u> **נגזרת**: הוא high pass בצורה ברורה, בכל שעולים לתדר גבוה הוא מגביר יותר. ב-DC הוא אפס, כי זה נגזרת של קבוע. לכן **אינטגרל** באופן ההפוך יהיה low pass.

What does the analog derivative look like in the frequency domain?

Here is it easy enough to use sines

The derivative of $x(t) = \sin(\omega t)$ is $y(t) = \frac{d\,x(t)}{dt} = \omega \cos(\omega t)$

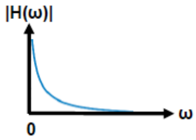
so $|H(\omega)| = \omega$ and there is a 90 degree phase shift



What about the analog integral?

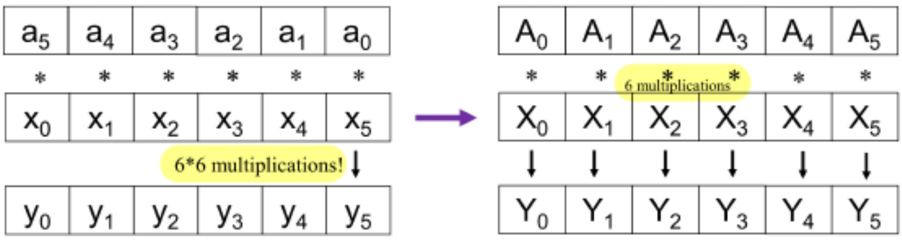
The integral of $x(t) = \sin(\omega t)$ is $y(t) = \int x(t)dt = -(1/\omega)\cos(\omega t)$

so $|H(\omega)| = 1/\omega$ and there is a 90 degree phase shift



מסנני AR ו-ARMA

ראינו שיש קשר בין קונבולוציה למכפלה: קונבולוציה של סיגנלים בציר הזמן, קשורה למכפלה של סיגנלים בציר התדר.



נוכל לחשב ממוצע על הערכים של סיגנל קלט באופן רקורסיבי.

$$\begin{aligned}
y_0 &= x_0 &&= x_0 \\
y_1 &= (x_0 + x_1)/2 &&= 1/2\, x_1 + 1/2\, y_0 \\
y_2 &= (x_0 + x_1 + x_2)/3 &&= 1/3\, x_2 + 2/3\, y_1 \\
y_3 &= (x_0 + x_1 + x_2 + x_3)/4 &&= 1/4\, x_3 + 3/4\, y_2 \\
y_n &= 1/(n+1)\, x_n + n/(n+1)\, y_{n-1} &&= (1-\beta)\, x_n + \beta\, y_{n-1}
\end{aligned}$$

- זה רקורסיבי כי כל y תלוי גם ב-y הקודם.
- זה סיבתי כי זה תלוי ב-x הנוכחי וב-y הקודמים, שהם בעצם תלויים ב-x הקודמים.
- זה ליניארי ו-TI (עם β קבוע) ולכן זה מסנן.

מסנן AR: **מסנן רקורסיבי**, **Auto-Regression**. יש דברים מסוימים שניתן לעשות בצורה יותר חזקה עם מסנן AR מאשר MA.

איך נחשב את התגובה לתדר $H(\omega)$?

1) לפתוח את הרקורסיה: עבור מסנן AR פשוט $y_n = x_n + \beta y_{n-1}$.

$$\begin{aligned} y_n &= x_n + \beta y_{n-1} \\ &= x_n + \beta(x_{n-1} + \beta y_{n-2}) = x_n + \beta x_{n-1} + \beta^2 y_{n-2} \\ &= x_n + \beta x_{n-1} + \beta^2(x_{n-2} + \beta y_{n-3}) = x_n + \beta x_{n-1} + \beta^2 x_{n-2} + \beta^3 y_{n-3} \\ &= \cdots \\ &= x_n + \beta x_{n-1} + \beta^2 x_{n-2} + \beta^3 x_{n-3} + \beta^4 x_{n-4} + \cdots \\ &= x_n + \sum_{m=1}^{\infty} \beta^m x_{n-m} \;=\; \sum_{m=0}^{\infty} \beta^m x_{n-m} \end{aligned}$$

קיבלנו סכום אינסופי: $y_n = \sum_{l=0}^{\infty} h_l x_{n-l}$. זה על פניו נראה כמו קונבולוציה, אבל הגדרנו אותה תמיד כסופית. לכן קוראים לזה **קונבולוציה אינסופית**. ההבדל בין AR ל-MA הוא ש-**MA קונבולוציה סופית ו-AR אינסופית**.

נשים לב כי AR הוא מסנן (בדומה להוכחה עבור MA):

- ליניארי: אפשר להראות.
- TI: נובע מהעובדה שהמקדם אינו תלוי בזמן.

2) דרך יותר פשוטה:

- עבור DC: לפי משפט המסנן, מתקיים כי $y_n = H_0$ קבוע כלשהו.
- עבור Nyquist: גם כאן נניח את משפט המסנן ונקרא לזה הפעם H_π.
- עבור המקרה הכללי נציב $H(\omega)$, ונקבל בסוף את התוצאה.
- **אנחנו מעלים בריבוע כי אנחנו רוצים לדעת רק את הערך המוחלט.**

Let's try our simple AR example $y_n = (1-\beta)\, x_n + \beta\, y_{n-1}$

What happens for DC? we purposely put the input gain back in!

We know that for all n $x_n=1$ and $y_n = H_0$

so $H_0 = (1-\beta) + \beta H_0$ or $H_0 (1-\beta) = (1-\beta)$ so $H_0 = 1$

What happens for Nyquist ?

We know that $x_n = \ldots -1 +1 -1 +1 \ldots$ and $y_n = \ldots -H_\pi +H_\pi -H_\pi +H_\pi \ldots$

so $H_\pi = (1-\beta) - \beta H_\pi$ or $H_\pi (1+\beta) = (1-\beta)$ so $H_\pi = (1-\beta) / (1+\beta)$

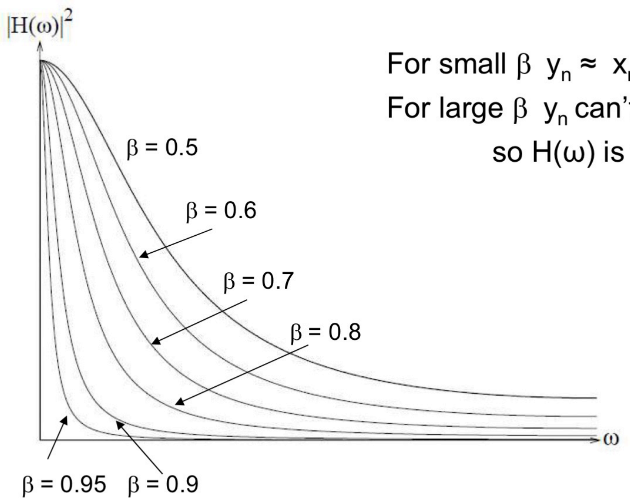
For general frequency ω : $x_n = e^{i\omega n}$ and $y_n = H(\omega)e^{i\omega n}$

so $H(\omega)e^{i\omega n} = (1-\beta)e^{i\omega n} + \beta H(\omega)e^{i\omega(n-1)}$ so $H(\omega) = (1-\beta) + \beta H(\omega)\, e^{-i\omega}$

so $H(\omega) = (1-\beta) / (1 - \beta\, e^{-i\omega})$ Why is this complex (i.e., has a phase shift)?

and $|H(\omega)|^2 = 1 / (1 - 2\beta\cos(\omega) + \beta^2)$

ככל שמגדילים את המקדמים, נקבל שזה מסנן low-pass:



$y_n = (1-\beta)\, x_n + \beta\, y_{n-1}$

For small β $y_n \approx x_n$ so $H(\omega) \approx 1$

For large β y_n can't keep up with x

so $H(\omega)$ is very low-pass

<u>דוגמה נוספת:</u> אופרטור $\hat{\gamma}$ (הצובר) כמסנן AR:

We once defined the accumulator y = $\hat{\Upsilon}$ x

by $y_n = \sum_{m=0}^{\infty} x_{n-m}$

(the inverse of the first finite difference - $\hat{\Upsilon}\hat{\Delta} = \hat{\Delta}\hat{\Upsilon} = 1$)

We can write the accumulator as an AR filter

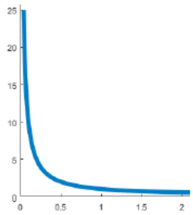
$$y_n = x_n + y_{n-1}$$

If we input DC this explodes! (AR filters can be *unstable*)

What is the frequency response?

$$H(\omega)e^{\mathrm{i}\omega n} = e^{i\omega n} + H(\omega)e^{\mathrm{i}\omega(n-1)}$$

Thus $$H(\omega) = \frac{1}{1-e^{-\mathrm{i}\omega}} = -\mathrm{i}e^{\mathrm{i}\omega/2}\frac{1}{2\sin(\omega/2)}$$



So |H(ω)| is very similar to the FR of the true integrator!

<u>מסנן ARMA:</u> ל-MA לא היה שום רקורסיה (ה-y של העבר), ול-AR יש רק את ה-x הנוכחי ולא של העבר. ARMA הוא המסנן הכללי ביותר, והוא מכיל את ה-MA (במקרה פרטי שבו $b_m = 0$) ואת ה-AR (במקרה פרטי שבו $a_l = 0$ חוץ מ-a_0). **כל מערכת שהיא ליניארית ו-TI ניתן לכתוב כמסנן ARMA.**

$$y_n = \sum_{n=0}^{L-1} a_l x_{n-l} + \sum_{m=1}^{M} b_m y_{n-m}$$

הצורה הסימטרית של ARMA – כל ה-y בצד שמאל, וכל ה-x בצד ימין:

$$y_n = \sum_{l=0}^{L-1} \alpha_l x_{n-l} + \sum_{m=1}^{M} \beta_m y_{n-m}$$

$$y_n - \sum_{m=1}^{M} \beta_m y_{n-m} = \sum_{l=0}^{L-1} \alpha^l x_{n-l}$$

where

$$\forall l \ \ \alpha_l = a_l \qquad \beta_0 = 1 \quad \forall m > 0 \ \ \beta_m = -b_m$$

אם כן, יש 3 דרכים שונות לכתוב את ARMA:

ARMA form $$y_n = \sum_{l=0}^{L-1} a_l x_{n-l} + \sum_{m=1}^{M} b_m y_{n-m}$$

Symmetric form (difference equation) $$\sum_{m=0}^{M} \beta_m y_{n-m} = \sum_{l=0}^{L} \alpha_l x_{n-l}$$ it looks *nicer* with L

Infinite convolution $$y_n = \sum_{l=0}^{\infty} h_l x_{n-l}$$

בעיית זיהוי המערכת

נתונה לנו מערכת עם קלט ופלט יחידים, עלינו להבין מה המערכת עושה.

בעיית זיהוי המערכת הקלה: מקבלים מערכת, **ואפשר לתת לה קלטים ככל שנרצה**, ולראות מה הפלטים שמקבלים. המערכת חייבת להיות TI ולינארית, כלומר היא **מסנן**.

- **פתרון בציר הזמן (IR – תגובה להלם):** הדבר הראשון שנעשה זה להכניס את סיגנל **ההלם (0 בכל הזמנים פרט לזמן $n=0$ בו הוא 1)** – נוכל לדעת מה יקרה לכל קלט אפשרי. ה-SUI הם בסיס, ולכן כל הקלטים שנכניס הם הזזה/הכפלה של SUI.
  - אם נכניס סיגנל עם 2 במקום 1 בזמן $n=0$ נקבל פלט פי 2 בגלל לינאריות.
  - אם נכניס סיגנל עם 1 בזמן $n=1$ ו-0 בכל זמן אחר (SUI), נקבל אותו דבר עם הזזה בגלל TI.

  כלומר UI הוא פתרון לבעיה הקלה של זיהוי המערכת. אם נדע את התגובה ל-UI, נדע את התגובה לכל SUI, ובגלל לינאריות ו-TI גם נדע את התגובה לכל צ"ל של SUI, וביוון שהם בסיס נדע את התגובה לכל אות x.

  The impulse response of a filter is universally called h_n

  - תגובה להלם של מסנן MA: אינו אפס לזמן סופי (L) ואז דועך – לכן נקרא **FIR**.

  What is the impulse response for an **MA** filter?

  $$h_n = \sum_{l=0}^{L-1} a_l \, \delta_{n-l,0} = a_n$$

  So, the MA coefficients are exactly the impulse response

  - תגובה להלם של מסנן ARMA/AR: אינו אפס לזמן אינסופי (בגלל הרקורסיה) – לכן נקרא **IIR**.

  What is the impulse response for an **ARMA** filter ?

  Use the infinite convolution form!

  $$h_n = \sum_{l=0}^{\infty} h_l \, \delta_{n-l,0} = h_n$$

  which is why we called these coefficients h in the first place!

- **פתרון בציר התדר (FR – תגובה לתדר):** לכל מסנן יש סיגנלים עצמיים שהם סינוסים. נוכל להכניס את כל התדרים בין 0 עד Nyquist של סינוסים. 0 זה DC.

  Another common choice of input are the *sinusoids*

  $$x_n = \sin(k\,n)$$

  But we need to enter all possible sinusoids (k=0, 1, …)

  However, from the filter law we know that sinusoids are *eigensignals* of filters

  the response to a sinusoid of frequency ω : $\sin(\omega\, n)$ is a sinusoid of frequency ω (or zero output)

  $$y_n = A_\omega \sin(\omega\, n + \phi_\omega)$$

  So we input all possible sinusoids but record only the **frequency response** FR

  - the gain A_k
  - the phase shift ϕ_k

k	A_k	ϕ_k
0		
1		
2		

- פתרון נוסף יכול להיות להכניס רעש לבן, כי הוא מכיל את כל התדרים. עם סיגנל ההלם, צריך להכניס **סיגנל אחד** ומחכים n דגימות. עבור תדר מכניסים n **סיגנלים**. **כלומר סה"כ צריך n דגימות**.

גרסה מוחלשת של בעיית זיהוי המערכת הקשה: יש לנו מסנן MA עם 3 מקדמים. בנוסף, הסיגנל בקלט יהיה 0 עד זמן מסוים. נסמן בה"כ $n=0$ עבור הזמן הראשון בו הסיגנל בקלט אינו 0. איך נמצא את המקדמים?

$$y_n = \sum_{l=0}^{L=2} a_l\, x_{n-l} = a_0 x_n + a_1 x_{n-1} + a_2 x_{n-2}$$

- We further assume that the input was zero until time n=0
  (we can always take the time the signal starts to be n=0 …)
  so $x_{n<0} = 0$

We need to find 3 unknowns – a_0, a_1, and a_2
so we will need three equations to solve

ראשית עבור $n=0$ נקבל את המקדם הראשון, לאחר מכן עבור $n=1$, לבסוף $n=2$. זה קל לפתור כי הנחנו ש-$x_{n<0}=0$.

First let's write the equation for n=0

$$y_0 = a_0 x_0 + a_1 x_{-1} + a_2 x_{-2} = a_0 x_0$$

Since $x_0 \neq 0$ we can divide to find $a_0 = y_0 / x_0$

Next we write the equation for n=1

$$y_1 = a_0 x_1 + a_1 x_0 + a_2 x_{-1} = a_0 x_1 + a_1 x_0$$

What do we already know?

$$y_1 = a_0 x_1 + a_1 x_0 \quad \text{so} \quad a_1 = (y_1 - a_0 x_1) / x_0$$

which is OK since $x_0 \neq 0$

Finally we write the equation for n=2

$$y_2 = a_0 x_2 + a_1 x_1 + a_2 x_0$$

so $a_2 = (y_2 - a_0 x_2 - a_1 x_1) / x_0$ which is OK since $x_0 \neq 0$

יש דרך יותר פשוטה לכתוב, באופן מטריציוני. יש כאן קשר בין קונבולוציה לבין מכפלה מטריציונית. המטריצה מהצורה הזאת נקראת טפליצית (Toeplitz), יש לה את אותו הערך לאורך כל האלכסונים.

$$y_0 = a_0 x_0$$
$$y_1 = a_0 x_1 + a_1 x_0$$
$$y_2 = a_0 x_2 + a_1 x_1 + a_2 x_0$$

in matrix format (with the coefficient as the vector)

הפיכה כי היא
משולשית תחתונה

$$\begin{pmatrix} y_0 \\ y_1 \\ y_2 \end{pmatrix} = \underbrace{\begin{pmatrix} x_0 & 0 & 0 \\ x_1 & x_0 & 0 \\ x_2 & x_1 & x_0 \end{pmatrix}} \begin{pmatrix} a_0 \\ a_1 \\ a_2 \end{pmatrix}$$

which can be solved by inverting the matrix

$$\begin{pmatrix} a_0 \\ a_1 \\ a_2 \end{pmatrix} = \underbrace{\begin{pmatrix} x_0 & 0 & 0 \\ x_1 & x_0 & 0 \\ x_2 & x_1 & x_0 \end{pmatrix}^{-1}}_{\text{מטריצה טפליצית}} \begin{pmatrix} y_0 \\ y_1 \\ y_2 \end{pmatrix}$$

הבעיה הקשה: **המסנן עדיין MA** עם 3 מקדמים, אבל הסיגנל לא מתאפס מזמן מסוים. שוב צריך 3 משוואות כדי למצוא 3 מקדמים. לכן נוסיף 2 משוואות עוקבות. זה מצריך שימוש ב-5 זמנים:

So pick any n (there is nothing special about any time) and write three consecutive equations

$$\begin{aligned} y_n &= a_0 x_n + a_1 x_{n-1} + a_2 x_{n-2} \\ y_{n+1} &= a_0 x_{n+1} + a_1 x_n + a_2 x_{n-1} \\ y_{n+2} &= a_0 x_{n+2} + a_1 x_{n+1} + a_2 x_n \end{aligned}$$

Note that we need to observe 5 consecutive times

n-2 x_{n-2} **n-1** x_{n-1} **n** x_n and y_n

n+1 x_{n+1} and y_{n+1} **n+2** x_{n+2} and y_{n+1}

נכתוב בצורה מטריציונית. זו אינה מטריצה משולשית תחתונה, אבל היא עדיין טפליצית. יש אלגוריתם יציב (Levinson-Durbin) שמאפשר להפוך אותה בזמן $O(n^2)$.

$$\begin{pmatrix} y_n \\ y_{n+1} \\ y_{n+2} \end{pmatrix} = \begin{pmatrix} x_n & x_{n-1} & x_{n-2} \\ x_{n+1} & x_n & x_{n-1} \\ x_{n+2} & x_{n+1} & x_n \end{pmatrix} \begin{pmatrix} a_0 \\ a_1 \\ a_2 \end{pmatrix}$$

משוואות אלה נקראות משוואות **Wiener-Hopf**. בגלל רעש, אם נפתור פעם שנייה נקבל מקדמים מעט שונים. נוכל להשתמש **במיצוע מההתחלה לפני שהופכים את המטריצה** (לא ניכנס לזה).

<u>אותה בעיה עם **מסנן AR**:</u>

$$y_n = x_n + \sum_{m=1}^{M=3} b_m y_{n-m} = x_n + b_1 y_{n-1} + b_2 y_{n-2} + b_3 y_{n-3}$$

צריך 6 זמנים עוקבים כדי לחשב את המקדמים:

$$\begin{aligned} y_n &= x_n + b_1 y_{n-1} + b_2 y_{n-2} + b_3 y_{n-3} \\ y_{n+1} &= x_{n+1} + b_1 y_n + b_2 y_{n-1} + b_3 y_{n-2} \\ y_{n+2} &= x_{n+2} + b_1 y_{n+1} + b_2 y_n + b_3 y_{n-1} \end{aligned}$$

בבתיב מטריציוני:

$$\begin{pmatrix} y_n \\ y_{n+1} \\ y_{n+2} \end{pmatrix} = \begin{pmatrix} x_n \\ x_{n+1} \\ x_{n+2} \end{pmatrix} + \begin{pmatrix} y_{n-1} & y_{n-2} & y_{n-3} \\ y_n & y_{n-1} & y_{n-2} \\ y_{n+1} & y_n & y_{n-1} \end{pmatrix} \begin{pmatrix} b_1 \\ b_2 \\ b_3 \end{pmatrix}$$

$$\underset{\text{vector}}{\underline{y}} = \underline{x} + \underset{\text{matrix}}{\underline{\underline{y}}}\,\underline{b} \qquad \text{so} \qquad \underline{b} = \underline{\underline{y}}^{-1}\,\underline{(y - x)}$$

משוואות אלה נקראות משוואות **Yule-Walker**.

פתרון הבעיה הקשה בציר התדר: יש לנו משהו שעובד במישור Z. אפשר להשתמש בחוק המסנן ישירות. כיוון ש-$Y_k = H_k X_k$.

נתחיל מהצורה של ARMA בכתיבה של קונבולוציה אינסופית: $y_n = \sum_{k=0}^{\infty} h_k x_{n-k}$.

ניעזר בהתמרת Z ונפעיל אותה על שני הצדדים של המשוואה:

$$\begin{aligned} Y(z) &= \sum_{n=-\infty}^{\infty} y_n z^{-n} \\ &= \sum_{n=-\infty}^{\infty} \left\{ \sum_{k=-\infty}^{\infty} h_k x_{n-k} \right\} z^{-n} \\ &= \sum_{k=-\infty}^{\infty} h_k \sum_{n=-\infty}^{\infty} x_{n-k} z^{-n} \\ &= \sum_{k=-\infty}^{\infty} h_k z^{-k} \sum_{m=-\infty}^{\infty} x_m z^{-m} \\ &= \quad H(z) \qquad X(z) \end{aligned}$$

- אם נסתכל רק על מעגל היחידה נקבל $Y(\omega) = H(\omega) \cdot X(\omega)$ (נחזור להתמרת פורייה, ולמשפט המסנן).
- בנוסף, **התגובה לתדר H_k היא התמרת פורייה של התגובה להלם h_n.**
- נגדיר את **פונקציית התמסורת**: $H(z) = \sum_{n=-\infty}^{\infty} h_n z^{-n}$. היא מעבירה אותנו במישור ה-Z מה-$zT(X)$ ל-$zT(Y)$.

כעת לצורה של ARMA הסימטרית: באותו אופן נהפוך את סדר הסכימה.

$$\sum_{m=0}^{M} \beta_m y_{n-m} = \sum_{l=0}^{L} \alpha_l x_{n-l} \qquad \text{remember } \beta_0 = 1$$

Now we take the zT of both sides

$$\sum_{n=-\infty}^{\infty} \left(\sum_{m=0}^{M} \beta_m y_{n-m} \right) z^{-n} = \sum_{n=-\infty}^{\infty} \left(\sum_{l=0}^{L} \alpha_l x_{n-l} \right) z^{-n}$$

and do our usual tricks to get

$$\begin{array}{ccccc} \sum_{m=0}^{M} \beta_m z^{-m} & \sum_{n=-\infty}^{\infty} y_n z^{-n} & = & \sum_{l=0}^{L} \alpha_l z^{-l} & \sum_{n=-\infty}^{\infty} x_n z^{-n} \\ B(z) & Y(z) & = & A(z) & X(z) \end{array}$$

נקבל:

- $Y(z) = \frac{A(z)}{B(z)} \cdot X(z)$ ולכן $H(z) \cdot X(z) = Y(z)$. $\boldsymbol{H(z) = \frac{A(z)}{B(z)}}$ שזו פונקציה רציונלית. מהמשפט היסודי של האלגברה, מעל המרוכבים לפולינום ממעלה n יש n שורשים.
- לשורשים של $A(z)$ קוראים **אפסים**. אלו ה-z שעבורם המנה מתאפסת.
- לשורשים של $B(z)$ קוראים **קטבים**, שם $H(z)$ מתפוצץ.

$$A(z) = \sum_{l=0}^{L} \alpha_l z^l = \prod_{l=0}^{L} (z - \zeta_l)$$

$$B(z) = \sum_{m=0}^{M} \beta_m z^m = \prod_{m=0}^{M} (z - \pi_m)$$

משפט: כל פונקציה רציונלית נקבעת לפי השורשים והקטבים בצורה חד-ערכית עד כדי הכפלה בקבוע. כלומר בהינתן השורשים והקטבים נוכל לחשב את $H(z)$ באופן חד-ערכי.

- אם נמצא פונקציית תמסורת $H(z)$ שיש לה **רק אפסים ולא קטבים**, ל-$B(z)$ אין שורשים ולכן הוא פולינום קבוע, כלומר כל β_m שווים אפס ונקבל שזה מסנן **MA**.
- אם יש **קטבים בלבד ואין אפסים**, $A(z)$ קבוע. כלומר α_n שווים אפס ונקבל מסנן **AR**.
- אם יש **גם שורשים וגם קטבים** נקבל **ARMA** כללי.

FIR	MA	all zeros
IIR	AR	all poles
	ARMA	zeros and poles

סיכום וניתוח מסננים

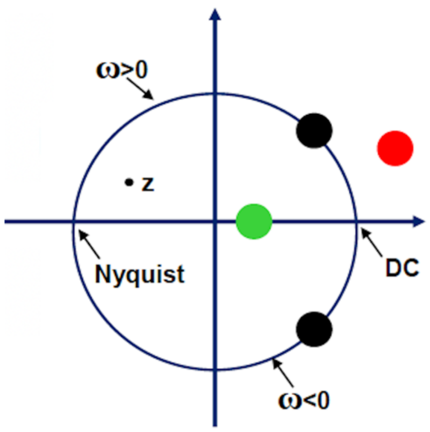
מסנני MA שראינו עם אפסים:

מסנן	פרטים
$y_n = x_n - x_{n-1}$	הפרש סופי ראשון, DC אפס (נקודה בימין)
$y_n = x_n + x_{n-1}$	Nyquist אפס (נקודה בשמאל)
$y_n = x_n + x_{n-2}$	אפס בחצי Nyquist, שזה $\omega = \frac{\pi}{2}$ (נקודה מעל ונקודה מתחת)

מסנני AR שראינו עם קטבים:

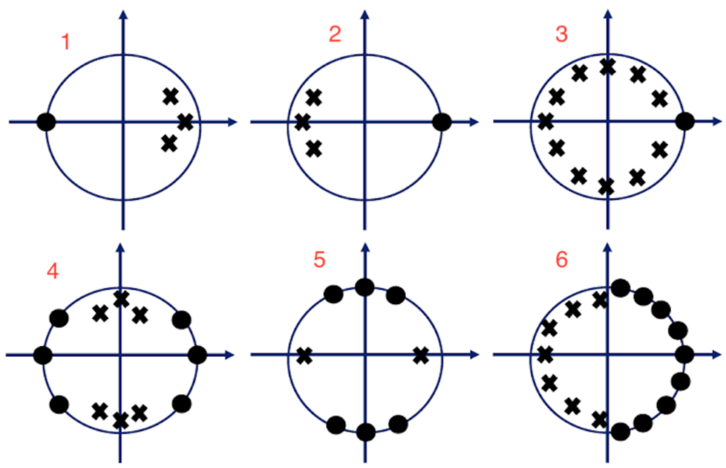
מסנן	פרטים
$y_n = x_n + y_{n-1}$	הצובר, יש לו קוטב ב-DC
$y_n = x_n - y_{n-1}$	קוטב ב-Nyquist
$y_n = x_n - y_{n-2}$	קוטב בחצי Nyquist, שזה $\omega = \frac{\pi}{2}$

נקודות חשובות במישור Z:



- אפסים:
  - אם יש לנו אפס של $H(z)$ בנקודה של ה-DC, נקבל שעבור קלט $x(z)$ הפלט הוא $0 = x(z) \cdot x(z)$, כלומר זה מאפס את ה-DC.
  - אם יש אפס ב-Nyquist זה מאפס אותו.
  - אם יש אפסים בצד ימין של ציר x זה מסנן high pass, כי מאפסים תדרים נמוכים.
  - אם יש אפסים בצד שמאל של ציר x זה מסנן low pass.
  - אם יש אפסים בכל מיני מקומות אז המסנן מאפס כמה תדרים.
- קטבים:
  - קוטב משמעותו הגברה. לכן אם יש קוטב על מעגל היחידה, זה אומר שמכניסים סינוס בתדר הזה, זה מתפוצץ ולא נרצה את זה אף פעם.
  - קטבים על ומחוץ למעגל היחידה גורמים למסנן להיות לא יציב.
  - **קטבים בתוך מעגל היחידה מגבירים את האות.**

דיאגרמת אפסים וקטבים:



- נסמן קטבים ב-X ואפסים בנקודה.
- כאשר שמים קטבים/אפסים על ציר x שמים רק אחד. מעל/מתחת לציר x שמים שניים – בשיקוף לציר x. הסיבה היא שאנחנו רוצים לקבל סיגנל ממשי – סינוסים. אם יש שורש מדומה, צריך להכפיל אותו במינוס של הצמוד כדי לקבל מספר ממשי.
- 1 – הגברנו תדרים נמוכים, איפסנו תדר Nyquist שהוא התדר הכי גבוה, ויש שני קטבים לא על ציר x ולכן פונקציית התמסורת ממשית והמסנן יציב – **low pass**.
- 2 – הפוך – **high pass**.
- 3 – זהו **notch** של DC. זה מאפשר לתקן את הסיגנל כך שהממוצע יהיה סביב 0. הכול מוגבר חוץ מתדר אחד – DC. אם לא היו קטבים זה היה מנחית את כל התדרים הנמוכים.
- 4 – מגבירים את התדר שהוא חצי Nyquist ומנחיתים תדרים גם גבוהים וגם נמוכים – **band pass**.
- 5 – מורידים תדר חצי Nyquist ומגבירים תדר DC וגם Nyquist – **band stop**.

איך מנתחים מסנן?

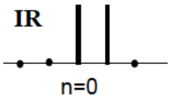
1. מזיזים את כל ה-y לצד שמאל וה-x לצד ימין – יוצרים את הצורה הסימטרית.
2. כותבים את המשוואה במונחי סיגנלים באמצעות אופרטור השהייה $\hat{z}^{-1}$.
3. לוקחים התמרת Z על שני צידי המשוואה עם החוק שלנו $zT(\hat{z}^{-1}x) = z^{-1} \cdot zT(x)$.
4. מחלקים כדי לקבל את $Y(z)$ בצד שמאל.
5. הופכים את המונה ואת המכנה לפולינומים (ללא z^M).
6. מוצאים את השורשים של הפולינומים.

מה מאפשר לזהות מסנן? המרות:

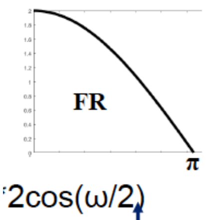
to → from ↓	**a, b coefficients**	**α,β coefficients**	**impulse response**	**frequency response**	**transfer function**	**gain and pole-zero diagram**
a, b coefficients	identity	subtraction of y terms	MA: h=a AR + ARMA: recursion	substitute $x=e^{ikn}$	write using z^{-1} and extract	through transfer function
α,β coefficients	addition of y terms	identity	same as a,b	same as a,b	same as a,b	same as a,b
impulse response	MA: a=h ARMA: recursion	through a,b	identity	DFT	zT	through transfer function
frequency response	through IR or transfer function	same as a,b	iDFT	identity	analytic continuation	through transfer function
transfer function	through α,β	B(z) Y(z) = A(z) X(z)	izT	substitute $z = e^{i\omega}$	identity	find roots
gain and pole-zero diagram	through transfer function	through transfer function	through transfer function	substitution	multiply terms to get polynomial	identity

דוגמה 1 (MA סיבתי): $y_n = x_n + x_{n-1}$

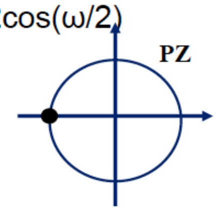
- MA סיבתי כי הוא תלוי ב-$n, n-1$.
- אפשר לראות שהוא עושה מיצוע של איברים קרובים (סכום), אפשר לנחש שהוא low pass.
- <u>תגובה לתדר (מקרים בסיסיים):</u>
  - נבדוק DC: נקבל $y_n = 2$ ולכן נקבל $H(DC) = 2$.
  - נבדוק Nyquist: נקבל $y_n = 0$ ולכן נקבל $H(Nyquist) = 0$.
  - בשרטוט אנחנו מחפשים את $|H(\omega)|^2$, אז נתחיל מ-4 ונסיים ב-0.
- <u>תגובה להלם:</u>
  - עד ל-$0 = n$ חייב להיות 0 כי זה סיבתי. **התגובה להלם של כל מסנן סיבתי חייבת להיות 0 עד $n = 0$ (לא יודע שהולך להיות הלם).**
  - $y_0 = x_0 + x_{-1} = 1 + 0 = 1$, ואז $y_1 = x_1 + x_0 = 0 + 1 = 1$. נקבל $\mathbf{1, 1}$.



- <u>תגובה לתדר (מקרה כללי):</u>
  - תדר כללי $x_n = e^{i\omega n}$. נקבל $H(\omega)e^{i\omega n} = y_n = e^{i\omega n} + e^{i\omega(n-1)} = e^{i\omega n}(1 + e^{-i\omega})$.
  - נקבל $H(\omega) = 1 + e^{-i\omega} = e^{-i\frac{\omega}{2}}\left(e^{+i\frac{\omega}{2}} + e^{-i\frac{\omega}{2}}\right) = phase \cdot 2\cos\left(\frac{\omega}{2}\right)$.

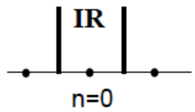


- <u>פונקציית התמסורת:</u>
  - כותבים את המשוואה לא בתור ערכים של סיגנל, אלא בתור **משוואה של סיגנלים**:
    $$y_n = x_n + x_{n-1} \to y = x + \hat{z}^{-1}x = (1 + \hat{z}^{-1})x$$
  - נפעיל התמרת Z על שני הצדדים:
    $$Y(z) = (1 + z^{-1})X(z)$$
  - מותר להכפיל בכל חזקה של z כדי לקבל פולינום. נכפיל ב-z^1 ונקבל: $\left(1 + \frac{1}{z}\right) \cdot z = \mathbf{z + 1}$.
  - קיבלנו $H(z) = \frac{z+1}{1}$.
  - הפולינום ממעלה 1, אין קטבים ולכן זה MA. השורש של המונה הוא $\mathbf{1}-$ (מה שמאפס אותו).
  - נשים לב שיש לנו 0 בתדר Nyquist, זה אומר שמאפסים תדרים גבוהים ולכן זה **low pass**.

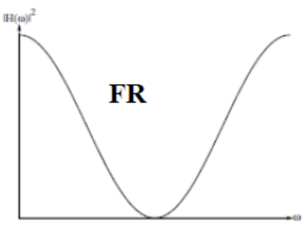


דוגמה 2 (MA לא סיבתי) – $y_n = x_{n-1} + x_{n+1}$:

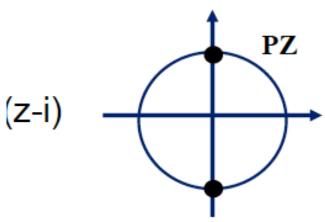
- MA לא סיבתי כי הוא תלוי ב-$n + 1$.
- <u>תגובה לתדר (מקרים בסיסיים):</u>
  - נבדוק DC: נקבל $y_n = 2$ ולכן נקבל $H(DC) = 2$.
  - נבדוק Nyquist: נקבל $y_n = 2, -2, 2, -2, \ldots$ ולכן נקבל $H(Nyquist) = 2$.
  - אפשר לנחש שזה band-stop עבור הסיגנל $x = 1, 0, -1, 0, 1, 0, -1, 0, \ldots$ ונקבל $H(mid) = 0$.



- <u>תגובה להלם:</u>
  - $y_{-1} = x_{-2} + x_0 = 1$. $y_0 = x_{-1} + x_1 = 0$, $y_1 = x_0 + x_2 = 1$. נקבל $\mathbf{1, 0, 1}$.
- <u>תגובה לתדר (מקרה כללי):</u>
  - תדר כללי $x_n = e^{i\omega n}$.
    נקבל $H(\omega)e^{i\omega n} = y_n = e^{i\omega(n-1)} + e^{i\omega(n+1)} = e^{i\omega n}(e^{-i\omega} + e^{i\omega})$.
  - נקבל $H(\omega) = e^{-i\omega} + e^{i\omega} = 2\cos(\omega)$.

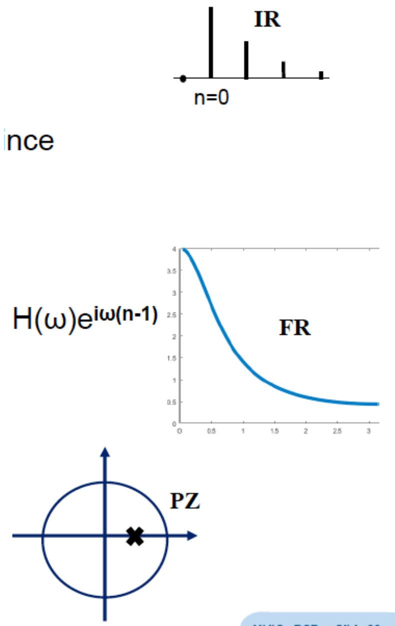


- <u>פונקציית התמסורת:</u>
  - המשוואה היא $y = \hat{z}^{-1}x + \hat{z}^{+1}x = (\hat{z}^{-1} + \hat{z}^{+1})x$.
  - נפעיל התמרת Z על שני הצדדים: $Y(z) = (z^{-1} + z)X(z)$.
  - נכפיל שוב ב-z כדי לקבל פולינום וקיבלנו $H(z) = \frac{1+z^2}{1}$. השורשים הם $\pm\boldsymbol{i}$.
- אנחנו רוצים את ה-H ממשי, ולכן האפסים יבואו בזוגות שהם צמודים אחד לשני, כדי שה-z יהיה ממשי.



דוגמה 3 (AR סיבתי: לשים לב כאן עושים שימוש בהצבה של H) – $y_n = x_n + \frac{1}{2}y_{n-1}$:

- AR סיבתי כי הוא תלוי ב-x_n וב-y קודמים.
- אפשר לראות שהוא עושה מיצוע של איברים קרובים (סכום), אפשר לנחש שהוא low pass.
- <u>תגובה לתדר (מקרים בסיסיים):</u>
  - כאן ניעזר במשפט המסנן, $y_n = H_0 \cdot x_n$ לכל n במקרים הבסיסיים.
  - נבדוק DC: נקבל כי $y_n = H_0, H_0, H_0, \dots$ ולכן $H_0 = 1 + \frac{1}{2}H_0$ ולכן נקבל $H(DC) = 2$.
  - נבדוק Nyquist: נקבל כי $y_n = H_0, -H_0, H_0, \dots$ ולכן:
    - $H_0 = 1 - \frac{1}{2}H_0$ ולכן נקבל $H(Nyquist) = \frac{2}{3}$.
    - $-H_0 = -1 + \frac{1}{2}H_0$ ולכן גם נקבל $H(Nyquist) = \frac{2}{3}$.
- <u>תגובה להלם:</u>
  - עד ל-$n = 0$ חייב להיות 0 כי זה סיבתי. **התגובה להלם של כל מסנן סיבתי חייבת להיות 0 עד $n = 0$ (לא יודע שהולך להיות הלם).**
  - $y_0 = x_0 + \frac{1}{2}y_{-1} = 1 + 0 = 1$, ואז $y_1 = x_1 + \frac{1}{2}y_0 = 0 + \frac{1}{2} = \frac{1}{2}$. נקבל $\mathbf{1, \frac{1}{2}, \frac{1}{4}, \frac{1}{8}}, \dots$.
- <u>תגובה לתדר (מקרה כללי):</u>
  - תדר כללי $x_n = e^{i\omega n}$. נקבל $H(\omega)e^{i\omega n} = y_n = e^{i\omega n} + \frac{1}{2}H(\omega)e^{i\omega(n-1)} = e^{i\omega n}\left(1 + \frac{1}{2}H(\omega)e^{-i\omega}\right)$.
  - נקבל $H(\omega) = 1 + \frac{1}{2}H(\omega)e^{-i\omega} \Leftrightarrow H(\omega) - \frac{1}{2}H(\omega)e^{-i\omega} = 1 \Leftrightarrow H(\omega)\left(1 - \frac{1}{2}e^{-i\omega}\right) = 1$.
  - לכן סה"כ $H(\omega) = \frac{1}{1-\frac{1}{2}e^{-i\omega}} = \frac{2}{2-e^{-i\omega}}$.
  - משם נקבל $|H(\omega)|^2 = \frac{4}{5-4\cos(\omega)}$.
  -
- <u>פונקציית התמסורת:</u>
  - ראשית נעביר אגפים כדי לקבל צורה סימטרית:
  $$y_n = x_n + \frac{1}{2}y_{n-1} = y_n - \frac{1}{2}y_{n-1} = x_n$$
  - כותבים את המשוואה לא בתור ערכים של סיגנל, אלא בתור **משוואה של סיגנלים**:
  $$y_n - \frac{1}{2}y_{n-1} = x_n \rightarrow x = y - \frac{1}{2}\hat{z}^{-1}y = \left(1 - \frac{1}{2}\hat{z}^{-1}\right)y$$
  $$\Leftrightarrow y = \frac{1}{1 - \frac{1}{2}\hat{z}^{-1}}x$$
  - נפעיל התמרת Z על שני הצדדים:
  $$Y(z) = \frac{1}{1 - \frac{1}{2}z^{-1}}X(z)$$
  - נכפיל ב-z^1 וקיבלנו $H(z) = \frac{1}{z-\frac{1}{2}}$.
  - הפעם הפולינום במכנה ולכן זה AR. השורש שלו הוא $\frac{1}{2}$ וזה **קוטב**.



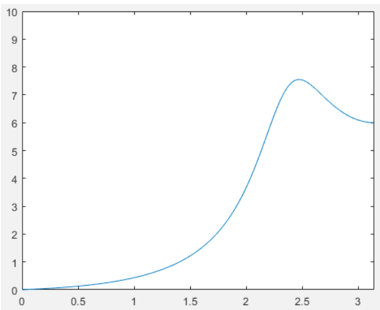
דוגמה 4 (ARMA סיבתי) – $y_n = x_n - \frac{3}{2}x_{n-1} + \frac{1}{2}x_{n-2} - y_{n-1} - \frac{1}{2}y_{n-2}$:

- ARMA סיבתי כי הוא תלוי גם ב-x קודמים וגם ב-y קודמים.
- <u>תגובה לתדר (מקרים בסיסיים):</u>
  - כאן ניעזר במשפט המסנן, $y_n = H_0 \cdot x_n$ לכל n במקרים הבסיסיים.
  - נבדוק DC: $H_0 = 1 - \frac{3}{2} + \frac{1}{2} - H_0 - \frac{1}{2}H_0$ ולכן $H(DC) = 0$.
  - נבדוק Nyquist: $H_0 = 1 + \frac{3}{2} + \frac{1}{2} + H_0 - \frac{1}{2}H_0$ ולכן $H(Nyquist) = 6$.
- <u>תגובה להלם:</u>
  - עד ל-0 = n חייב להיות 0 כי זה סיבתי. **התגובה להלם של כל מסנן סיבתי חייבת להיות 0 עד $n = 0$ (לא יודע שהולך להיות הלם).**
  - נחשב ונקבל ... $\mathbf{0}, -\frac{5}{4}, -\frac{5}{2}, +\frac{5}{2}, -\frac{5}{2}, \mathbf{1}$.
- <u>תגובה לתדר (מקרה כללי):</u>
  - תדר כללי $x_n = e^{i\omega n}$. שוב נציב ונקבל:

$$H(\omega)e^{i\omega n} = e^{i\omega n} - \frac{3}{2}e^{i\omega(n-1)} + \frac{1}{2}e^{i\omega(n-2)} - H(\omega)e^{i\omega(n-1)} - \frac{1}{2}H(\omega)e^{i\omega(n-2)} \Leftrightarrow$$

$$H(\omega) = \frac{1 - \frac{3}{2}e^{i\omega} + \frac{1}{2}e^{2i\omega}}{1 + e^{i\omega} + \frac{1}{2}e^{2i\omega}} = \frac{-\frac{3}{2} + \cos(\omega) - \frac{1}{2}e^{-i\omega}}{1 + \cos(\omega) - \frac{1}{2}e^{-i\omega}}$$

  - עבור השרטוט נעלה בריבוע:



$$|H(\omega)|^2 = \frac{\frac{10}{4} - \frac{9}{2}\cos(\omega) + 2\cos^2(\omega)}{\frac{5}{4} + 3\cos(\omega) + 2\cos^2(\omega)}$$

- <u>פונקציית התמסורת (pzex.pdf):</u>
  - ראשית נעביר אגפים כדי לקבל צורה סימטרית:

$$y_n + y_{n-1} + \frac{1}{2}y_{n-2} = x_n - \frac{3}{2}x_{n-1} + \frac{1}{2}x_{n-2}$$

  - נכתוב בתור משוואה של סיגנלים:

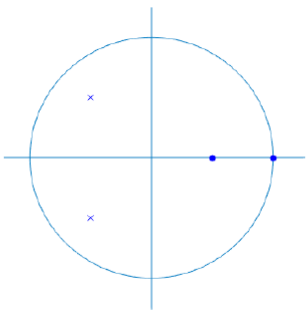
$$y + \hat{z}^{-1}y + \frac{1}{2}\hat{z}^{-2}y = x - \frac{3}{2}\hat{z}^{-1}x + \frac{1}{2}\hat{z}^{-2}x \Leftrightarrow \left(1 + \hat{z}^{-1} + \frac{1}{2}\hat{z}^{-2}\right)y = \left(1 - \frac{3}{2}\hat{z}^{-1} + \frac{1}{2}\hat{z}^{-2}\right)x$$

  - נפעיל התמרת Z ונקבל:

$$\left(1 + z^{-1} + \frac{1}{2}z^{-2}\right)Y(z) = \left(1 - \frac{3}{2}z^{-1} + \frac{1}{2}z^{-2}\right)X(z)$$

$$\Leftrightarrow H(z) = \frac{1 - \frac{3}{2}z^{-1} + \frac{1}{2}z^{-2}}{1 + z^{-1} + \frac{1}{2}z^{-2}} \underset{\cdot z^2}{=} \frac{z^2 - \frac{3}{2}z + \frac{1}{2}}{z^2 + z + \frac{1}{2}} = \frac{(z-1)(z-\frac{1}{2})}{\left(z + \frac{1}{2}(1+i)\right)\left(z + \frac{1}{2}(1-i)\right)}$$

  - כלומר השורשים במונה (האפסים) הם $1, \frac{1}{2}$.
  - השורשים במכנה (הקטבים) הם $-\frac{1}{2}(1+i), -\frac{1}{2}(1-i)$.



2 – אלגוריתמים ויישומים

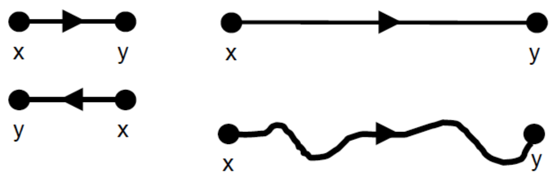
אלגוריתמים

גרפים ב-DSP

גרף: גרף מורכב מנקודות וקווים שמחברים ביניהן. ב-DSP נשתמש בגרפים מכוונים.

טופולוגיה: הרחבה של גיאומטריה. בגיאומטריה מאפשרים טרנספורמציה של הזזה וסיבוב (כמו חפיפת משולשים).

- בטופולוגיה מאפשרים גם **כיווץ והגדלה** – למשל דמיון משולשים. זה אותו משולש עד כדי כיווץ/הגדלה של הצלעות.
- בנוסף, ניתן לעשות טרנספורמציה של **הומיאומורפיזם** – מותר לעשות הכול חוץ מלחתוך או להדביק. כלומר, עיגול וריבוע הם אותו דבר, אבל הספרה 8 ומעגל לא, כי כדי לעבור מ-8 למעגל צריך לחתוך עיגול אחד. זו הפעולה שעושים כדי לעבור ממפה למשל לייצוג על ידי גרף.
- המשמעות של גרף היא טופולוגית בלבד: **לכל קו יש כיוון וצריך להקפיד עליו! אורך המסלול לא משנה.**

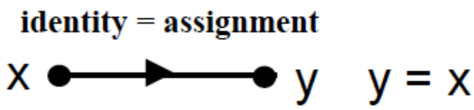


תורת הגרפים במדעי המחשב:

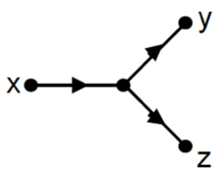
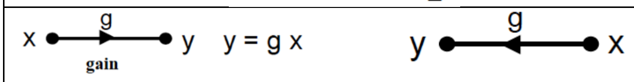
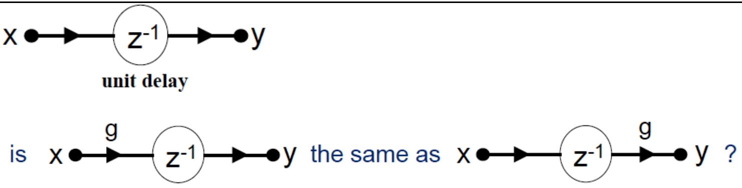
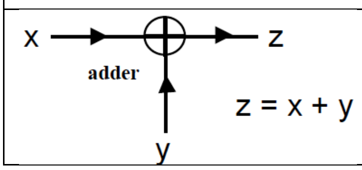
- בעבר השתמשו בתרשימי זרימה (סוג של גרף): שימוש בהם בנוסף לקוד דורש לתחזק שתי מערכות שונות. בתרשימי זרימה הפרדיגמה היא של תכנות אימפרטיבי – שמה דגש על רצפים של פקודות, תוך שימוש ב-goto כדי לעבור ממקום למקום. היום חושבים על תוכניות בחשיבה של הכלה – top-down design. מתאר אלגוריתם בלבד ולא מבנה נתונים ולכן בעייתי. משפט wirth: תכנות = אלגוריתמים + מבני נתונים.
- ב-DSP משתמשים בתרשימי זרימה שמתעדים לא רק את האלגוריתמים, אלא גם את **מבנה הנתונים**, וגם נותנים כלי לשפר את התוכנה. יש כלים שניתן להפעיל על הגרפים הקיימים **שמשפרים את סיבוכיות החישוב**, אבל לא משנים את הנכונות, ומורידים את צריכת הזיכרון.

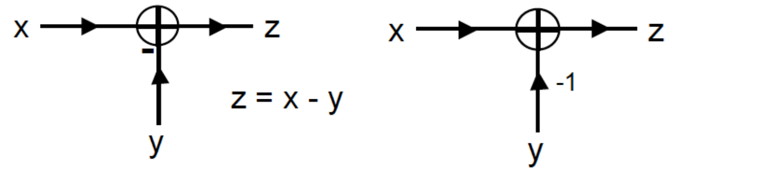
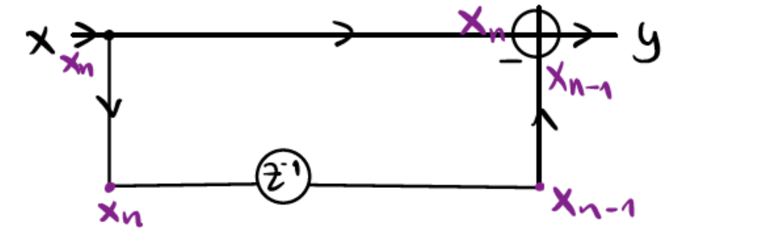
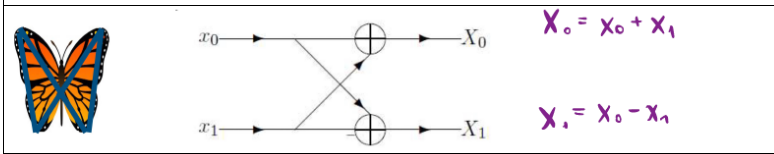
גרפים ב-DSP:

- כל **נקודה** מייצגת **אות**. למשל x מסמל את הסיגנל x, אבל לפעמים נכתוב x_n שזה אומר שבזמן n הערך של הסיגנל הוא x_n. כך למשל במקום לכתוב $\hat{z}^{-1}$ נכתוב x_{n-1}, שזה הסיגנל שבזמן n שווה ל-x_{n-1}.
- **קווים** מייצגים **מערכות לעיבוד אות**. **אות ליד קו מייצג gain.**
- **השמה**: המשמעות של הגרף הבא, הוא שהסיגנל y שווה לסיגנל x. זו השמה, y מקבל את הערך של x לכל אורך הסיגנל.



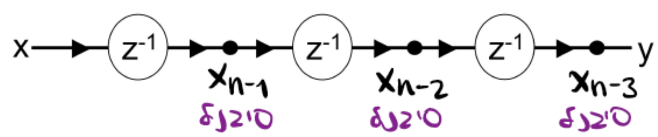
דוגמאות לגרפים:

שם	משמעות	הגרף
T-connector/ splitter	• מתקיים $y = x$ וגם $z = x$. • לא מתחלק חצי-חצי! • יש לו תמיד קלט 1 ו-2 פלטים.	
Gain	• האות g ליד הקו מייצגת את ההגבר. • מתקיים $y = gx$ בכיוון הזה.	
Delay	• מתקיים $y = \hat{z}^{-1}x$. • אפשר לחבר ביחד עם Gain ואזי נקבל $y = g\hat{z}^{-1}x$. • ההרכבה הזאת שקולה בשני הכיוונים, המסננים מתחלפים.	
מחבר	• חיבור: $z_n = x_n + y_n$. • יש לו תמיד 2 קלטים ופלט 1.	

מחסר	• חיסור: $z_n = x_n - y_n$.	
הפרש סופי ראשון	• יש לנו קודם מפצל. • לאחר מכן x_n שמחובר עם מחסר והערך x_{n-1}. • אנחנו מסתכלים על זמן n.	
פרפר	• מתקשר ל-DFT. • זה לא באמת גרף DSP כי x_0, x_1 הם ערכים בציור התדר, לא סיגנלים.	

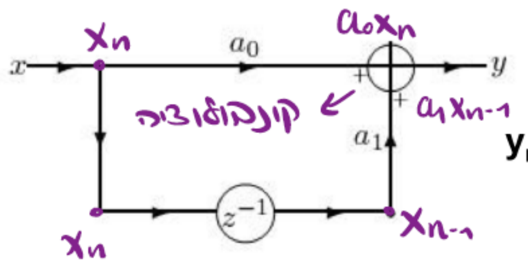
<u>נייצג עם הנקודה memory point:</u> כל פעם שנרצה להראות שצריך זיכרון כדי לבטא משהו, **נצייר במפורש את הנקודה**. נשים נקודת זיכרון אחת **אחרי האופרטור**.

For example, we see that $y = \hat{z}^{-3}\, x$ requires 3 memory points



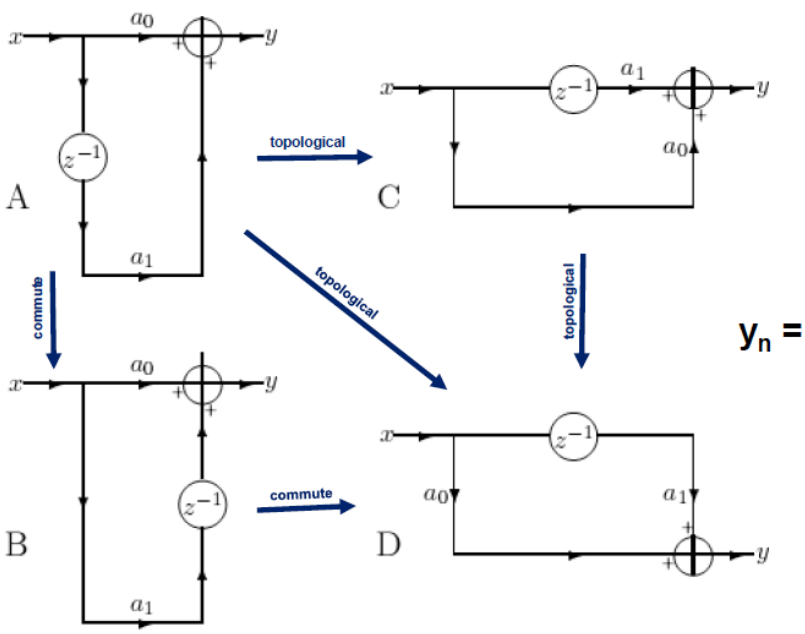
גרף עבור מסנן MA:

<u>בלוק MA בסיס עם שני מקדמים:</u>



$$y_n = a_0\, x_n + a_1\, x_{n-1}$$

ישנן 4 דרכים לכתוב את ה-MA הבסיסי:

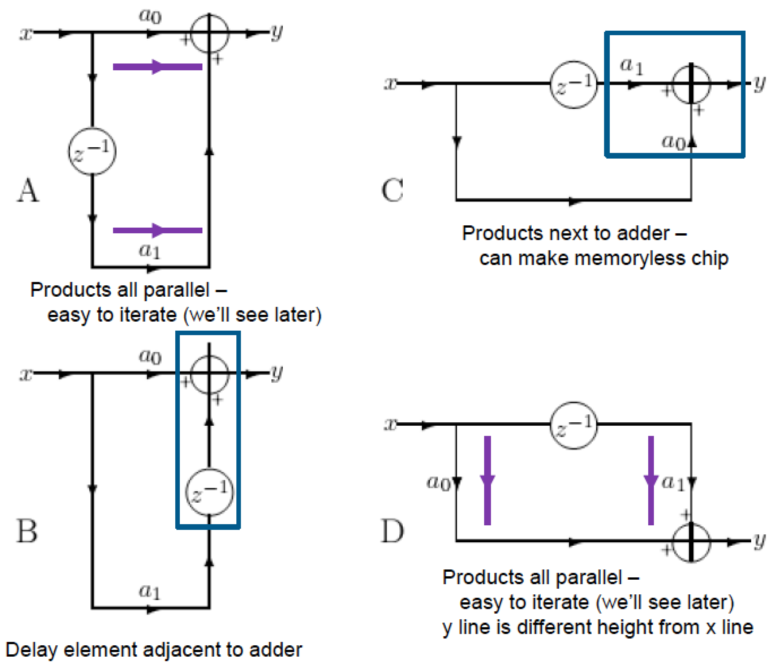


$$y_n = a_0\, x_n + a_1\, x_{n-1}$$

למה משתמשים ב-4 הצורות הללו?

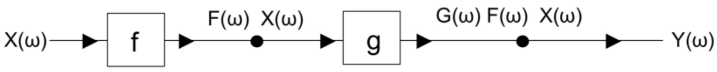
נשים לב:

- כדי לעבור מ-A ל-C דרושים רק שינויים של מתיחות/סיבובים שלא משנים את הטופולוגיה.
- כדי לעבור מ-A ל-B צריך לשנות את הסדר של המסננים.

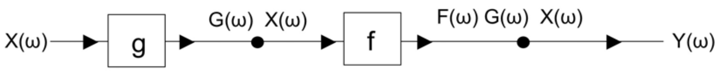


<u>קומוטטיביות:</u> **כל שני מסננים קומוטטיבים – ניתן להחליף את הסדר בגרף של כל שני מסננים.** מסובך להוכיח בציר הזמן, אבל קל להוכיח בציר התדר (או במישור Z). לפי משפט המסנן: $Y(\omega)=H(\omega)X(\omega)$, ולכן עבור שני מסננים f,g זה מתקיים.

Since filters obey Y(ω) = H(ω) X(ω)
two filters – f and g – in series obey Y(ω) = G(ω) F(ω) X(ω)



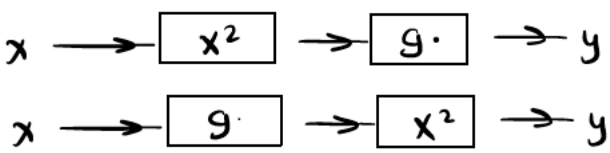
while in the opposite order Y(ω) = F(ω) G(ω) X(ω)



which is the same thing since functions commute!

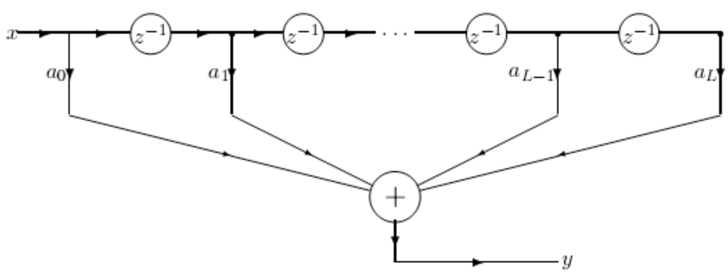
<u>הערה:</u> גרף היחידה יכול להתחלף עם כל מערכת, גם אם היא לא מסנן.

דוגמה למערכת שאינה קומוטטיבית:



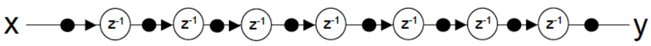
$$y=gx^2$$
$$\neq$$
$$y=(gx)^2=g^2x^2$$

<u>מסנן MA כללי:</u> עבור קונבולוציה כללית היינו רוצים לצייר כך:

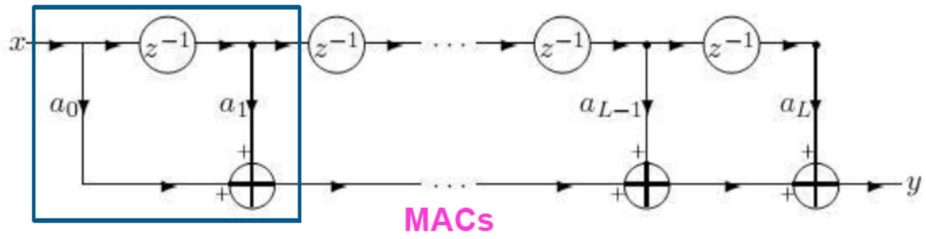


$$y_n=\sum_{l=0}^{L}a_l x_{n-l}$$

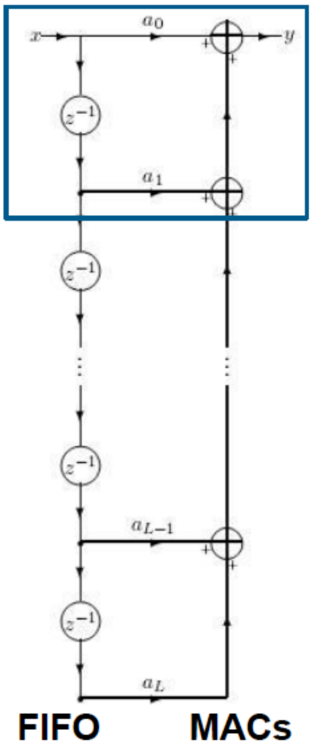
נשתמש במבנה נתונים, שנקרא תור shift register/FIFO. ערכים שנכנסים ראשונים משמאל, יוצאים ראשונים מימין.



x_8	x_7	x_6	x_5	x_4	x_3	x_2	x_1
x_9	x_8	x_7	x_6	x_5	x_4	x_3	x_2
x_{10}	x_9	x_8	x_7	x_6	x_5	x_4	x_3
x_{11}	x_{10}	x_9	x_8	x_7	x_6	x_5	x_4



<u>מימוש ראשון:</u> לקחנו את בלוק D הקודם, ושכפלנו אותו. החישוב הוא אוסף של MAC-ים. לקחנו את **הבלוק הבסיסי D**, והפעלנו אותו **איטרטיבית** עם שרשור L פעמים. צוברים כל פעם מכפלה של שני ערכים. תרשים הזרימה הוא מנקודת המבט של הסיגנל. אנחנו נדגיש את **האלגוריתמים**.



<u>מימוש שני:</u> כאן משתמשים ב**בלוק A**, מבנה הנתונים עדיין FIFO. חיבור הוא קומוטטיבי, לכן תאורטית אין הבדל.

Is there any difference between these two ways?

- the 1st way MACs from the new x backward
  $a_0\, x_n + a_1\, x_{n-1} + \ldots + a_L\, x_{n-L}$
- the 2nd way MACs from the earliest x forward
  $a_L\, x_{n-L} + \ldots + a_1\, x_{n-1} + a_0\, x_n$

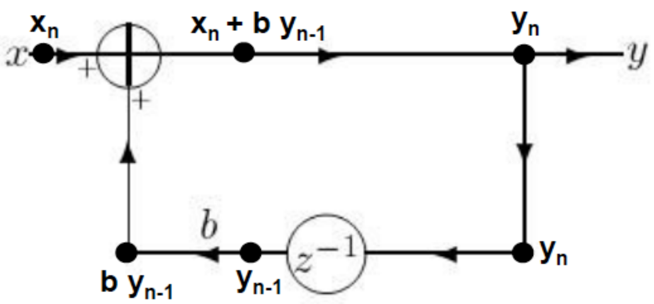
Theoretically there is no difference
(addition is commutative)
but in practice there may be

מכאן ואילך נראה רק גרפים של דברים סיבתיים.

<u>גרף עבור מסנן AR:</u>

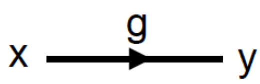
<u>בלוק AR בסיסי:</u>

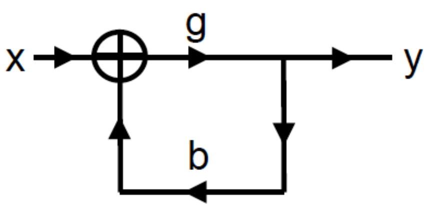
זו פעם ראשונה שאנחנו רואים לולאה. כל לולאה מציינת קוטב בפונקציית התמסורת של המסנן.



$$y_n = x_n + b y_{n-1}$$

הערה: לולאה בלי זיכרון ($\hat{z}^{-1}$) אינה יציבה. יש השהייה של הזמן שלקח למעבר אבל מקבלים הגברה. לא מקבלים אינסוף כי ניתן להגיע רק להספק מקסימלי.





Then instead of y = g x we have y = g(x+by) or y – bgy = gx

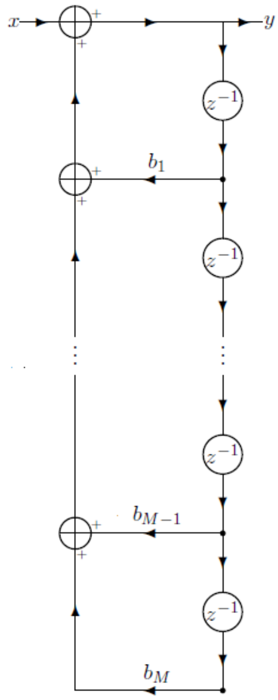
and hence $y = \frac{g}{1-bg}\, x$

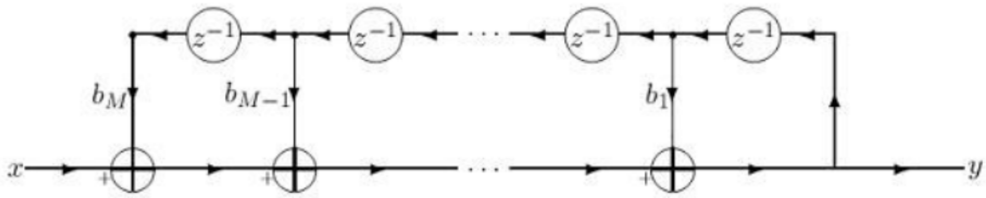
So the feedback increases the amplifier's gain when b < 1/g
but explodes as b → 1/g

<u>מסנן AR כללי:</u>

אפשר לכתוב בשתי צורות:

$$y_n = x_n + \sum_{m=1}^{M} b_m y_{n-m}$$

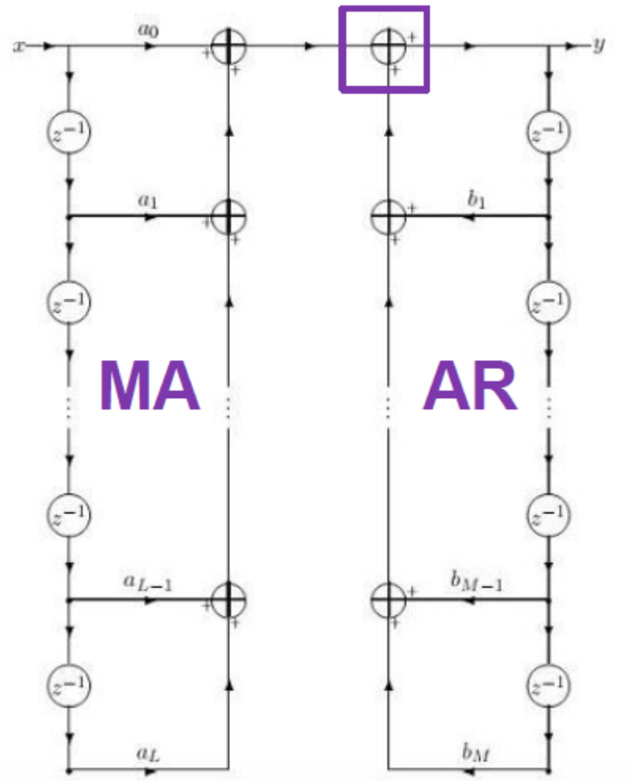




<u>גרף עבור מסנן ARMA:</u>

למסנן ARMA יש חלק של MA וחלק של AR. בשלב ראשון נוכל להשתמש במימוש של MA, ואז במימוש של AR **ולחבר** את שניהם.

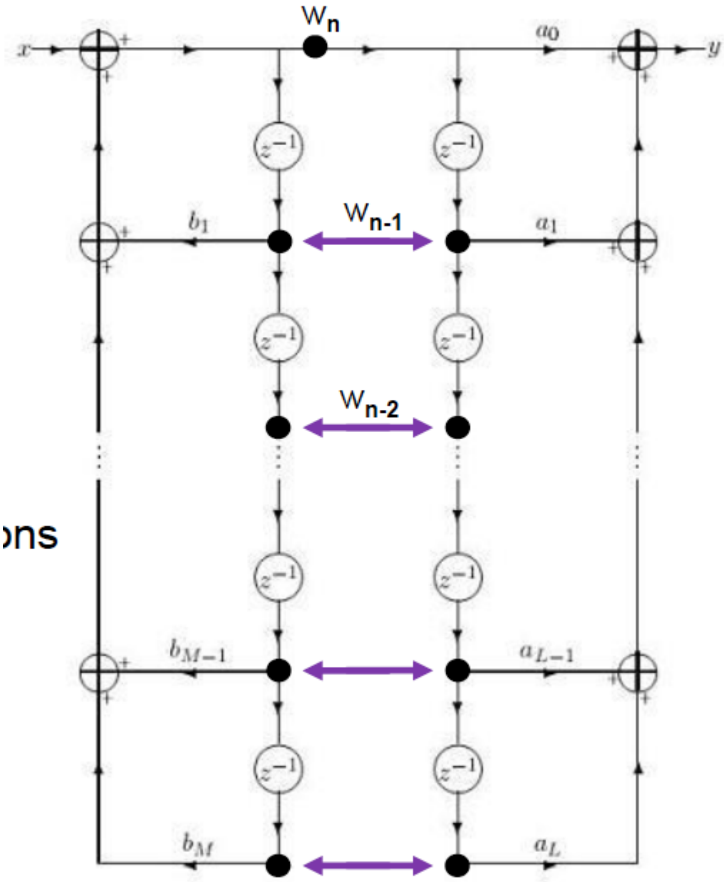
כדי לממש את זה צריך $L+M$ נקודות זיכרון. בה"כ $L=M$ (אם L גדול מ-M אז נרפד את ה-AR עד ש-M שווה L עם מקדמים 0), אז יש לנו $2L$ נקודות זיכרון. נרצה להוריד את סיבוכיות הזיכרון שלנו, ולכן נשתמש בחילוף.

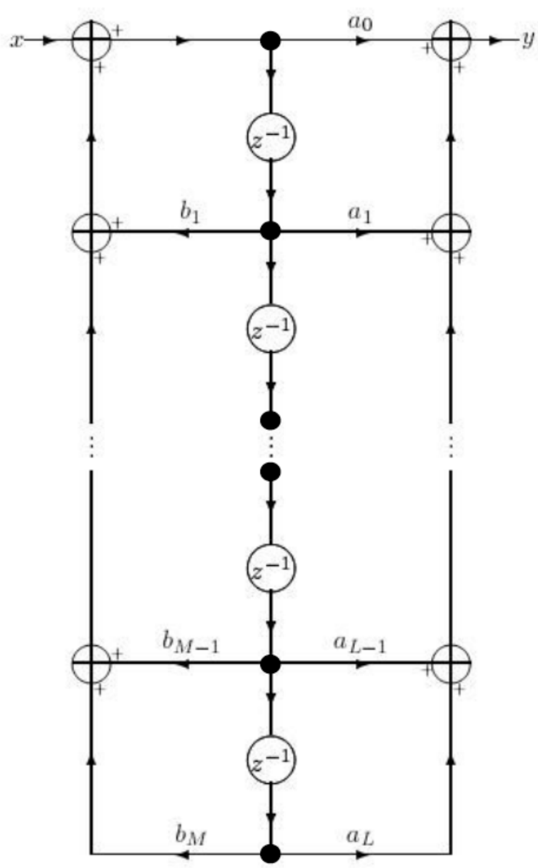


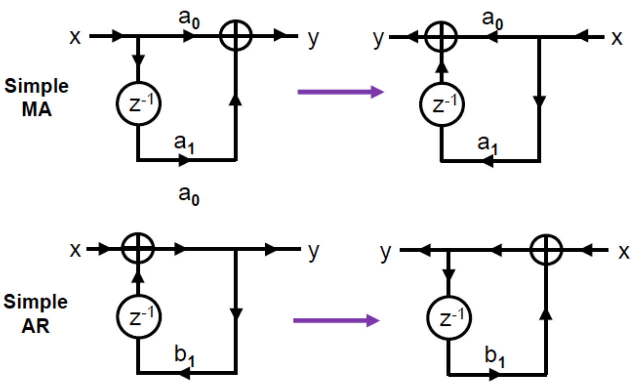
$$y_n = \underbrace{\sum_{l=0}^{L} a_l x_{n-l}}_{\text{MA}} + \underbrace{\sum_{m=1}^{M} b_m y_{n-m}}_{\text{AR}}$$

בעת, נהפוך את הסדר של ה-MA וה-AR. שניהם מסננים, ולמדנו שכל שני מסננים הם קומוטטיבים, אפשר להחליף את הסדר. ה-y ייישאר אותו דבר. כעת הסיגנל שיוצא מה-AR יסומן w.

אנחנו רואים שיש **נקודות שמייצגות את אותו הסיגנל**. כל פעם שנזהה בגרף DSP שתי נקודות שמייצגות את אותו הסיגנל בדיוק, אפשר לאחד אותן. קיבלנו חצי מנקודות הזיכרון!







<u>כלל הטרנספוזיציה:</u> עבור מערכת שיש לה קלט בודד x ופלט בודד y:

- נחליף את הסדר בגרף **בין x ל-y** כך ש-y קלט ו-x פלט.
- נהפוך את **הכיוון** של כל החיצים.
- נחליף כל **מחבר בפיצול**, וכל **פיצול במחבר**.
- נקבל גרף תקין עם אותה המשמעות – **גרף הטרנספוזיציה**.

אלה כללים מכניים שאפשר ללמד תוכנה לעשות אותם. אפשר לקחת גרף מורכב עם הרבה כללים, ולמצוא מערכת שקולה עם פחות זיכרון.

<u>סיכום כל 4 הטרנספורמציות שאפשר לבצע על גרפים בלי לשנות את המשמעות שלהם:</u>

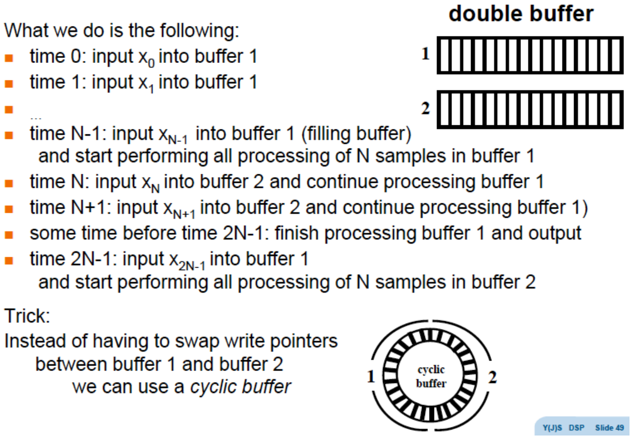
1. אפשר למתוח/לסובב כל עוד שומרים על הטופולוגיה.
2. אפשר להחליף את הסדר של שני מסננים.
3. אפשר להדביק נקודות ולזרוק דברים מיותרים.
4. כלל הטרנספוזיציה.

מערכות זמן אמת ומושגים נוספים

מערכות זמן אמת:

יש לנו מערכת עם קלט שהוא סיגנל ספרתי, נניח כל מילישנייה (1000 פעמים בשניה x חדש), ואנחנו צריכים לספק פלט y גם כל מילישנייה. כלומר צריך לעבד את x_n ולפלוט את y_n לפני שמגיע x_{n+1}. יש שני סוגים של זמן אמת:

- <u>hard real-time</u>: **תמיד** צריך לסיים את החישוב של ה-y החדש תוך מילישנייה או פחות. ב-DSP תמיד נדרוש שזה יהיה כך, כי הולכים לפי ה-worst case. זה נכון גם למסננים MA, AR, ARMA.
- <u>slow real-time</u>: **בממוצע** צריך לסיים את החישוב תוך מילישנייה או פחות. הפעולות הן לא תמיד אותו דבר (יש תנאים, if כלשהו בלוגיקה). כשזה לוקח יותר ממילישנייה, אנחנו שמים את ה-x הבא בצד בתוך באפר, ומובטח שבהמשך יהיה חישוב קצר, ונוכל לשלוף את ה-x מהבאפר. צריך בשביל זה להשתמש **בזיכרון**.



במערכת עם הרבה קלטים ופלטים, נחשוב על real time בשתי דרכים:

1. צריך להספיק לחשב תוצאה על $x_0 x_1 \dots x_{n-1}$ לפני ש-x_n מגיע.
2. double buffer: ממלאים שני באפרים. למלא את הבאפר השני לוקח n מילישניות. צריך לסיים לעבד את הבאפר הראשון, לפני שנסיים לקבל עוד n דגימות בבאפר השני. משתמשים בבאפר ציקלי לשם פשטות.

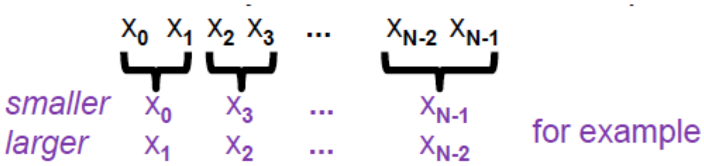
<u>המשפט היסודי של זמן אמת</u>: ניתן לעמוד בזמן אמת כל עוד הסיבוכיות של החישוב היא $O(n)$. כלומר, מכיוון ש-DFT מתבצע ב-$O(n^2)$, הוא לא עומד בזמן אמת!

נראה שקיים אלגוריתם שמבצע DFT בזמן $O(nlogn)$ ואז הוא עמד בזמן אמת עבור מקרים מסוימים.

אלגוריתם ה-FFT:

<u>שאלת חימום 1</u>: נתון אוסף של N מספרים: $x_0, x_1, \dots, x_{N-1}$, ואנחנו רוצים למצוא את המינימום ואת המקסימום שלהם.

- כמה השוואות נצטרך בשביל למצוא מינימום? לפחות $N-1$ השוואות. חייבים לראות את כל המספרים כדי לחשב את המינימום. זה נכון גם עבור חישוב מקסימום. אפשר למצוא גם מינימום וגם מקסימום עם $2N$ השוואות בוודאות.
- כדי לחשב גם מינימום וגם מקסימום ביחד צריך $1.5N$ השוואות. משווים בין כל שני זוגות, ומחשבים את ערך המינימום ביניהם. זה לוקח $\frac{N}{2}$ פעולות. המינימום והמקסימום הגלובליים שייכים לאחד מבין ערכי המינימום הלוקאליים שמצאנו. זה ייקח גם $\frac{N}{2}$. מחשבים גם מקסימום ולכן זה לוקח $1.5N$ השוואות.



- כדי לבצע את האלגוריתם in place נוכל להחליף בין הזוגות לאחר ההשוואה וכך להשתמש באותה כמות זיכרון. גם ב-DSP נחפש אלגוריתמים שפועלים in place.

<u>דסימציה</u>: לחלק סדרה לשתי תתי-סדרות. למשל זוגיים ואי-זוגיים. אנחנו מסתכלים על זוגות קרובים אחד לשני.

<u>שאלת חימום 2</u>: על מנת להכפיל מספרים A, B עם N ספרות כל אחד, צריך לבצע N^2 הכפלות, כמו בבעיית הקונבולוציה. נפתור את הבעיה על ידי חלוקה לשתי סדרות באורך $\frac{N}{2}$.

<u>חלוקה</u>: פיצול סדרה באמצע לשני חלקים: ראשון ושני. אם מדובר בביטים, זה כמו לעשות shift left/right ואז לחבר. נניח עבור A אנחנו לוקחים את 4 הביטים השמאליים, וכופלים אותם ב-2^4 (עושים shift שמאלה) ואז מחברים את החלק שנשאר A_R.

נכתוב את המשוואה של המכפלה בצורה אחרת, ואז נקבל כי המכפלה הראשונה בעלות $\frac{N^2}{4}$, וכך גם האחרות, יש סה"כ 3 כאלה וקיבלנו $\frac{3N^2}{4}$.

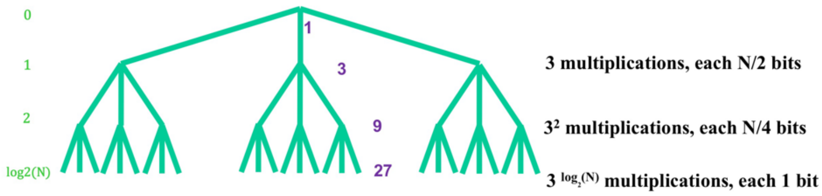
A7 A6 A5 A4 A3 A2 A1 A0 = A_L A_R
B7 B6 B5 A4 B3 B2 B1 B0 = B_L B_R

Now

$$
\begin{aligned}
A &= A_L 2^{\frac{N}{2}} + A_R \\
B &= B_L 2^{\frac{N}{2}} + B_R \\
C &= A_L B_L 2^N + (A_L B_R + A_R B_L) 2^{\frac{N}{2}} + A_R B_R \\
&= A_L B_L (2^N + 2^{\frac{N}{2}}) + (A_L - A_R)(B_R - B_L) 2^{\frac{N}{2}} + A_R B_R (2^{\frac{N}{2}} + 1)
\end{aligned}
$$

נוכל להמשיך לפרק כל חלוקה לחצי, ואז לצמצם זמן חישוב. shift יותר זול מאשר מכפלה. נקבל עץ: זה נקרא אלגוריתם מכפלה לפי תום-קוק.

אם נמצא דרך לחבר את שני חלקי הסדרה בפחות מ-$\frac{N^2}{2}$ נוכל להקטין את זמן הריצה. בנוסף, נוכל לחלק ליותר מ-2 חלקים (כל עוד הם באותו גודל). אלגוריתמים כאלה נקראים radix-2.



Now, $3^{\log_2(N)} = N^{\log_2 3}$ Why is $a^{\log_k(b)} = b^{\log_k a}$?

So, the complexity is $O(N^{\log_2 3}) \approx O(N^{1.585})$
and $O(N^{1.585}) < O(N^2)$ -- the O complexity has been reduced!

דסימציה מול חלוקה:

מחלקים את הסדרה בצורה שונה מבחינת LSB ו-MSB. בדסימציה בודקים את הביט הימני ביותר, ה-LSB אם הוא 0 זה זוגי, ואם 1 זה אי-זוגי. בחלוקה בודקים את הביט השמאלי ביותר, ה-MSB.

$$X_0\ X_1\ X_2\ X_3\ X_4\ X_5\ X_6\ X_7$$

Decimation		Partition	
$X_0\ X_2\ X_4\ X_6$	EVEN	$X_0\ X_1\ X_2\ X_3$	LEFT
$X_1\ X_3\ X_5\ X_7$	ODD	$X_4\ X_5\ X_6\ X_7$	RIGHT
LSB sort		MSB sort	

דסימציה בציר הזמן היא חלוקה בציר התדר ולהפך $DIT = PIF$:

כיוון ראשון: $DIT \Rightarrow PIF$:

נסתכל על דסימציה בזמן: נניח שיש לנו את הסיגנל הבא $x_0 \dots x_7$ שדגמנו בתדירות דגימה f_s. אם נעשה דסימציה, וניקח למשל את הזוגיים: x_0, x_2, x_4, x_6, הזמן בין כל שתי דגימות גדול פי 2 מהזמן שהיה לפני הדסימציה. לכן כאן זה חצי מקצב הדגימה, $\frac{f_s}{2}$.
לפי משפט הדגימה, כל עוד הספקטרום מגיע ל-$f_{Nyquist}$ שהוא פחות מחצי f_s, הדגימה תקינה. אם תדירות הדגימה קטנה בחצי, **מכסים לכל היותר חצי מהספקטרום הקודם**. קיבלנו חלוקה בציר התדר.

כיוון שני: $PIF \Rightarrow DIT$:

נסתכל על חלוקה בתדר: יש לנו סיגנל $x_0 \dots x_{n-1}$ והסתכלנו עליו זמן T. נסתכל רק על החלק השמאלי, עכשיו נסתכל רק על זמן $\frac{T}{2}$.
לפי עקרון אי-הוודאות ($\Delta w \cdot \Delta t > constant$), כיוון שהקטנו את T בחצי, Δw גדל פי 2! הרזולוציה שלנו פחות טובה, עכשיו **נראה אחד כן אחד לא**, וקיבלנו דסימציה בציר הזמן.

ההקשר ההיסטורי: יש שני אלגוריתמים מפורסמים:

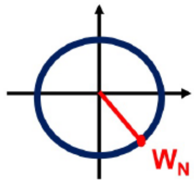
- אלגוריתם של Cooley-Tukey שהוא **דסימציה בזמן**, ומבצע DFT בנפרד על הזוגיים ועל האי-זוגיים.
- אלגוריתם של Sande-Tukey שהוא **חלוקה בתדר**, ומבצע DFT בנפרד על החצי הראשון ועל החצי השני.

FFT

תזכורת על DFT ו-W_N:

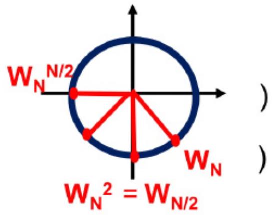
Recall that the DFT is $X_k = \sum_{n=0}^{N-1} x_n \; W_N^{nk}$

where W_N is the N^{th} root of unity $W_N = e^{-i\frac{2\pi}{N}}$



We will need three *trigonometric identities*

1. $W_N^N = 1$ (that's the definition!)
2. $W_N^{N/2} = -1$ $(e^{-i\frac{2\pi N}{N\,2}} = e^{-i\pi} = -1$ or
3. $W_N^2 = W_{N/2}$ $(e^{-i\frac{2\pi}{N}2} = e^{-i\frac{2\pi}{N/2}}$ or



They are *trigonometric* identities

since $W_N = \cos(\frac{2\pi}{N}) - i\sin(\frac{2\pi}{N})$

<u>נבצע radix-2 DFT FFT:</u> ניקח סדרה באורך n, ולחלק אותה לשתי תתי-סדרות באורך $\frac{n}{2}$. כאן נעשה דסימציה לזוגיים ולאי-זוגיים. ניעזר בזהויות, ונוציא את W_N^k מהסכום כי הוא לא תלוי ב-n.

$$X_k = \sum_{n=0}^{N-1} x_n W_N^{nk} = \sum_{n=0}^{\frac{N}{2}-1}\left(\underbrace{x_{2n}W_N^{2nk}}_{\text{זוגיים}} + \underbrace{x_{2n+1}W_N^{(2n+1)k}}_{\text{אי זוגיים}}\right)$$

Now, $W_N^{2nk} \overset{3^{rd}\ \text{identity}}{=} W_{N/2}^{nk}$ and $W_N^{(2n+1)k} = W_N^k\, W_N^{2nk}$ and so we can rewrite:

$$X_k = \underbrace{\sum_{n=0}^{\frac{N}{2}-1} x_{2n} W_{\frac{N}{2}}^{nk}}_{\text{DFT of evens}} + W_N^k \underbrace{\sum_{n=0}^{\frac{N}{2}-1} x_{2n+1} W_{\frac{N}{2}}^{nk}}_{\text{DFT of odds}}$$

כלומר קיבלנו:

$$X_k = \sum_{n=0}^{\frac{N}{2}-1} \overset{\text{Even}}{x_n^E}\, W_{\frac{N}{2}}^{nk} + W_N^k \sum_{n=0}^{\frac{N}{2}-1} \overset{\text{Odd}}{x_n^O}\, W_{\frac{N}{2}}^{nk}$$

נרצה להשתמש בכך שדסימציה בזמן היא **חלוקה בתדר**. נניח ש-k הוא מספר בין 0 ל-$\frac{N}{2}$ (כלומר בחלק השמאלי של החלוקה). האיבר המקביל ל-X_k בחלק הימני הוא $X_k + \frac{N}{2}$.

$$X_k = \sum_{n=0}^{N-1} x_n W_N^{nk}$$

$$X_{k+\frac{N}{2}} = \sum_{n=0}^{N-1} x_n W_N^{nk}\, W_N^{\frac{Nn}{2}}$$

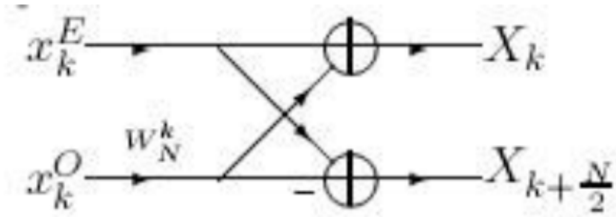
$$= \sum_{n=0}^{N-1} x_n W_N^{nk}\,(-1)^n \xleftarrow{} \begin{array}{l}2^{nd}\ \text{identity}\\ W_N^{N/2} = -1\end{array}$$

כלומר בהינתן החצי הראשון, נדע את המכפלות בחצי השני, ולכן הורדנו את המכפלות בפקטור 2. נוכל לבצע את זה in place. כלומר סה"כ אם נחבר את זה:

$$X_{k+\frac{N}{2}} = \sum_{n=0}^{\frac{N}{2}-1} x_n^E\, W_{\frac{N}{2}}^{nk} - W_N^k \sum_{n=0}^{\frac{N}{2}-1} x_n^O\, W_{\frac{N}{2}}^{nk}$$

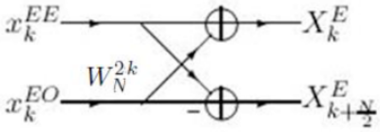
אם נצייר את זה בגרף נקבל גרף פרפר עם דבק (twiddle):

x_k^E is the DFT sum
don't confuse it with x_n^E !
(it should have an intermediate sized x …)



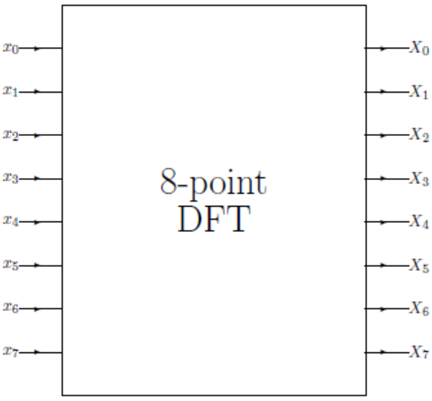
כעת צריך לחשב DFT של הזוגיים ושל האי-זוגיים. נוכל לפצל את הזוגיים גם לשתי תתי-סדרות:

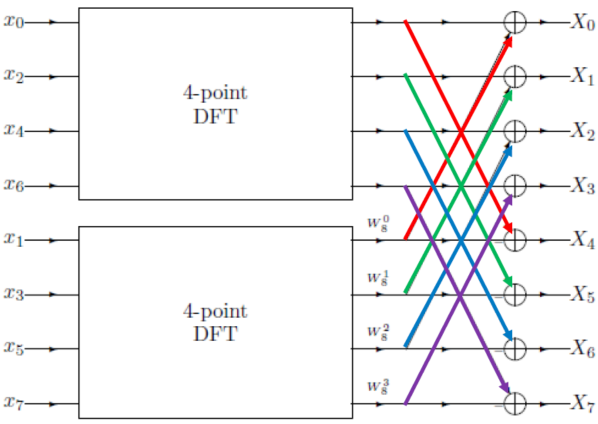
Assuming that N is a power of 2
we continue decimating until we get to the basic N=2 butterfly
Note that since $W_{N/2} = W_N^2$ we can draw this



and we only have to keep one table of W_N^k

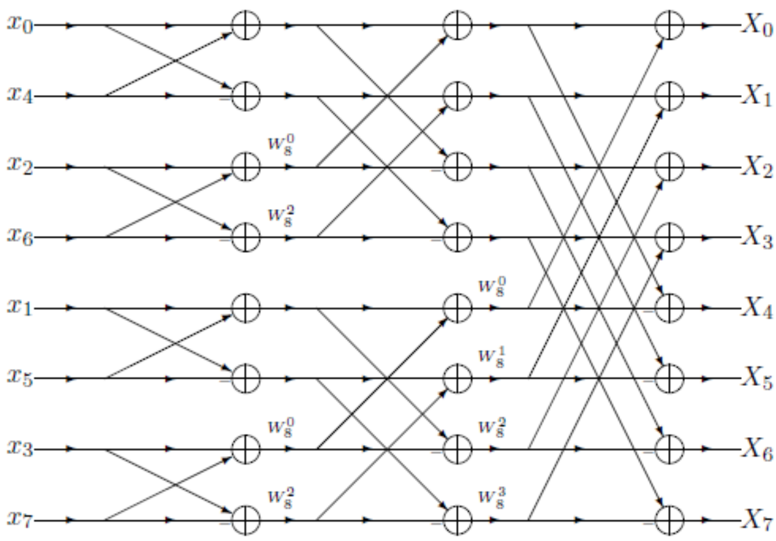
<u>דוגמה של DFT עבור $N = 8$:</u>



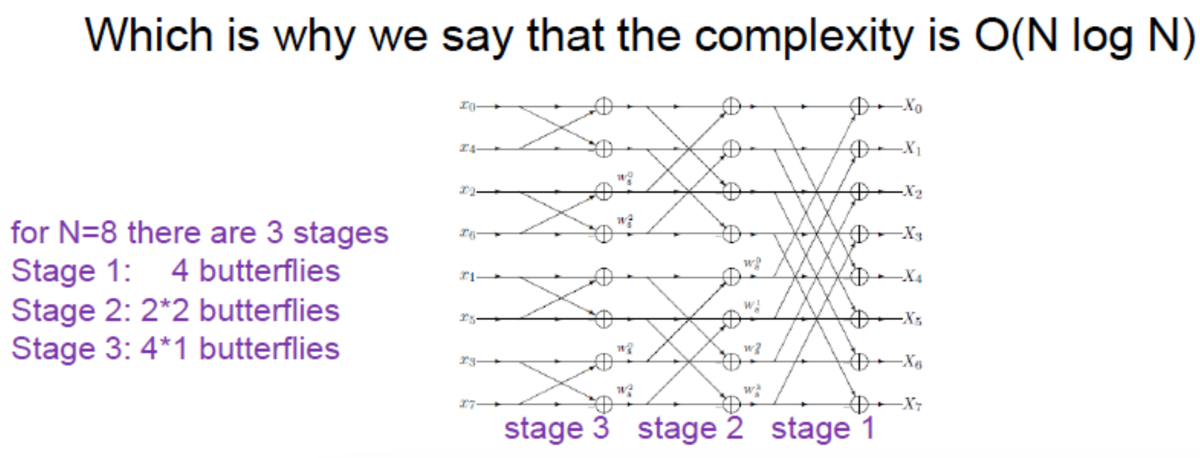


$W_4^0 = W_8^0$
$W_4^1 = W_8^2$

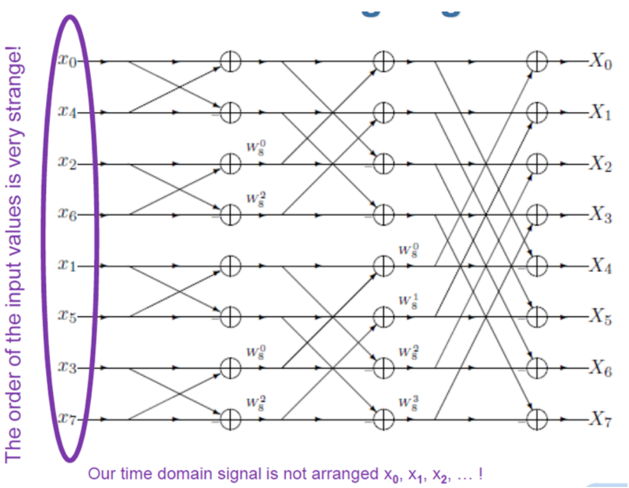
נשים לב שבאמצע לא השתמשנו ב-W_4 אלא גם ב-W_8, כי אנחנו עובדים עם טבלה אחת נתונה של W_N^k.



סיבוכיות: בכל פרפר יש מכפלה אחת. לכן נבצע מכפלות כמספר הפרפרים. יש $\log_2 N$ שכבות של פרפרים. בכל פרפר יש מכפלה אחת מרוכבת ו-2 חיבורים (חיבור אחד וחיסור). יש $\frac{N}{2}$ פרפרים בכל שכבה. לכן סה"כ יש $\frac{N}{2}\log_2 N$ מכפלות מרוכבות ו-$Nlog_2N$ חיבורים מרוכבים. כל מכפלה ממשית יכולה להיות ממומשת ע"י 3 מכפלות ו-5 חיבורים.
הערה: אם שואלים במבחן כמה מכפלות אמיתיות יש – זה אומר בכלי לכלול את ההכפלה ב-1 $W_N^0 =$.



Bit reversal: כאשר עוברים מהשלב האחרון לשלב לפני אחרון, עושים דסימציה in place. נרצה לחשב את ה-x הגדולים בצורה מהירה יותר. לכן לא נחשב אותם מהקטן לגדול, נשתמש ב-shift ציקלי.

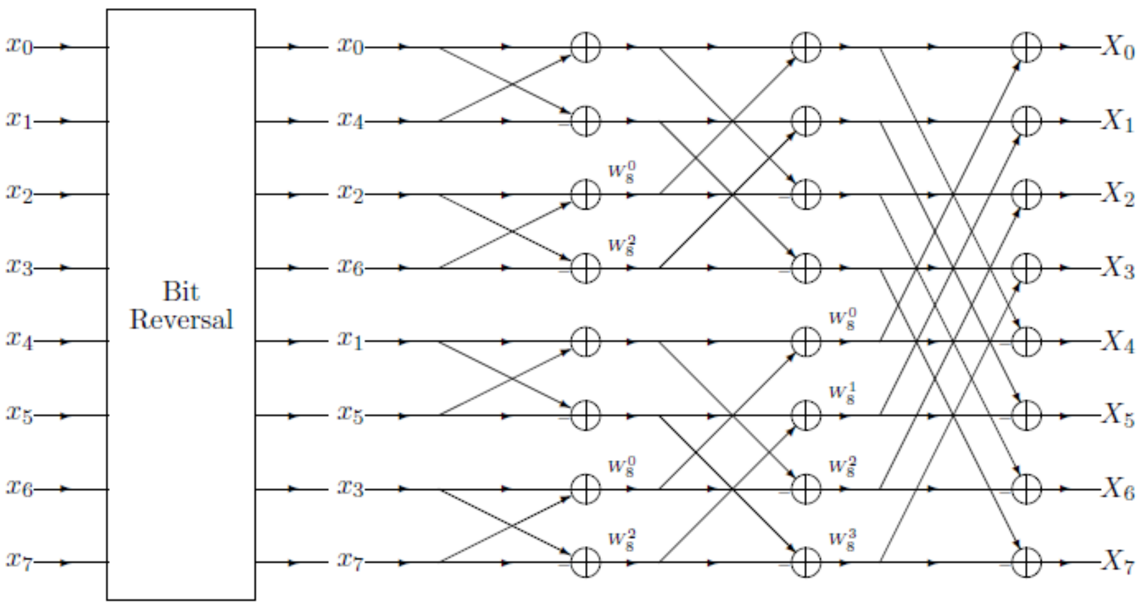


Let's see if we can figure it out! Here for **N=16** IN-PLACE!

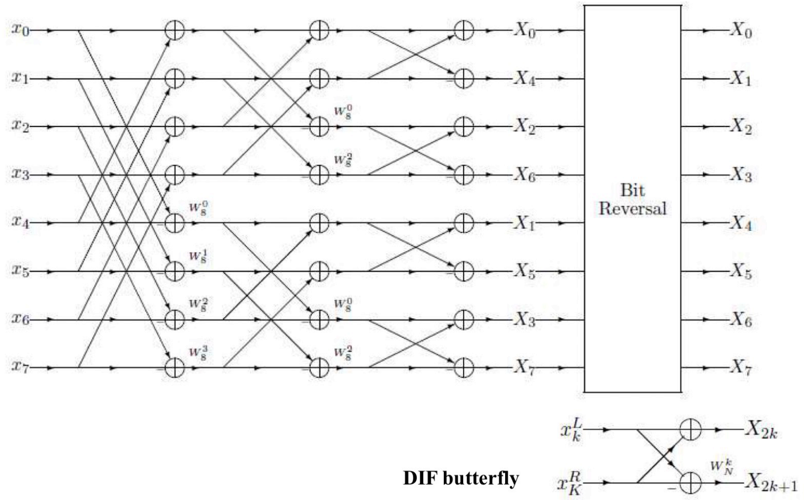
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
0	2	4	6	8	10	12	14	1	3	5	7	9	11	13	15
0000	0010	0100	0110	1000	1010	1100	1110	0001	0011	0101	0111	1001	1011	1101	1111
0	4	8	12	2	6	10	14	1	5	9	13	3	7	11	15
0000	0100	1000	1100	0010	0110	1010	1110	0001	0101	1001	1101	0011	0111	1011	1111
0	8	4	12	2	10	6	14	1	9	5	13	3	11	7	15
0000	1000	0100	1100	0010	1010	0110	1110	0001	1001	0101	1101	0011	1011	0111	1111

1st transition is cyclic left shift
2nd transition freezes the LSB and cyclic left shifts the rest
3rd transition freezes the 2 LSBs and cyclic left shifts (swaps) the rest
Altogether we find **abcd → bcda → cdba → dcba**

נקבל:



אפשר גם להוכיח את זה על ידי דסימציה בתדר, ואז להשתמש בתוצאה שעשינו עם משפט הטרנספוזיציה:



:FIFO FFT

כאשר אנחנו רוצים לחשב FFT של N מספרים עם shift ביחידת זמן של חלון מסוים, כמו שעשינו עם חישוב של ממוצע מספרים. לא תמיד אפשר להשתמש ב-MA. ניתן להשתמש ב-MA עם מקדמים מסוימים. משתמשים כל פעם בתוצאות הקודמות, וה-DFT בייצוג מטריציוני הוא עם מקדמים של טור גיאומטרי.

In other words, $[x_0, x_1, x_2, \ldots x_{N-1}], [x_1, x_2, x_3, \ldots x_N], [x_2, x_3, x_4, \ldots x_{N+1}], \ldots$

You might already know the trick
on how to update a simple moving average

$A_1 = x_0 + x_1 + x_2 + \ldots + x_{N-1}$
$A_2 = x_1 + x_2 + x_3 + \ldots + x_N \quad = A_1 - x_0 + x_N$
$A_3 = x_2 + x_3 + x_4 + \ldots + x_{N+1} \quad = A_2 - x_1 + x_{N+1}$

This is implemented by maintaining a FIFO of length N
adding the new input and subtracting the one to be discarded

A similar trick works for weighted MA
if the weights form a geometric progression

$A_1 = x_0 + q\, x_1 + q^2\, x_2 + \ldots + q^{N-1}\, x_{N-1}$
$A_2 = x_1 + q\, x_2 + q^2\, x_3 + \ldots + q^{N-1}\, x_N \quad = (A_1 - x_0)/q + q^{N-1}\, x_N$
$A_3 = x_2 + q\, x_3 + q^2\, x_4 + \ldots + q^{N-1}\, x_{N+1} \quad = (A_2 - x_1)/q + q^{N-1}\, x_{N+1}$

The DFT is just such a weighted moving average, with $q = W_N^k$
but when moving from time to time we shouldn't *reset the clock*!

So

$$\begin{aligned} X_{k_m} &= \sum_{n=0}^{N-1} x_{m+n} W_N^{(m+n)k} \\ X_{k_{m+1}} &= \sum_{n=0}^{N-1} x_{m+1+n} W_N^{(m+1+n)k} \\ &= \sum_{n=1}^{N} x_{m+n} W_N^{(m+n)k} \\ &= \sum_{n=0}^{N-1} x_{m+n} W_N^{(m+n)k} - x_m W_N^{mk} + x_{m+N} W_N^{(m+N)k} \end{aligned}$$

and hence $X_{k_{m+1}} = X_{k_m} + (x_{m+N} - x_m) W_N^{mk}$

requiring only 2 complex additions and one multiplication per k
or altogether N multiplications and 2N additions!

נושאים נוספים (שקפים 88-99): אלגוריתם גרצל, FFT842, שערוך ספקטרלי, הכפלה באמצעות FFT.

מעבדי DSP

מעבד DSP: מעבד ייעודי שעושים בו שימוש באפליקציות של עיבוד אותות. הוא בעל התכונות הבאות: קטן פיזית, חסכוני באנרגיה, זול יחסית.

במעבד אותות צריך לבצע פעולות MAC בצורה יעילה בשביל לבצע פעולות נפוצות כמו MA, AR, קורלציה, חישוב אנרגיה, FFT וכו. נרצה להגיע למצב בו פעולת **MAC** מתבצעת **במחזור שעון אחד**. מה שקובע את מחזור השעון הוא הגביש שנמצא במעבד. נגיד שכל מחזור שעון הוא בערך פקודת מכונה אחת – opcode.

נתחיל מהמודל הגנרי הפשוט של ארכיטקטורת CPU, ונראה אילו אלמנטים צריך להוסיף. כל דבר שלא תורם לפעולות DSP נוציא, ונוסיף דברים שאין ב-CPU סטנדרטי כדי שנעשה MAC בצורה יעילה.

המודל של מעבד רגיל (CPU):

CPU פשוט:

- יש רכיב ALU לביצוע פעולות אריתמטיות, ומשתמשים ברגיסטרים ייעודיים לביצוע הפעולות הללו, אבל זה גורם לכך שנצטרך לעשות load ו-store מזיכרון חיצוני.
- program counter – הכתובת בזיכרון של הפקודה הבאה לביצוע.
- stack pointer – רגיסטר ששומר את המיקום במחסנית – לאן לחזור בקוד למשל אחרי יציאה מפונקציה.
- accumulator – יש לו ביטים נוספים שמאפשרים לבצע חיבור בלי גלישה, עבור מספרים גדולים. רגיסטר מיוחד ל-DSP.

פעולת MAC:

לולאה חיצונית על כל הזמנים. ולולאה פנימית שצוברת לתוך y_n את המכפלות של $a_i x_j$.

בתכנות רגיל היינו כותבים $a[i]$ אבל זה בעצם חישוב של $a[0] + i \cdot word\_len$.

The basic *MAC loop* in high level languages is (assuming that a and x are in static buffers)

```
loop over all times n
    initialize y_n ← 0
    loop over i from 1 to number of coefficients (L)
        y_n ← y_n + a_i * x_j     (j somehow related to i)
    output y_n
```

For *energy* and *correlation* i and j increase together
For *convolution* i increases and j decreases

בשפת אסמבלי נממש את הרעיון של להשתמש במצביעים, וכל פעם נעדכן את מיקום המצביע (נקדם אותו בגודל מילה), במקום לעשות את הגישה למערך כל פעם.

נניח שיש לנו רגיסטרים a, x שבפנים יש מצביעים. אנחנו מעדכן את המצביע שיש ב-a, ואז הולך לכתובת שאליו הוא מצביע ולוקח את הערך. כנ"ל עבור x. נעשה את המכפלה, ואת החיבור, וכך בלולאה.

```
loop over all times n
   clear y register
   set number-of-iterations to L
   loop
      decrement number-of-iterations
      if number-of-iterations = 0 then terminate loop
      update a pointer
      load contents of memory addressed by a into register a
      update x pointer
      load contents of memory addressed by x into register x
      multiply z ← a * x    (register operation!)
      increment y ← y + z   (register operation!)
   store y
```

אנחנו נצלול לתוך הלולאה הפנימית, ונעשה אופטימיזציה על מה שקורה שם. הלולאה החיצונית קורית גם ככה. כדי לטפל בלולאה החיצונית, יש רגיסטר loop counter מיוחד שמאפשר לנו לבצע **zero-overhead loop**. הרגיסטר הוא עבור המונה של הלולאה, הוא אוטומטית self-decrementing ואם הוא מגיע ל-0 זה יוצא מה-loop. זה נפוץ בהרבה CPU לכן זה לא מאוד מיוחד עבור DSP.

cycle counting: כמה מחזורי שעון יקח ל-CPU שאינו DSP, לעשות MAC אחד, לולאה אחת פנימית? זה יקח לכל הפחות 10.

בפועל כל פעולה כזאת לא מתבצעת במחזור שעון בודד באמת, יש עוד פעולות שמתחבאות מאחורי הקלעים. נרצה להוריד 9 פעולות ולהישאר עם מחזור שעון בודד!

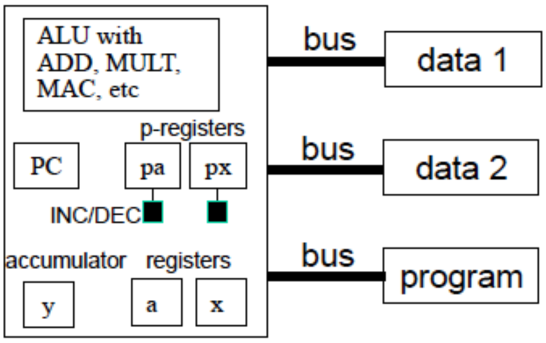
So the clocks operations *inside* the outer loop look something like this:

1. Update pointer to a_i
2. Update pointer to x_j
3. LOAD contents of a_i into register a
4. LOAD contents of x_j into register x
5. Fetch operation (MULT)
6. Decode operation (MULT)
7. MULT a*x with result in register z (MULT really takes >1 clock!)
8. Fetch operation (INC)
9. Decode operation (INC)
10. INC register y by contents of register z

So, it takes at least 10 cycles to perform each MAC using a ***regular*** CPU

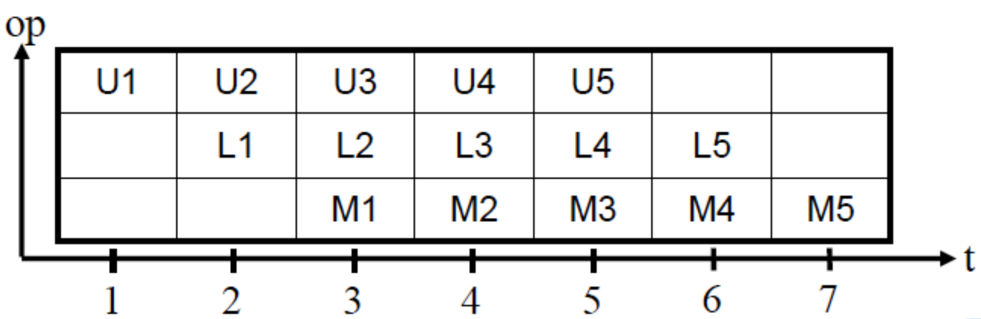
<u>נבנים שיפורים:</u>

- <u>צעד ראשון – נוסיף opcode:</u> נוסיף פעולה בשם MAC. כעת כל 3 הפעולות האחרונות נעלמו, ונשארנו עם 7 בלבד.
- <u>צעד שני – הוספת רגיסטרים:</u> נוסיף שני רגיסטרים חדשים שיאפשרו לנו לבצע פעולות במקביל (את פעולות 1 ו-2). אלו רגיסטרים של עדכון מצביע. רגיסטרים pa, px שיש להם יחידה קטנה של inc/dec כך שהם יודעים לקדם באורך מילה את הכתובת של המצביע. הם עובדים בלי להעסיק את ה-ALU. עכשיו נשארנו עם 6 פקודות.
- <u>צעד שלישי – עוד bus עבור data:</u> נוסיף עוד bus, שמחבר אותנו לעוד bank של זיכרון. נוסיף עוד צ'יפ של זיכרון, כדי למקבל את ה-LOAD בו-זמנית (פעולות 3 ו-4). יש לנו שני banks נפרדים של זיכרון. נישאר עם 5 פקודות.
- <u>צעד רביעי – עוד bus עבור program:</u>
  - ארכיטקטורת Harvard: היה CPU, program memory (לא ניתן לשנות בזמן ריצה, רק קריאה) ובנפרד היה data memory (קריאה וכתיבה).
  - ארכיטקטורת פון-נוימן: הוא שילב את שני הזכרונות הנפרדים לזיכרון אחד. **כל המחשבים עובדים היום עם הארכיטקטורה הזאת, חוץ מ-DSP, שעובד עם Harvard.**
  - לכן, נוסיף עוד צ'יפ של זיכרון עבור ה-program, ואז נוכל לבצע גם את ה-fetch במקביל, ובנוסף להיפטר גם מפקודת ה-decode. עכשיו נישאר עם 3 פקודות!




```
1. Update pointer to a_i || Update pointer to x_j
2. LOAD a_i into a || LOAD x_j into x
3. MAC a*x with incremented to accumulator y
```

אם נרצה למקבל בין 1 ל-2 זה נראה לא אפשרי. אבל אם נסתכל על **pipeline** בעומק 3:



יש ברגע חורים ב-pipeline. נרצה למלא אותם כדי לבצע את 3 הפעולות במקביל. נוכל לבצע פעולות בסדר שונה ובסופו של דבר מבצעים את כל הפעולות במקביל. pipeline עולה ב-delay של כל פעולה ולכן לא מבצעים את כל ה-10, עדיין נצטמצם ל-3.

<u>תכונות DSP:</u>

<u>Saturation Arithmetic:</u> למה אפשר להכפיל ולחבר שני מספרים ואין שגיאה? מספרים גדולים מ-MAXINT הופכים ל-MAXINT, וכנ"ל עבור MAXINT-. יש שגיאה, אבל לא נוראית כמו ב-CPU רגיל שבו היינו מקבלים מספר שלילי.

<u>מה אין ב-DSP:</u> אין opcode לחלוקה, אין פעולה של שורש, אין cache כי זה עולה המון ללכת לזיכרון חיצוני. לכן למעבדי DSP יש המון זיכרון פנימי.

<u>מה עושים בלי חלוקה (Automatic gain control):</u> משהו שאם מדברים חזק מנמיך את הקול, ולהיפך, על מנת שעוצמת הקול תהיה קבועה. כדי לדעת כמה gain לשים צריך לדעת מה האנרגיה של הסיגנל. ניתן לחשב בקלות סכום ריבועים ע"י MAC אבל אחר כך לבצע חלוקה ב-gain שמצאנו כדי שזה יישאר קבוע כל הזמן.

יש opcode שנותן ניחוש טוב ל-$\frac{1}{x}$ על סמך seed כלשהו, ואז בלולאה מתכנסים לתשובה. עבור חלוקה מלאה מבצעים חלוקה על סמך Goldschmidt – מנחשים מונה ומכנה. מאפשר לבצע חילוק ארוך בצורה יעילה. מאפשר חלוקה ב-3 פעולות בצורה רקורסיבית.

```
N' ← N
D' ← D
Loop
    y ← 2 − D'
    N' ← N' * y
    D' ← D' * y
```

<u>מה עושים בלי שורש:</u> אפשר להשתמש באלגוריתם של **ניוטון-רפסון**. ב-DSP צריך לרוב שורש בשביל לחשב **סכום פיתגורני**. לכן יש את אלגוריתם **מולר-מוריסון**.

בנוסף, איך מחשבים סינוס וקוסינוס, אלגוריתם CORDIC וכו'

יישומים

Speech Compression

Speech is a wave traveling through space
at any given point in space it is a signal in time

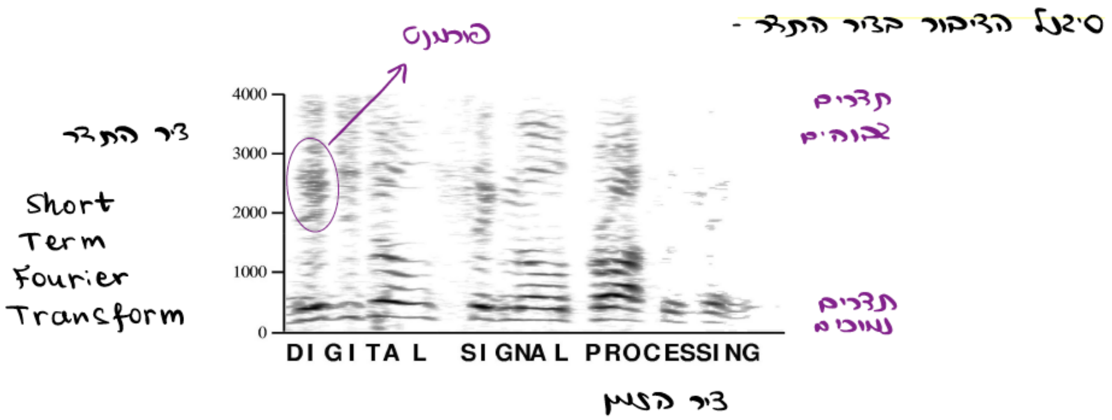
The speech values are pressure differences (or molecule velocities)

There are many reasons to process speech, for example

- speech storage / communications
- speech compression (coding)
- speed changing, lip sync
- text to speech (speech synthesis)
- speech to text (speech recognition)
- translating telephone
- speech control (commands)
- speaker recognition (forensic, access control, spotting, …)
- language recognition, speech polygraph, …
- voice fonts

<u>מבוא (פונטיקה – בלשנות ☺):</u>

<u>דיבור:</u> שינוי בלחץ האוויר על ידי הפקת צלילים מסוימים, סיגנל שעובר באוויר. קשה מאוד לזהות בציר הזמן את המילים מתוך האקוסטיקה. ניתן להבחין כי בתוך התנועות יש הרבה אנרגיה, ובעיצורים פחות. ניתן להסתכל על ציר הזמן וציר התדר בו זמנית באמצעות **סונוגרמה**. הקווים האופקיים נקראים **פורמנטים**.



<u>פונמה:</u> היחידה האקוסטית הקטנה ביותר, שיכולה לשנות משמעות. לכל שפה יש פונמות משלה. לכן ישנן פונמות מסוימות שבשפות אחרות על אף שניתן לשמוע אותן, קשה להגות אותן (הן לא נלמדות במערכת הפונולוגית של השפה, לא ניתן להבחין ביניהן לפונמות אחרות): בערבית /p/ הופך ל-/b/, בעברית /ð/ הופך ל-/z/ וכו'. ישנם סוגים שונים של פונמות: עיצורים (סותמים, חוככים, מקורבים),

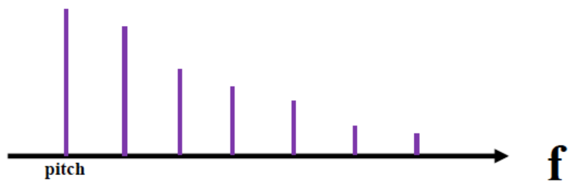
<u>BSP – (biological signal processing):</u> שתי מערכות שאינן מותאמות זו לזו:

1. <u>מערכת שמייצרת את הדיבור (פונטיקה חיתוכית):</u> ריאות, מיתרי הקול, חלל הפה, אזור ברוקה במוח וכו'.
2. <u>מערכת שמזהה את הדיבור (פונטיקה תפישתית):</u> האוזן החיצונית והפנימית, אזור ורניקה במוח וכו'.

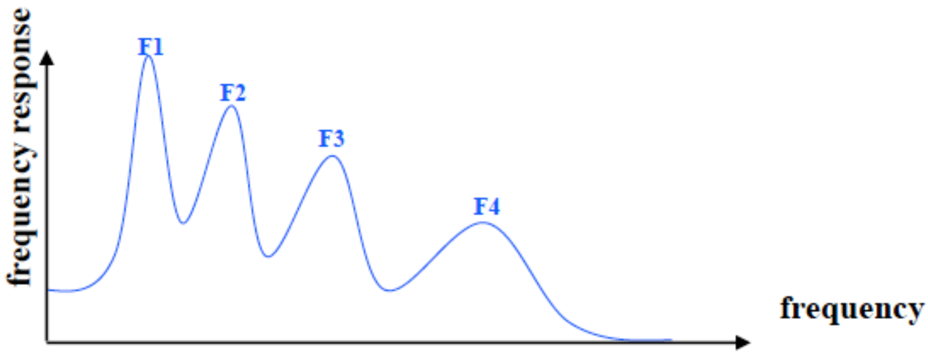
<u>מיתרי הקול:</u> מטרתם לשחרר פולסים של אוויר בקצב מסוים. הקצב של הפולסים האלה נקרא ה-**pitch**, התדר הבסיסי של הקול (נע בין 50 ל-250 הרץ). אפשר לפתוח את מיתרי הקול ולתת לאוויר לעבור. חלק מההגאים משתמשים במיתרי הקול (קוליים – **voiced**), וחלק לא (בלתי קוליים – unvoiced). בלחישה לא משתמשים במיתרי הקול והכול בלתי קולי.

<u>איברי השמיעה:</u> **האוזן החיצונית** מטרתה להגביר את הצלילים שנשמעים, דרך תעלת האוזן הצלילים מגיעים לתוף שזז עם שינויי הלחץ. התוף מחובר לעצם הפטיש, והוא מאוד רגיש לשינויים בעור התוף, שמבצע שינויים גדולים. באוזן הפנימית ישנה עצם **השבלול**, הלשונית שלה מחוברת לשערות קטנות, כל אחת מחוברת לעצב (נוירון) – שמוציא **פולס חשמלי**. אם נכנס צליל עם תדר מוגדר (סינוס), אזור מסוים של הלשונית נכנס לויברציות והנוירונים ישלחו את הפולס המתאים.

כאשר יש צליל קולי – **הסיגנל הוא מחזורי**, ניתן לזהות ע"י ספקטרום מקלות (טור פורייה ולא רק התמרת פורייה), וע"י אוטוקורלציה עם פיקים. בציר התדר נראה תדר בסיסי וכפולות (הרמוניות) שלו בציר הזמן. זה לא סינוס, אלא פולסים. כאשר הוא בלתי קולי – הסיגנל אינו מחזורי, וישנו **רעש לבן**.



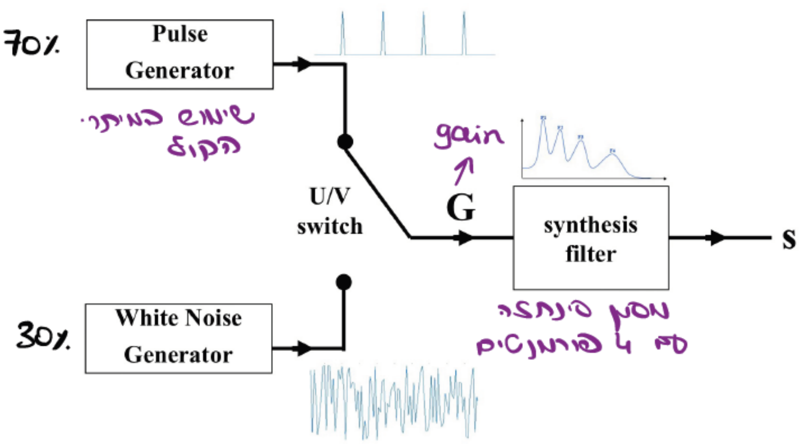
חלל הפה מגביר תדרים מסוימים על פני אחרים, לכן הוא **מסנן AR**, כי אין לו אפסים (שימוש באף כדי להוציא הגאים). הוא שם קטבים במקומות שונים. מקובל להגיד שהפה יכול לעשות עד 4 קטבים – תדרים שהוא יכול להגביר. תדרים אלו נקראים **פורמנטים**. הפורמט F_0 הוא ה-pitch.



לפי תיאוריה אחת, אם נדע את הפורמנטים, נדע לזהות את **התנועות** השונות. יש דאטא של Peterson-Barney שבו הסתכלו רק על הפורמנטים F_1, F_2 וידעו לזהות לפי משולש את התנועות השונות. זה רעיון ישן.

<u>מודל LPC:</u>

מודל בסיסי להפקת דיבור, מחקה את המערכת הביולוגית. הומצא על ידי Bishnu Atal בשנות ה-60. המתג של U/V בוחר בין unvoiced/voiced, עובר דרך gain שיכול להגביר או להנמיך את האנרגיה, ומסנן AR שמעביר את כל הפורמנטים. מניחים שאין אפסים במסנן, ושאין רעשים שלא קיימים בשפות מערביות (כמו קליקים).



<u>**דחיסת דיבור:**</u> נניח שהדיבור הוא בתדרים של עד 4000 הרץ (טלפון קלאסי). אם נדגום לפי משפט הדגימה, נצטרך לדגום לפחות 8000 דגימות בשנייה.

כמה ביטים צריך בכל דגימה: 8 ביטים בלבד, נשמע רעש בנוסף לדיבור, כי אנחנו כופים התאמה לרמות מסוימות. מתי הרעש הזה מפריע לנו? בסביבות ה-15 ביטים זה כבר בסדר. נניח **16 ביטים לדגימה**. נקבל **128Kb לשנייה**, רק כדי להעביר דיבור.

עם LPC אפשר יותר טוב:

- נניח שאנחנו מעבירים 25 פונמות בשנייה.
- pulse generator – צריך לדעת מה קצב הפולסים, ה-pitch שהוא נע בין 50 ל-250 הרץ, כלומר בערך 8 ביטים לקידוד.
- עבור מצב ה-switch – צריך ביט אחד. נשתמש ב-0 pitch עבור רעש לבן, וכל מספר אחר זה מקודד את התדר. לא צריך להקדיש בית מיוחד בשביל זה.
- gain – עוד 8 ביטים.
- synthesis filter – יש לו רק קטבים, לכל אחד צריך תדר + אמפליטודה. 8 ביטים כפול 2 פרמטרים, כפול 4 קטבים זה סה"כ 8 בתים.

נקבל 10 בתים, שהם 80 ביטים, כפול 25 ונקבל **2Kb לשנייה**.

<u>איך ממשים Linear Predictive Coding (קידוד על סמך ניבוי ליניארי של הדגימות):</u>

- gain – תלוי מה האנרגיה של הסיגנל. מחשבים סכום ריבועים כמו שאנחנו יודעים.
- U/V – voiced הוא מחזורי, unvoiced הוא סטוכסטי. בציר התדר אם זה מחזורי, הספקטרום נראה כמו מקלות.
- pulse – מה התדר של ה-pitch, פעולה בציר הזמן שנקראת אוטוקורלציה.
- synthesis – איך מוצאים את 4 הקטבים? יודעים מה הקלט ומה הפלט, **זו בעיית זיהוי המערכת הקשה (אין שליטה עליהם), וזה מסנן AR. צריך לפתור משוואות Yule-Walker**.

שיפור הדחיסה:

Compressing telephony quality speech
was once a tremendously important problem

The International Telecommunications Union set goals
to reduce speech transmission rates by factors of 2

We can thus compare:

- 128 kbit/sec – linearly quantized speech
- 64 kbit/sec – Pulse Code Modulation (logarithmic quantization)
- 32 kbit/sec – Adaptive Delta PCM
- 16 kbit/sec was never standardized
- 8 kbit/sec – Code Excited LPC

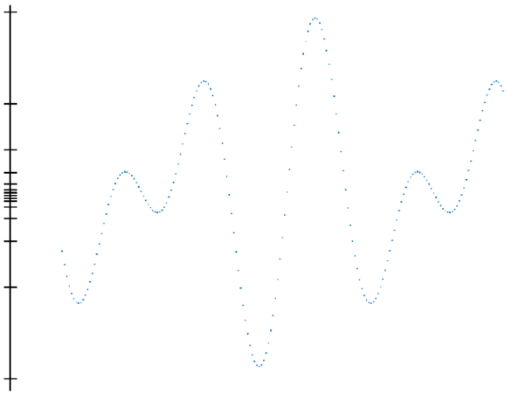
<u>פסיכופיזיקה:</u>

<u>הניסוי של Weber:</u> גילה שאנחנו לא חשים בהבדל של הפרשים, אלא יחסים. למשל, צריך להרגיש אחוז מסוים של הפרש כדי לחוש בהבדל בין מספר מטבעות שונים. זה מכונה **JND** (just noticeable difference): השינוי המינימלי הדרוש על מנת שנשים לב להפרש. זה רלוונטי לתפיסה של צלילים.

<u>חוק Fechner:</u> רצה לתרגם בין מה שהחושים קולטים לתחושה, ולא רק את הפרופורציה ביניהם. צריך מדד שמעביר בין מכפלה לחיבור. כלומר, כמעט כל התחושות שלנו לוגריתמיות בתופעות הפיזיקליות: $Y = A\log l + B$. החוויה הפנימית לוגריתמית בחוויה החיצונית. מבחינת שמיעה, **חווים את העוצמה באופן לוגריתמי** ולכן מודדים בדציבלים – אם משהו חזק פי 2, נשמע אותו יותר חזק ב-1. גם תדר חווים בצורה לוגריתמית, לכן הבדל של 7 סדרי גודל בין התדר הכי נמוך להכי גבוה ששומעים, זה הבדל של 7 רמות.

נתחיל לעשות את הפעולות כדי להקטין 128Kb ולהגיע ל-8Kb.

<u>PCM:</u>



הרעיון הוא לנצל את הלוגריתם. אמרנו שאנחנו משתמשים ב-2 בתים (16 ביטים) לכל דגימה כדי להוסיף כמה שפחות רעש. אמנם, **האוזן רגישה באופן לוגריתמי (חוק פכנר)**. בהתחלה נחלק לרמות מאוד צפופות סביב הערכים הקטנים, וככל שנתרחק נשתמש במעט רמות. אם נשתמש ב-256 רמות, **באופן לוגריתמי, לא שומעים את הרעש, ואז מספיק בית 1 במקום 2 בתים** – נוכל לעבור ל-**64Kb**.

<u>DPCM:</u> ה-pitch הוא פולסים עם הרמוניות שיורדות מהר (בערך 12 דציבל פר אוקטבה). כלומר סיגנל כמו פלט של מסנן low pass. יש יותר אנרגיה בתדרים נמוכים, ולכן הוא יחסית חלק, וניתן **לחזות** אותו, לפי הדגימות הקודמות. אם התדר נמוך, שתי דגימות הן **יחסית קרובות**. לכן נקודד את ההפרשים בין הסיגנל, לבין החיזוי שלו. בשביל לקודד הפרשים מספיקים 4 ביטים, במקום 8 ביטים. לכן נרד ל-**32Kb**. כאן נסתכל רק מקומית על ההפרשים, עם 4 ביטים יהיו לנו 16 רמות מסביב לנקודה הקודמת.

- <u>ADPCM:</u> מה המרווח בין 16 הרמות? Adaptive DPCM מאפשר אדפטציה של הרמות, ומבין אופטימלית מה הגודל.
- <u>AVQSBC:</u> לקודד בציר התדר לפי חוק פכנר. ב-FFT חילקנו את התדר לחלקים שווים, אבל אנחנו לא חווים אותם באותו אופן. צריך חלוקה לוגריתמית. משתמשים ב-**16Kb**.
- <u>דברים נוספים (שקפים 75 – 79):</u> LPC10, CELP, אנליזה וסינתזה, G.729.

Speech Recognition

זיהוי דיבור (STT): זה לא כל כך פשוט. יש הרבה חפיפה בין תנועות שונות, ואותה תנועה דוברים שונים בהקשרים שונים מבטאים בצורה שונה. ההקשר חשוב, המבטא, מהירות הדיבור וכו'. המוח שלנו משלים את כל הפערים שיש ומבין את מה שנאמר. גם אם מנסים לחזור על אותו משפט כמה פעמים ולבטא אותו באותה צורה – לא נצליח (nonlinear time warping).

מה שאנחנו עושים זה backtracking – מנסים לזהות, אם זה לא נכון הולכים אחורה ומנסים משהו אחר. נניח ברמה של מילה, המשמעות של FIRE משתנה בכל משפט. זה קורה גם ברמת הפונמה.

What does the word FIRE mean in the following sentence?

THE MAN SHOUTED

FIRE !

FIRE THE GUN !

FIRE THE GUN MAKER !

כאשר אנחנו שומעים שפה שאנחנו לא מכירים, יהיה קשה מאוד לעשות transcription, אנחנו לא רגילים לעשות backtracking בשפה זו. גם ההקשר משפיע על איך שאנחנו מפרשים את מה שאנחנו שומעים.

מרחק לוינשטיין:

נתחיל מגישת **DTW** – מבוסס על הרעיון שאנחנו יכולים להשוות שני צלילים ולראות מה הכי דומה, למרות ה-time warping הדינמי שדיברנו עליו. איך עובד תיקון אוטומטי של טקסט? יש מילון של מילים (יכול להיות דינמי ומושפע מסטטיסטיקה), וכאשר כותבים מילה, יש 3 פעולות שניתן לעשות על המילה – להוסיף אות (insertion), למחוק אות (deletion), ולהחליף אות (substitution). איך נמצא את המחיר המינימלי כדי לבחור את המילה הקרובה ביותר למילה שכתבנו?

מרחק לוינשטיין: אומד מרחק בין שתי מילים כאמור. נותנים ציון ל-3 סוגי השגיאות, כל שגיאה בעלות "1". משתמשים בתכנות דינמי כדי למצוא את המרחק המינימלי בין שתי מילים. החיסרון העיקרי – יקר בזמן ריצה.

Rules:

1 enter square from left (deletion) cost = 1

2 enter square from under (insertion) cost = 1

3a enter square from diagonal and same letter cost = 0

3b enter square from diagonal and different letter (substitution) cost = 1

4 Always use minimal cost

Backtrack to find path actually taken

g	9	8	7	7	6	5	5	5	4	3
n	8	7	6	6	5	4	4	4	3	4
i	7	6	5	5	4	3	3	3	5	5
s	6	5	4	4	3	2	3	4	4	5
e	5	4	3	3	2	3	3	3	4	5
s	4	3	2	2	2	2	2	3	4	5
s	3	2	1	1	2	2	3	4	5	6
o	2	1	0	1	2	3	4	5	6	7
r	1	0	1	2	3	4	5	6	7	8
p	0	1	2	3	4	5	6	7	8	9
	p	r	o	c	e	s	s	i	n	g

Remember:

The question is always how we got to a square

We see that the distance is 3

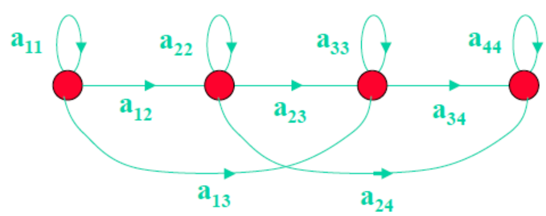
Since, e.g.,

1. **substitution**
2. **insertion**
3. **deletion**

מודלים מרקוביים:

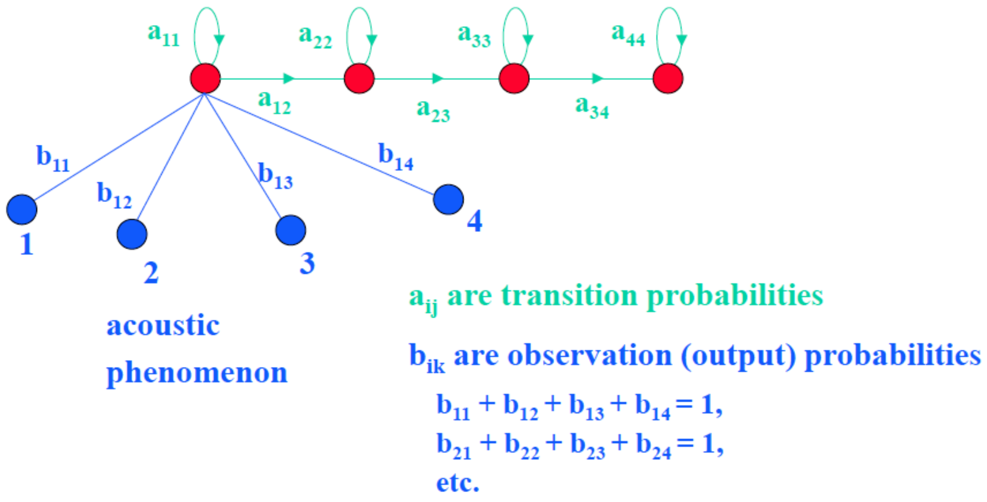
יש מצבים שהמודל עובר ביניהם. ניתן להישאר במצב בהסתברות מסוימת, ולעבור למצבים אחרים בהסתברות כלשהי. בהינתן הסתברויות מהעבר, ניתן לחשב הסתברות לקבלת מחרוזת מסוימת. במודל LR לא ניתן לחזור אחורה, זזים משמאל לימין. אפשר לדלג על מצב וכך נקבל deletion.

A DT LR second order Markov model



$a_{11} + a_{12} + a_{13} = 1,$
$a_{22} + a_{23} + a_{24} = 1,$
etc.

<u>מודל מרקוב חבוי (HMM):</u> הבעיה שבמודל מרקובי של דיבור, פונמה נהגית בצורה שונה בכל פעם. כאשר אנחנו במצב מסוים, אנחנו פולטים צליל 1 בהסתברות מסוימת, צליל 2 בהסתברות אחרת. **בנוסף להסתברויות של המעבר, יש הסתברות של הפלט עצמו שאנחנו מפיקים** (אלופון/פלט אקוסטי ספציפי). משתמשים באלגוריתם Baum-Welch של EM (expectation minimization).



עם מה עובדים היום?

DTWs and HMMs were the state of the art until recently

DTWs have little training time but long run times

HMM have long training times but short run times

Recently Deep Neural Networks have taken over many applications

DNNs
- use essentially no knowledge about the speech signal
- require astronomical amounts of data to train
- have very long training times
- provide no reasoning as to their decisions
- can frequently outperform classical methods
  but can also fail catastrophically
  and can be subject to *adversarial attacks*

Communication

Communications is moving information from place to place
Information is the amount of surprise, and can be quantified!
Communications was originally analog – telegraph, telephone
There are many forms of digital communications that we use every day

- Mobile (cellular)
  - Internet
  - from 4G *voice* is simply packet data
- Television
  - DVB (Idan plus)
  - Cable TV (DOCSIS)
- Home Internet access
  - DSL (ADSL, VDSL)
  - Cable modem (DOCSIS again)
- Internet of Things
  - smart home / smart city
  - Industry 4.0

Y(J)S DSP

מבוא:

תקשורת: העברת **אינפורמציה** ממקום למקום. עושים את זה דרך ערוצים (תווך דיגיטלי שיכול להעביר סיגנל). את האינפורמציה מוסיפים לסיגנל באמצעות **אפנון**. יש 2 מגבלות לכל ערוץ פיזיקלי:

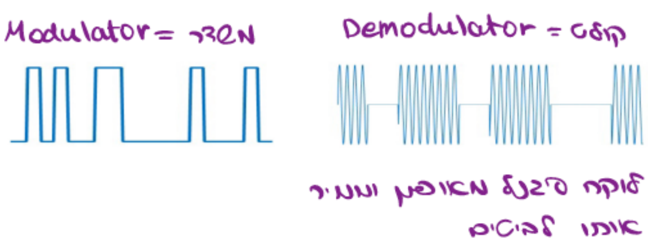
1. **מוסיף רעש** – מדובר בחוטי נחושת שמעבירים רעש מהסביבה, וגם נוצר רעש כשהם מתחממים.
2. **מגביל סרט** – יש תחום תדרים מוגבל שהוא יכול להעביר.

היתרון של דיגיטלי על אנלוגי: אנחנו מקבלים סיגנל מורעש. האם נוכל לתקן ולמצוא בחזרה את הסיגנל ללא הרעש? כאשר הסיגנל אנלוגי אין לנו דרך לבטל את הרעש. כאשר הסיגנל דיגיטלי – מדובר בביטים (0 או 1). אפשר לתקן את הערך, ובאמצעות **אלגוריתמים של תיקון שגיאות** אפשר להעביר מידע ממקום למקום ללא שגיאות בכלל (תחת תנאים מסוימים). לא נרצה להגביר את הסיגנל, כי זה גם יגביר את הרעש.

משפטי Shannon: כל המשפטים האלה אינם קונסטרוקטיביים, הם לא אומרים כיצד לבנות מערכת, הם אומרים מה המערכת הטובה ביותר האפשרית מסוגלת לעשות – באילו תנאים ניתן להעביר מידע דיגיטלי ללא שגיאה:

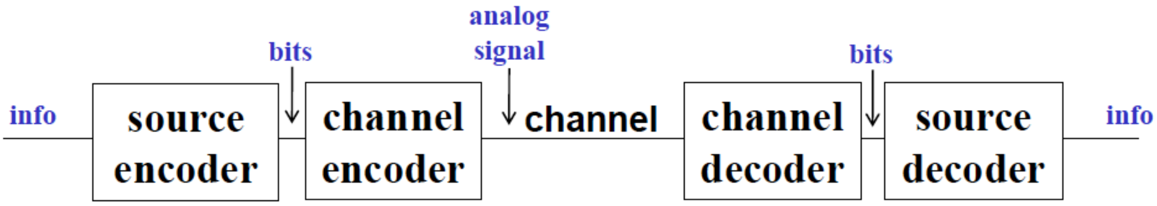
1. משפט ההפרדה (תקשורת).
2. משפט קידוד המקור (אינפורמציה).
3. משפט קיבולת הערוץ (DSP).

(1) משפט ההפרדה:



אפנון: לוקחים סיגנל שיש לו לפחות פרמטר אחד חופשי. בסינוס למשל יש את הגודל (A) ויש את התדר (f). נניח שהתדר קבוע ואז אפשר לדבר גם על הפאזה שלו. $Asin(\omega t+\phi)$. אפשר לשנות את A,ϕ כדי שהסיגנל ישתנה. נעביר ביטים על ידי שנשנה פרמטר של סיגנל. אפשר ערכים בדידים, לאו דווקא 0 ו-1 אלא יותר.

מכשיר שיש לו משדר וקולט נקרא **מודם** – משדרים ואז קולטים. לפי שאנון, הדרך האופטימלית להעביר סיגנל אנלוגי היא להכניס את האות למשהו שמקודד את האינפורמציה – **מקודד מקור**. הוא פולט ביטים, לא משנה איזה אות נכנס. אחרי שהפכנו אינפורמציה לביטים, מכניסים את זה **למקודד ערוץ** – לוקח ביטים והופך אותם לסיגנל אנלוגי מאופנן. עושים את זה כי הערוץ הפיזיקלי הוא אנלוגי. הוא מתעוות ומקבל רעש מהערוץ ונכנס ל**מפענח ערוץ**. זה לוקח את הסיגנל ומחזיר ביטים. אם זה עובד כמו שצריך, נקבל את אותם ביטים ששלחנו.



(2) משפט קידוד המקור:

אינפורמציה: שאנון הגדיר לנו מה זה אינפורמציה. יחידת האינפורמציה הבסיסית, האוניברסאלית היא **ביט**: מכמתת כל סוג של אינפורמציה, לא רק מספרים. נניח שלמישהו אחד יש אינפורמציה, ולאחר אין אותה. אם לאחר יש אותה – לגלות לו אותה זה לא אומר כלום. **אינפורמציה היא מידת ההפתעה כאשר הוא לומד משהו חדש**.

איך נכמת אינפורמציה? אפשר לשאול שאלות על האינפורמציה, כאלו שהתשובה עליהן היא כן/לא בלבד. הביטים של שאנון, היא כמות השאלות שאנחנו יכולים לשאול כדי לדעת את האינפורמציה – **המספר הקטן ביותר של שאלות כאלו**.

דוגמה בה הביטים של שאנון ושל תוכי שווים: צריך לנחש מספר נתון בין 0 ל-15, נרצה לחלק כל פעם בפקטור 2.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

1. Is greater than 7 ?   YES = **1**
2. Is greater than 11?   NO = **0**
3. Is it greater than 9 ?   YES = **1**
4. Is it greater than 10?   NO = **0**

So the Shannon information quantity is 4 bits
and the answers form the Tukey binary number for 10 – **1010**

המשפט של שאנון אומר כמה ביטים צריך באמת כדי לקודד את האינפורמציה. הוא לא מלמד אותנו איך לקודד בפועל את האינפורמציה. במשפט זה אפשר להשתמש להרבה דברים: בהינתן קובץ טקסט יש בו יתירות, יותר מידי אינפורמציה. יש את הדחיסה של למפל-זיו, כדי להפוך אותו לאוסף מינימלי של שאלות כן/לא. **לא ניתן להקטין יותר מהגבול של שאנון לפי משפט קידוד המקור**.

תיקון שגיאות: יש טכניקות בתורת האינפורמציה כדי לגלות שגיאות, וגם כדי לתקן שגיאות. עושים שימוש בביט זוגיות. מקודדים 7 ביטים ועוד ביט זוגיות בתוך בית 1. אם ב-7 ביטים של האינפורמציה יש מספר זוגי של אחדות, נשים בו 0, אחרת נשים בו 1. אם אחד מהביטים התהפך בגלל רעש למשל, נוכל לגלות מיד שיש שגיאה כי בבית יש מספר אי-זוגי של אחדות.

(3) משפט קיבולת הערוץ:

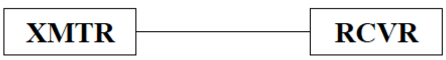
לכל ערוץ יש קיבולת מקסימלית, X ביטים לשניה שניתן להעביר דרך הערוץ. יש קיבולת שעד אליה יש דרך לבנות מקודד ערוץ שיכול להעביר את הביטים בערוץ ללא שגיאות כלל, ואם ננסה לעבור את הקיבולת הזאת, בהכרח יהיו שגיאות.

זה תלוי במה שאמרנו שיש לכל ערוץ פיזיקלי: מגביל רוחב סרט ומוסיף רעש. אם יודעים את היחס בין הסיגנל שמשדרים לבין הרעש ואת רוחב הסרט של הערוץ, נוכל לבנא כמה ביטים לשנייה לפי משפט ההפרדה ניתן להעביר.

הוכחה (פשטנית): נתחיל ממודל פשוט. יש לנו משדר ומקלט. המטרה היא להעביר אוסף ביטים מצד לצד בלי שגיאות בכלל. אין רעש בכלל. מתי אין מגבלה על מספר הביטים שניתן להעביר?

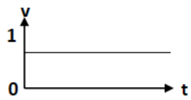
1. אם הערוץ לא מוסיף רעש בכלל – גם אם הוא מגביל מאוד רוחב סרט, ניתן להעביר כמה ביטים בשנייה שרק נרצה.
2. אם הערוץ מוסיף המון רעש ולא מגביל רוחב סרט – ניתן להעביר כמה ביטים שנרצה בשנייה.
3. אם נמצא תווך שאין בו רעש וגם לא מגביל רוחב סרט – נוכל להעביר כמה מידע שנרצה, בלי שגיאות בכלל.

נתחיל מהמקרה הראשון: יש לנו תווך עם רוחב סרט מוגבל אבל אין רעש בכלל.

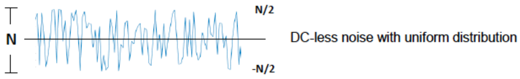


Let's take some large amount of information

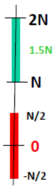
- source encode it into bits 10111010010111100010101000010...
- add "0." at the beginning 0.10111010010111100010101000010... to obtain a number between 0 and 1
- place precisely that voltage as DC on the *wire* connecting transmitter to receiver



- נקודד את המידע ע"י ביטים (נניח ASCII), ונוסיף "0" לפני מצד שמאל. קיבלנו מספר ממשי בין 0 ל-1.
- נעביר על התווך מתח בדיוק עם המספר הזה, שהוא מספר מדויק. התווך לא מוסיף רעש, ולכן המספר שיגיע לצד השני זה בדיוק המספר שנשלח – המתח.
- אין מגבלה על הזמן שלוקח למדוד את המתח הזה. הוא **קבוע** - מדובר בסיגנל DC, רוחב הסרט שלו הוא אפס. אין בעיה פיזיקלית למדוד ערך של סיגנל DC. אין מגבלה של עיקרון אי-הוודאות.



Let's take some large amount of information

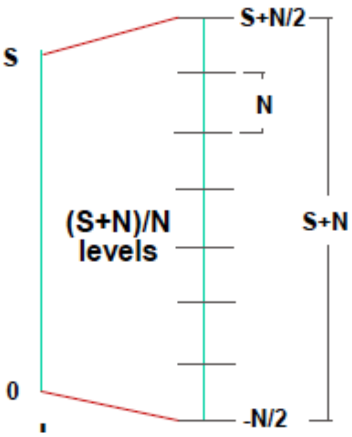


- source encode it into bits 1011101001011110001010100010...
- for every 0 bit transmit 0 voltage for T seconds
  for every 1 bit transmit 1.5N voltage for T seconds
- for every 0 bit the receiver will see a voltage between –N/2 and N/2
  for every 1 bit it will see a voltage between N and 2N
- since there is no overlap the receiver can identify 0/1 with no error
- since the bandwidth is infinite, we can send T→0

<u>המקרה השני:</u> יש רעש אבל אין מגבלה של רוחב סרט.

- נקודד באותו אופן. נניח שלרעש יש גודל N. על כל ערך שאנחנו משדרים, הוא יכול להוסיף או להחסיר $\frac{N}{2}$ לכל היותר.
- נשדר את הביטים, ובכל דקה נשדר מתח 0 או 1.5N. בצד השני לא נוכל לטעות, אין חפיפה בין הערכים האפשריים כולל הרעש.
- כיוון שרוחב הסרט לא מוגבל, אפשר **להקטין את המרווחים של זמן השידור** ככל שנרצה ללא שגיאה.

<u>המקרה השלישי:</u> יש גם רעש וגם מגבלה של רוחב סרט.



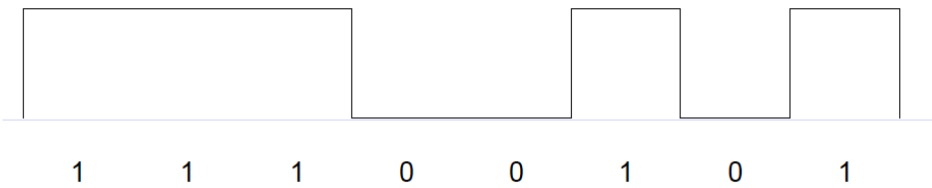
- נשדר ערכים בין 0 ל-S. הרעש הוא בגודל N.
- בתוספת הרעש, הצד השני יראה טווח של $S+N$.
- צריך לגרום לכך שהצד השני יוכל להפריד תדרים בלי שגיאה. נחלק את הטווח הזה לחלקים בגודל N. אם נקרב שני אזורים יותר מ-N, תהיה לנו חפיפה ולא נוכל להפריד. אם נרחיק יותר מ-N יהיו חלקים שנפספס ולא נשתמש בהם. יש לנו טווח של $S+N$ ומחלקים ל-N חלקים, כלומר מספר החלקים $\frac{S+N}{N}=\frac{S}{N}+1$. S/N מכונה **SNR** (יחס אות לרעש). כדי לייצג את זה צריך $\log_2\left(\frac{S}{N}+1\right)$ ביטים.
- כל פעם ששידרנו העברנו כמות כזאת של ביטים. כלומר אם למשל היו 8 רמות שונות שאפשר לשדר, ניתן לייצג כל רמה על ידי 3 ביטים. זה אומר כמה ביטים צריך כדי לייצג כל סימבול.
- משפט שאנון – **קיבולת הערוץ כאשר לערוץ יש רוחב סרט BW ויחס אות לרעש SNR, הוא $C=BWlog_2(SNR+1)$ ביטים לשנייה**. אם ננסה לשדר יותר ביטים מהמספר הזה, נקבל בהכרח שגיאות.
- רוחב סרט BW – זה אומר כמה פעמים בשנייה הסיגנל יכול להשתנות. אפשר לשלוח BW סימבולים בשנייה, שכל אחד מיוצג ע"י $\log_2(SNR+1)$ ביטים.

<u>מודמים:</u>

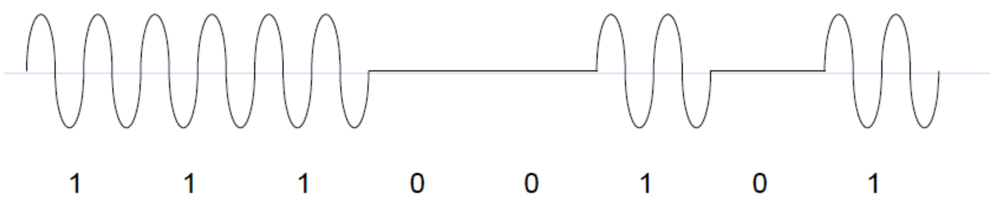
התיאוריות של שאנון הן לא קונסטרוקטיביות, צריך למצוא את הדרך להתקרב כמה שיותר לשאנון.

<u>NRZ (non return to zero):</u> המודם הפשוט ביותר האפשרי. משדרים את ערכים נפרדים עבור 1 (מתח כלשהו) ועבור 0. בכל זמן נתון משדרים ביט אחד. היא מאוד רחוקה מהקיבולת של שאנון.

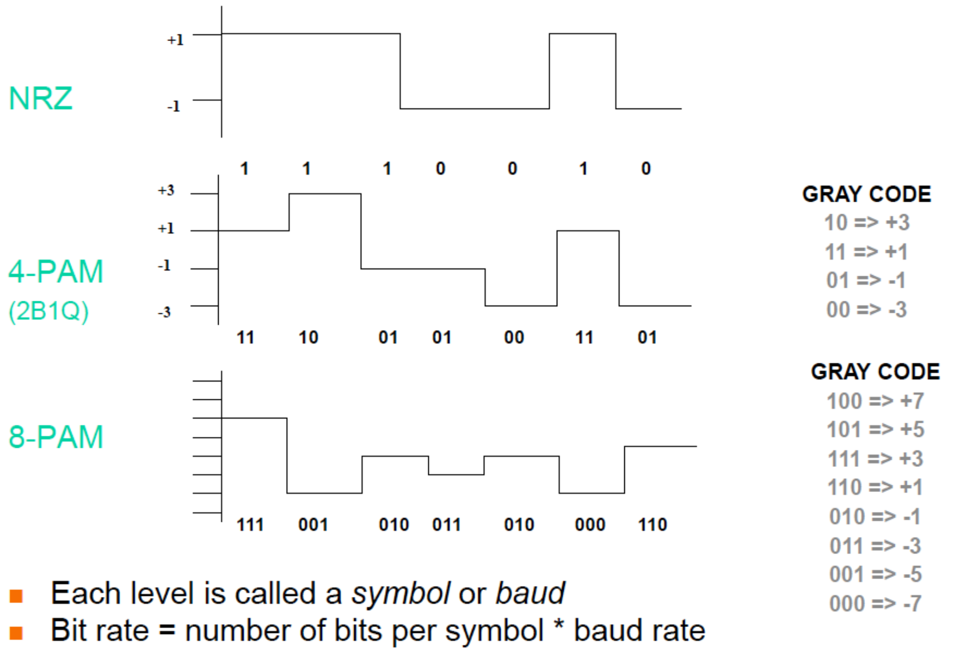
- זה עובד רק בשביל **DC**, חוט נחושת שמעביר תדר 0.
- זוג חוטים הוא low pass filter, שמעביר תדרים נמוכים, אבל כאשר יורדים מ-1 ל-0, ציירנו את זה בתור 90 מעלות. ה-1 זולג ל-0, אי אפשר **מיד** להוריד אותו ל-0. יש מגבלה של רוחב סרט (InterSymbol Interference).
- אנחנו זקוקים לרוחב סרט אינסופי, כאשר נתחיל להקטין את רוחב הסרט, לא נוכל להבחין בין הביטים.



<u>OOK (on off keying):</u> משדרים כמו מורס, אם לא ניתן להעביר DC בכלל (כמו בשידור רדיו) – משדרים (on) או לא משדרים (off) **סינוס**. במקום לאפנן DC, נאפנן את סינוס. OOK ממתג רק את האמפליטודה של סינוס, משתמש בפרמטר אחד ולכן הוא בזבזני.



<u>PAM (pulse amplitude modulation):</u> בכל רגע נתון במקום ביט אחד, יהיו יותר ביטים.



<u>שיפורים נוספים:</u>

- <u>ASK:</u> מכליל את OOK כמו ש-PAM הכליל את NRZ. משתמשים בכמה רמות של סינוס.
- <u>FSK:</u> במקום להשתמש באמפליטודה של הסינוס כדי לאפנן, נשתמש בתדר.
- <u>PSK:</u> לוקחים סינוס בתדר אחד, ומשדרים סינוס או קוסינוס בתדר הזה (shift בפאזה).
- <u>QAM:</u> משנה פאזה וגם אמפליטודה של סינוס. משתמשים בהתמרת הילברט כדי למצוא את A ואת Φ.
- <u>OFDM:</u> מה שעושים היום – מודדים את הערוץ בין שני הצדדים, משתמשים במשפט איסוף המים – זה אומר כמה ביטים שמים בכל תא, בכל אחד עושים QAM, ב-4G עושים לכאורה 1024 כאלה, אבל FFT מחשב את כולם בבת אחת.