DEC – SDR – DSP project 2017 (2)

Inhoud:

- DSP software en rekenen
  - Effect van type getallen (integer, float)
- Fundamenten onder DSP
  - Lezen van eenvoudige DSP formules ‘x[n]’
  - Lineariteit ($x \xrightarrow{\text{functie}} y$ dus k maal $x \xrightarrow{\text{functie}}$ k maal y)
  - Superpositie (= soort van optellen)

Getallen in computers: Integers

- Integers, dus gehele getallen
- b.v. 16 bit
- Diverse coderingen
- Voldoende voor ADC en DAC

- Rekenen gaat heel snel, echter beperkte nauwkeurigheid
- Beperkte range (zie onder)
- Bij rekenen grote kans op overflow (buiten 16 bits)
- Eventueel kan met long integers van 32 bit worden gewerkt of met ‘long long’ = 64 bit

UNSIGNED INTEGER

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

16 bit range:
0 to 65,535

OFFSET BINARY

Decimal	Bit Pattern
8	1111
7	1110
6	1101
5	1100
4	1011
3	1010
2	1001
1	1000
0	0111
-1	0110
-2	0101
-3	0100
-4	0011
-5	0010
-6	0001
-7	0000

16 bit range
-32,767 to 32,768

SIGN AND MAGNITUDE

Decimal	Bit Pattern
7	0111
6	0110
5	0101
4	0100
3	0011
2	0010
1	0001
0	0000
0	1000
-1	1001
-2	1010
-3	1011
-4	1100
-5	1101
-6	1110
-7	1111

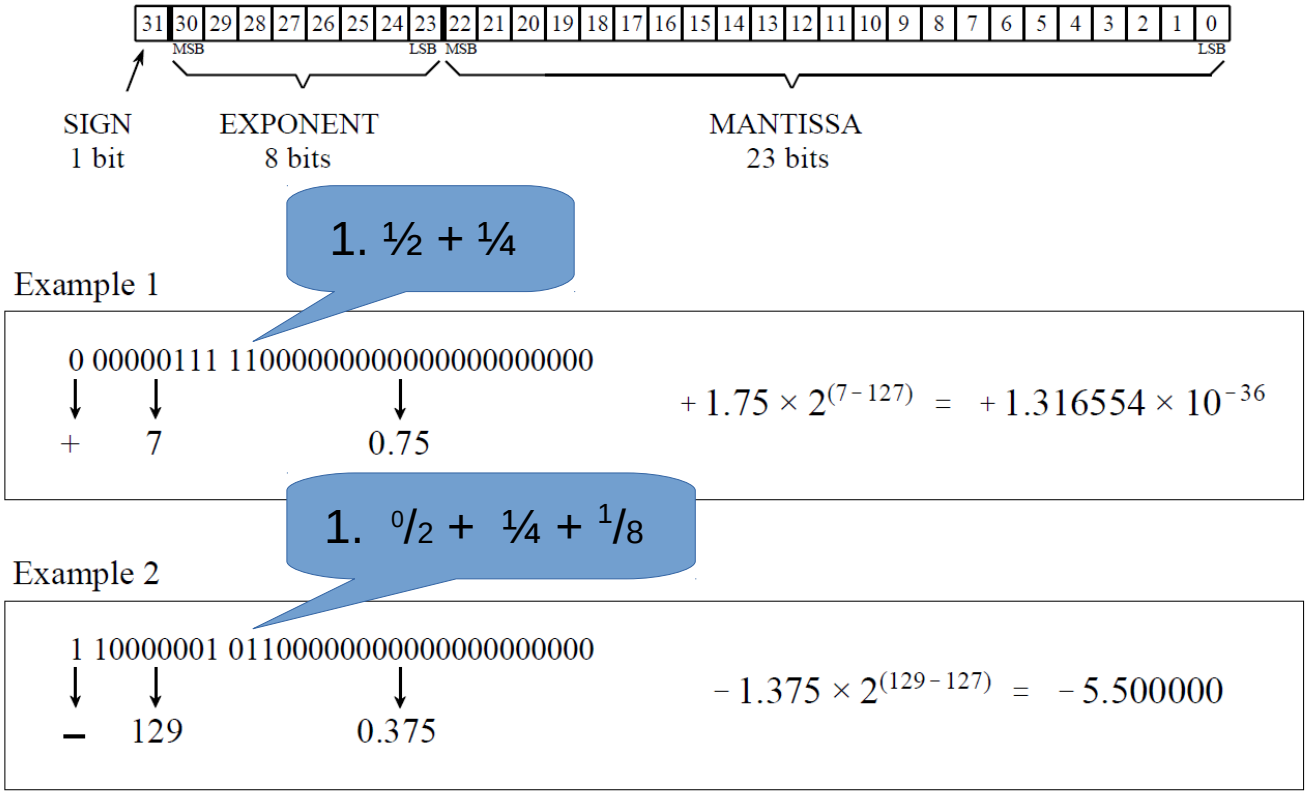
16 bit range
-32,767 to 32,767

TWO'S COMPLEMENT

Decimal	Bit Pattern
7	0111
6	0110
5	0101
4	0100
3	0011
2	0010
1	0001
0	0000
-1	1111
-2	1110
-3	1101
-4	1100
-5	1011
-6	1010
-7	1001
-8	1000

16 bit range
-32,768 to 32,767

Getallen in computers: Floating Point

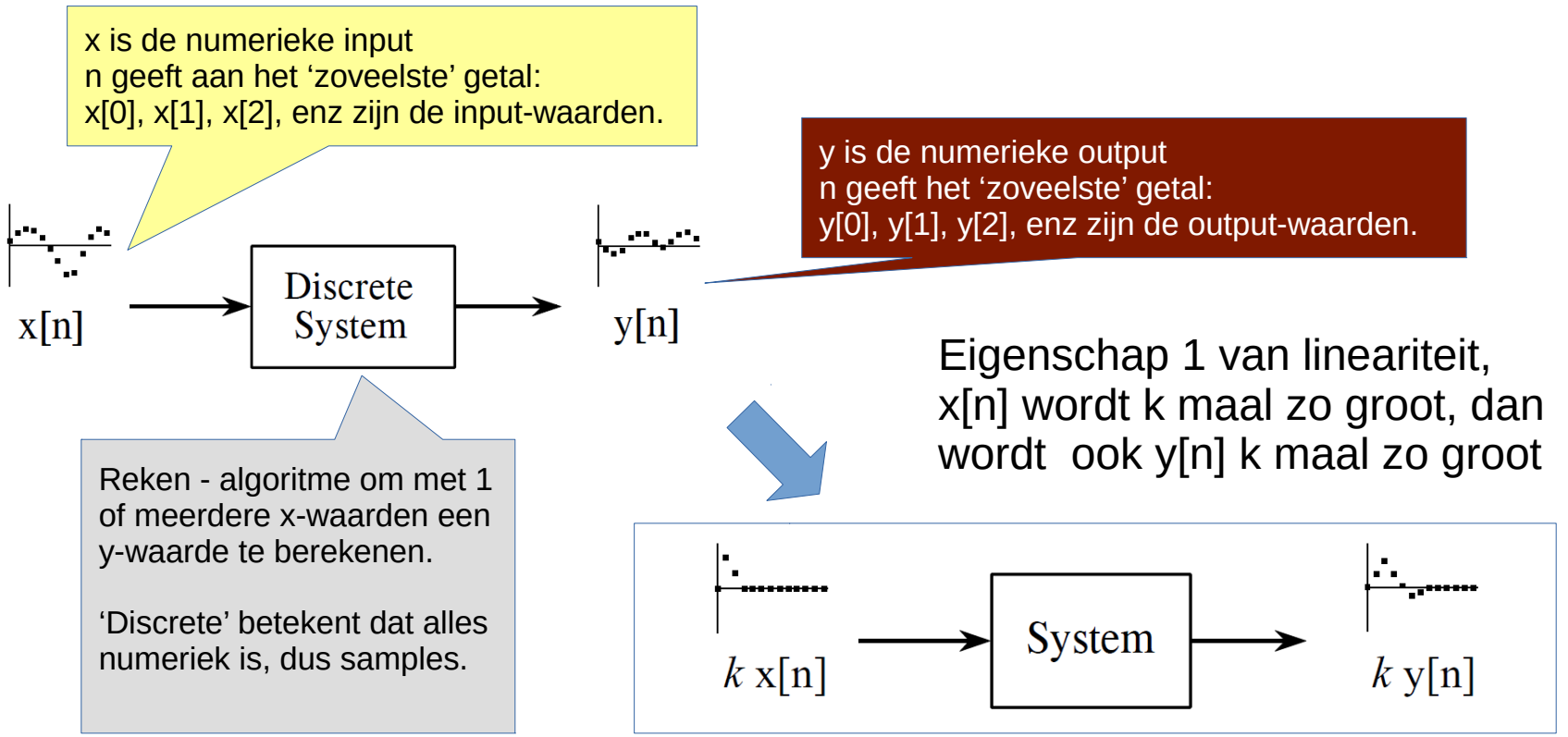


- Voor getallen als 6,34 x 10^{22}
- Single precision: 32 bit (zie voorbeeld boven)
- Double precision: 64 bit
  dus hogere machten en meer cijfers achter de komma

- Ondanks de grote range kan niet ieder getal binnen de range worden weergegeven. Dit heet *round-off error*.
  (er is niet een oneindig aantal stapjes tussen de max en de min)
- Fout in een berekening *kan* groot worden bij veel rekenen in een loop
- Dankzij huidige processoren in pc's geen probleem om snel te rekenen met 'floats'.

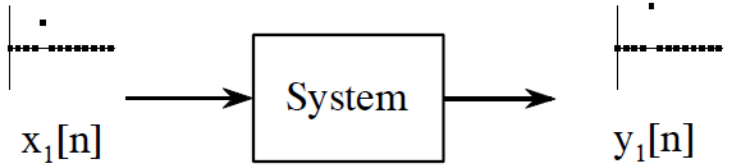
DSP: Lineariteit = (1)

- Hoe lees je de diverse figuren over DSP:

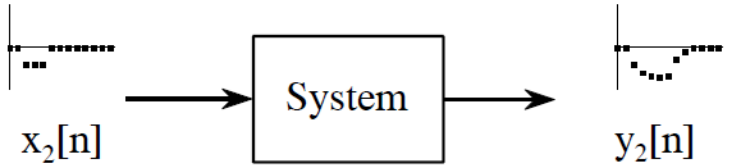


Lineariteit = (2)

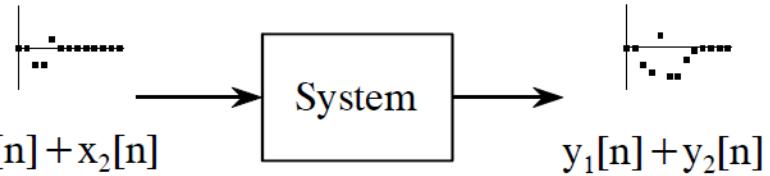
Als



En als

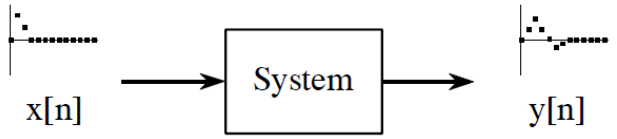


Dan is

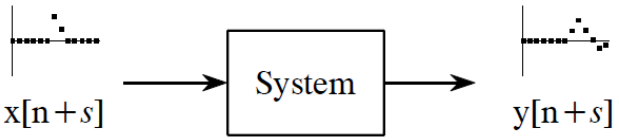


Als meerdere signalen aan de ingang worden opgeteld en dan door het systeem gaan, dan is de output gelijk aan de som van iedere output afzonderlijk

Als

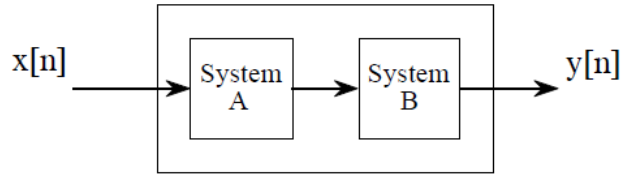


Dan is

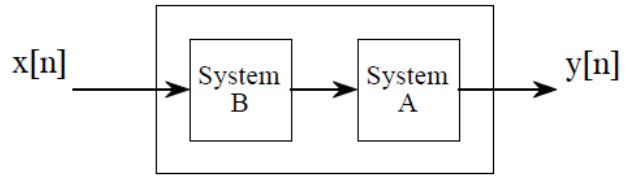


Als x[n] is vertraagd in de tijd, dan ook y[n] evenveel

Als

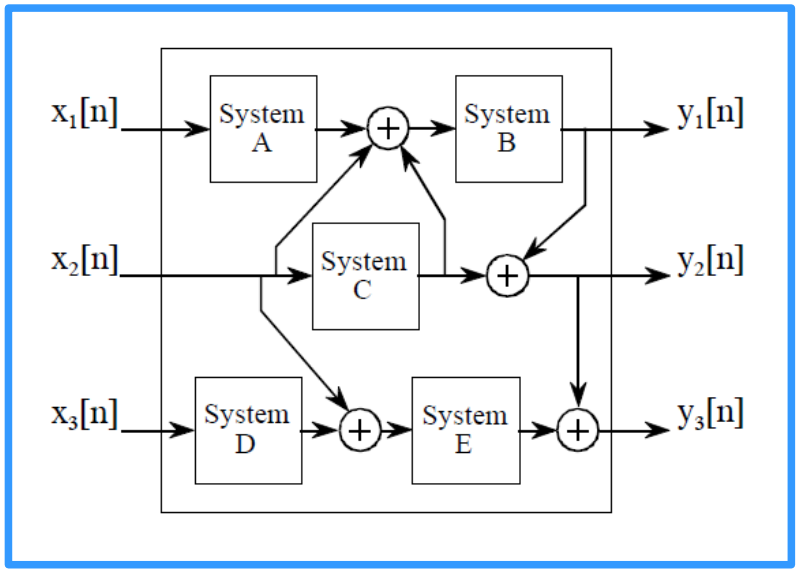


Dan is

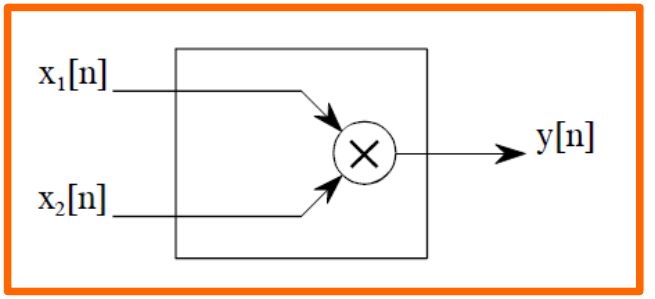


Twee bewerkingen (A en B) achter elkaar hebben hetzelfde effect op x[n] als dat ze in omgekeerde volgorde zouden staan.

Lineariteit = (3)



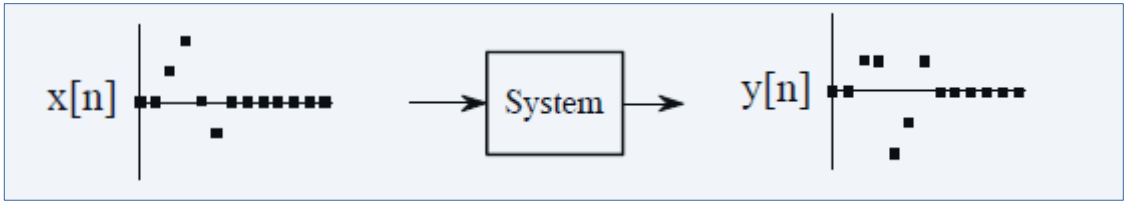
Zolang we signalen optellen blijft het systeem nog steeds lineair.



Signalen met elkaar *vermenigvuldigen*:
NIET lineair.

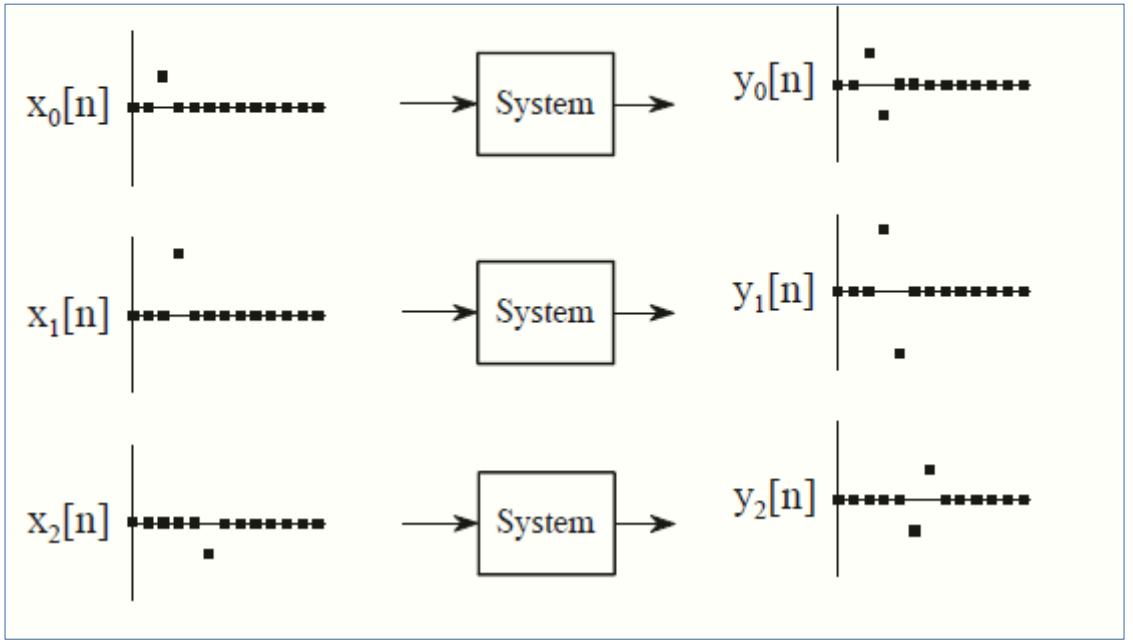
Let op: hier worden signalen *met elkaar* vermenigvuldigd en niet met een constante term ‘k’

DSP: Superpositie (= som)



Uitsplitsen
(Decompositie)

Samenvoegen
(Synthese)



x[n] wordt opgesplitst in signalen met een enkele impuls $x_1[n]$, $x_2[n]$, enz.

De impuls responsie van ieder puls apart, $y_1[n]$, $y_2[n]$, enz, worden gesommeerd tot y[n].

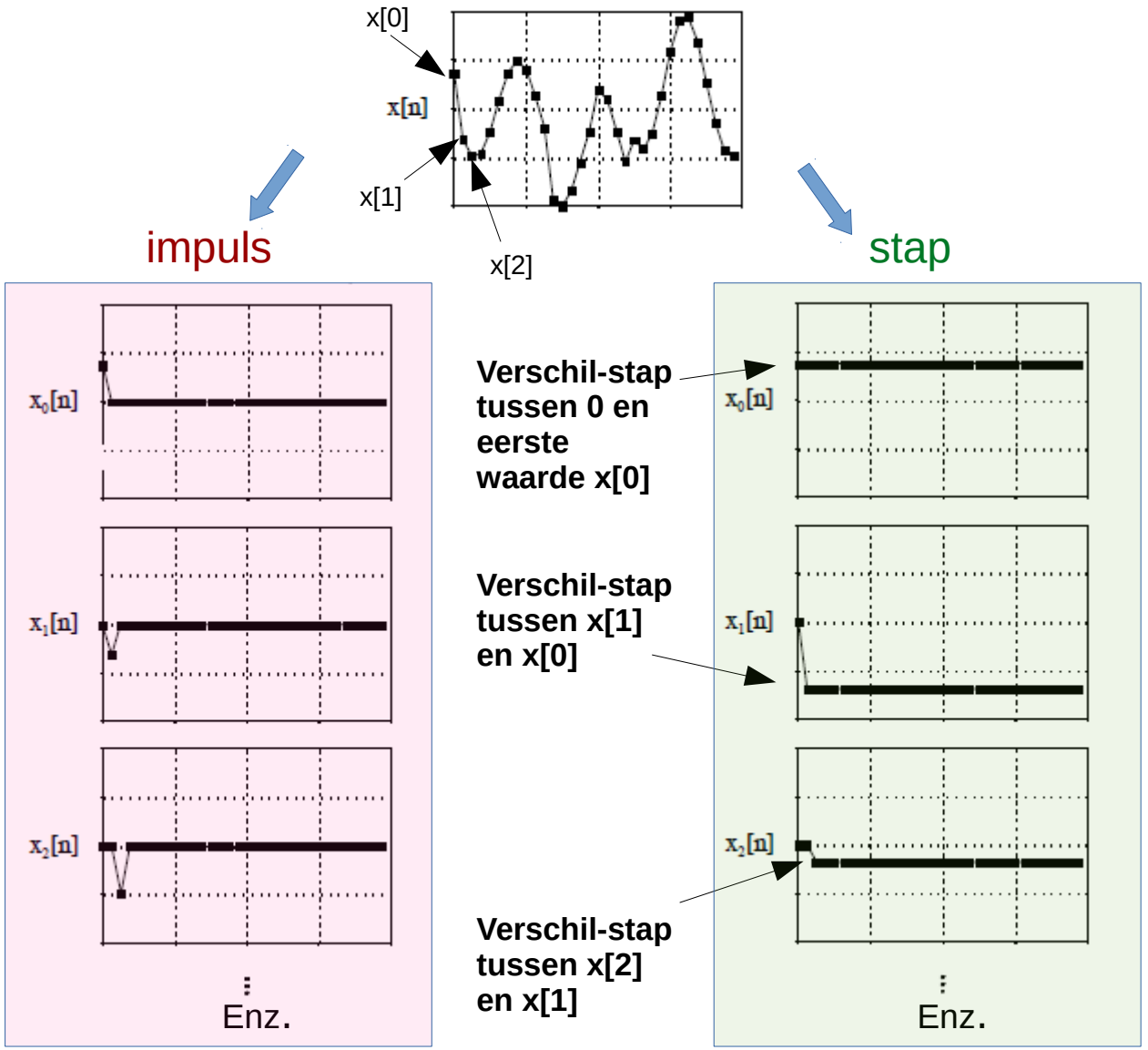
Deze samenvoeging is gelijk aan de output die we gehad zouden hebben als we het gehele signaal x[n] hadden gebruikt.

Verschillende decomposities (uitsplitsingen)

- Uitsplitsing naar **impulsen**
  (zie vorige sheet)

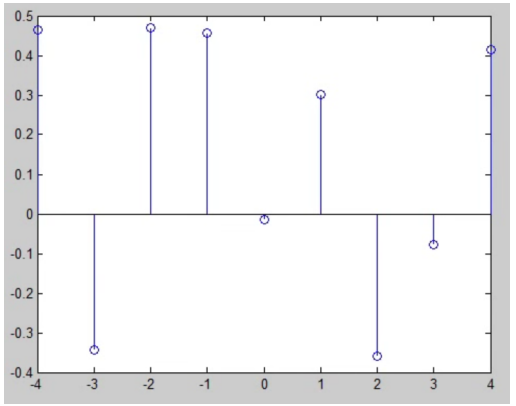
- Uitsplitsing naar **stappen**
  Dit is de verschil-stap die het signaal moet maken om naar het volgende punt te komen.

- Beide uitsplitsingen (impulsen en stappen) $x_1[n]$ tot $x_N[n]$ geven opgeteld uiteindelijk hetzelfde signaal x[n] terug.

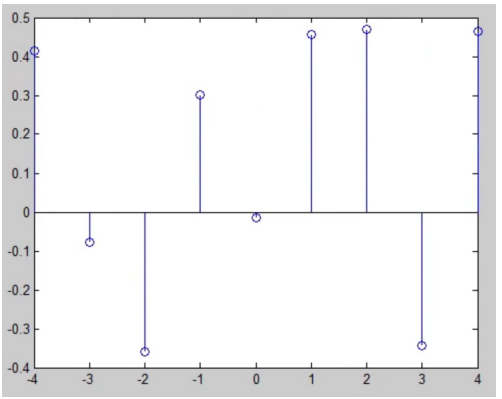


Even / Oneven symmetrische uitsplitsing

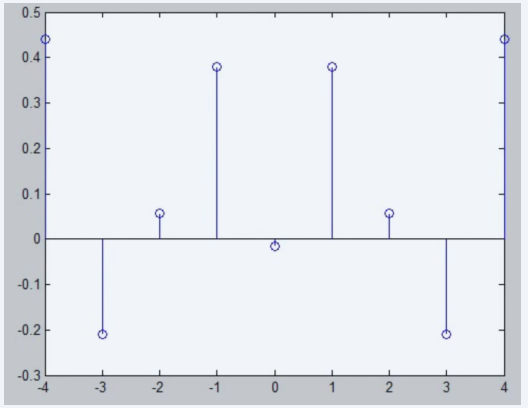
x[n]



x[-n]
(= x[n] maar geflipt om de x[0] heen)

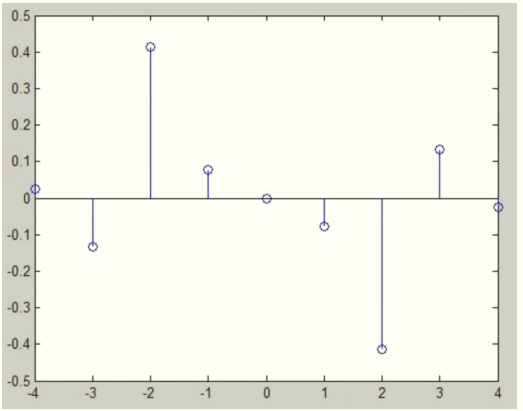


Even (symmetrie Y-as)
vergelijk: cosinus



$X_{even}[n] = \frac{1}{2}(x[n] + x[-n])$

Oneven (symmetrie 0-punt)
vergelijk: sinus



$X_{onev}[n] = \frac{1}{2}(x[n] - x[-n])$

Even: $X_{even}[n] = \frac{1}{2}(x[n] + x[-n])$

Oneven: $X_{onev}[n] = \frac{1}{2}(x[n] - x[-n])$ +

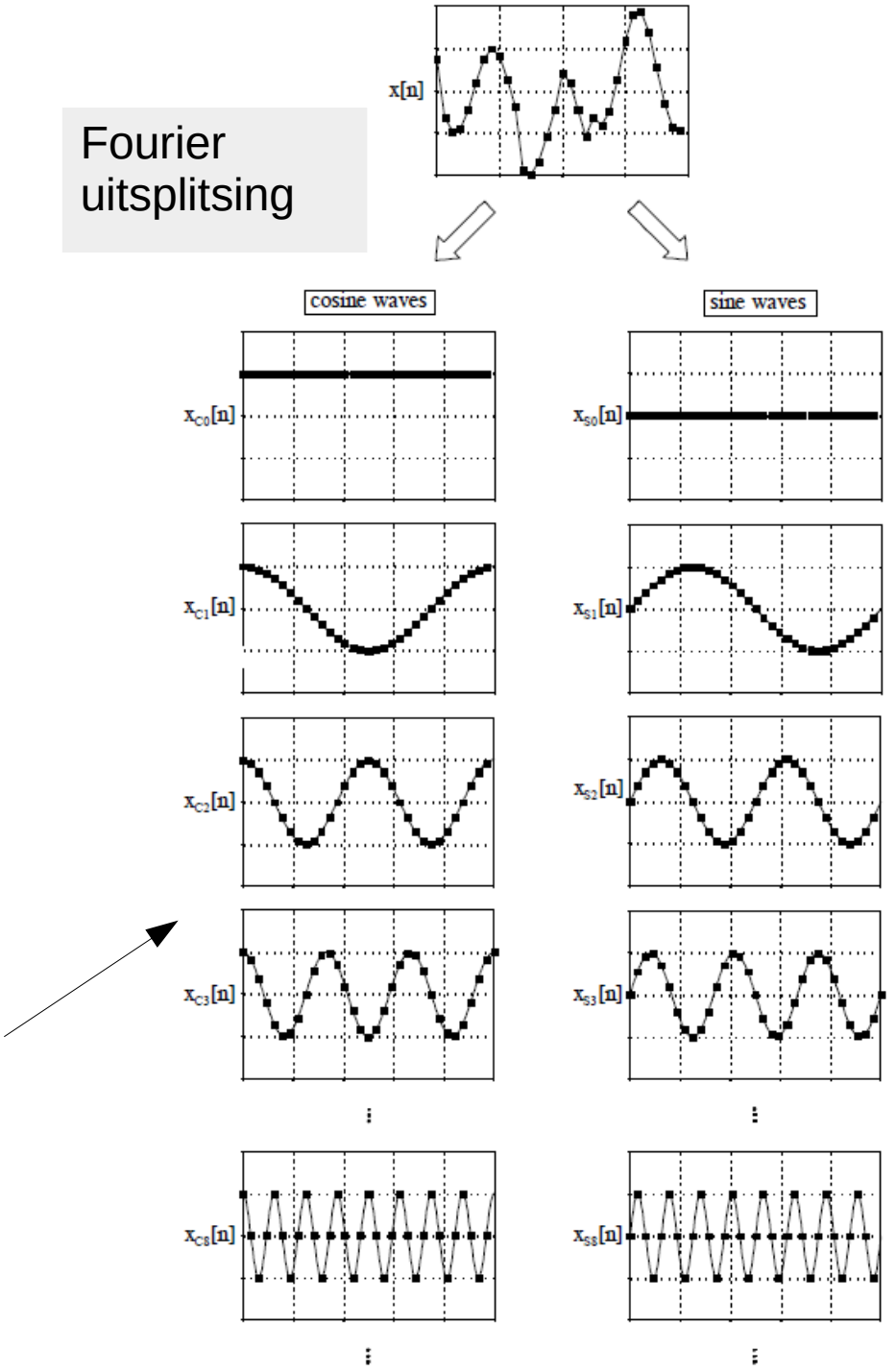
$X_{even}[n] + X_{onev}[n] = x[n]$

PI4DEC

Uitsplitsingen (de laatste)

- **Interlaced**: splits de samples in meerdere groepen: b.v. de even en de oneven samples:
  - Even: n[0], n[2], n[4], etc.
  - Oneven: n[1], n[3], n[5], etc.
  - Het herhaald uitsplitsen van de samples, zorgt b.v. dat het Fast Fourier Transform algoritme ook echt ‘fast’ wordt.

- Uitsplitsen in **harmonische sinus- en cosinussignalen** volgens wiskunde van **Fourier**.
  Een signaal van N samples wordt dan gesplits in N+2 sinus en cosinus signalen



Volgende keer

- Convolutie (als vervolg op Superpositie)
- Hoe met de tot nu toe behandelde technieken een output van eenvoudig filter wordt berekend.