Overaftelbaar oneindig veel getalsontwikkelingen

Hans Fritzsche

Voorwoord

Deze scriptie is geschreven als afronding voor mijn bachelor Wiskunde aan de Universiteit Leiden, in samenwerking met Dr. C.C.C.J. Kalle. Ik wil bij deze Dr. C.C.C.J. Kalle enorm bedanken voor haar inzet en begeleiding die mij geholpen hebben deze scriptie tot een succes te brengen.
Ik wens de lezer veel plezier bij het lezen van mijn scriptie.

Inhoudsopgave

1 Inleiding

In het dagelijks leven wordt met een 10-tallig stelsel gewerkt. Dit betekent dat elk getal (jaartal, prijs, leeftijd, hoeveelheid) opgedeeld kan worden in machten van 10. Het opdelen van een getal in een bepaald stelsel wordt ook wel het ontwikkelen van een getal genoemd. Zo kan het getal 4283,50 ontwikkeld worden in $4 \cdot 10^3 + 2 \cdot 10^2 + 8 \cdot 10^1 + 3 \cdot 10^0 + 5 \cdot 10^{-1}$. In mijn scriptie zal ik naar een speciaal soort stelsel kijken, namelijk het 'β-tallig stelsel' waarbij $1 < \beta < 2$.

Een verschil tussen het 10-tallig stelsel en het β-tallig stelsel is dat een getal zoals 4283,50 op meerdere verschillende manieren ontwikkeld kan worden. Een ander verschil is dat bij het 10-tallig stelsel ik de keuze heb getallen uit $\{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$ voor een macht van 10 te zetten, terwijl in het β-tallig stelsel ik alleen 0 of 1 mag gebruiken. Dit verschil is cruciaal, omdat met behulp van het β-tallig stelsel ontwikkelingen zullen ontstaan die alleen uit nullen en enen bestaan. Zulke rijtjes van nullen en enen worden gebruikt in elk digitaal aparaat. Dit stelsel zou dus gebruikt kunnen worden bij ontwikkeling van onder andere computers, mobiele telefoons, mp3-spelers enzovoort.

Je zou je af kunnen vragen waarom dit stelsel gebruikt zou worden. We hebben nu namelijk al een stelsel dat gebruikt wordt bij ontwikkeling van computers en dergelijke, namelijk het 2-tallig stelsel. Er is echter een belangrijk verschil tussen het 2-tallig stelsel en het β-tallig stelsel. Het zal namelijk blijken, zie sectie 5, dat het β-tallig stelsel zijn eigen foutjes kan herstellen waar het 2-tallig stelsel dat niet kan. Dit betekent dat als een computer bezig is een ontwikkeling te maken (een rijtje van nullen en enen) en op de i-de plek per ongeluk een 0 zet waar een 1 had moeten staan, de rest van de rij achter de i-de plek op een manier wordt veranderd zodat de gehele rij nog steeds dezelfde waarde aangeeft. Dit betekent dus dat als foutjes gemaakt worden het niet erg is voor de code. Dit kan helpen om storingen van signalen (zoals radiogolven) te voorkomen.

In deze scriptie komen de volgende onderwerpen aan bod:

> Hoofdstuk 2. Hier zal ik bespreken hoe een ontwikkeling gevonden kan worden, met zogenaamde dynamische systemen of met een algoritme.
>
> Hoofdstuk 3. Hier zal ik laten zien hoe verschillende ontwikkelingen voor hetzelfde getal gevonden worden.
>
> Hoofdstuk 4. Ik zal hier de hoofdstelling van de scriptie formuleren en bewijzen. Deze stelling zegt iets over het aantal verschillende ontwikkelingen voor bepaalde waarden van β.
>
> Hoofdstuk 5. Hier zal ik uitgebreid en gedetailleerd verklaren hoe de ontwikkelingen van het β-tallig stelsel gebruikt kan worden om een telefoon te laten werken. Ik zal laten zien hoe analoge signalen (een stem, muziek) omgezet kunnen worden naar digitale signalen (een rij van nullen en enen) met behulp van het β-tallig stelsel.

2 Bepalen van ontwikkelingen

2.1 Dynamische systemen

Een manier om een getal te ontwikkelen is met behulp van een dynamisch systeem. Een *dynamisch systeem* is een systeem dat door een functie

$$T(x) : \left[0, \frac{1}{\beta - 1}\right] \to \left[0, \frac{1}{\beta - 1}\right]$$

gegeven wordt, waarbij β de basis van het systeem is. Het systeem is dynamisch omdat elk volgende getal in de ontwikkeling bepaald wordt door een volgende iteratie van T te nemen. Tijdens dit hoofdstuk zal ik aannemen dat $1 < \beta < 2$ en $T(x) = \beta x - b_1(x)$ waar $b_1(x) = \begin{cases} 0 & \text{als } x < \frac{1}{\beta}, \\ 1 & \text{als } x \geq \frac{1}{\beta}. \end{cases}$

Ik laat hieronder zien hoe je met behulp van deze functie een ontwikkeling voor x maakt:
Zij β en T zoals beschreven en zij $x \in \left[0, \frac{1}{\beta-1}\right]$ willekeurig. Als ik T toepas op deze x geldt de volgende vergelijking:

$$T(x) = \beta x - b_1(x).$$

Als ik hieruit x isoleer krijg ik de vergelijking:

$$x = \frac{b_1(x)}{\beta} + \frac{T(x)}{\beta}.$$

Nu pas ik T nogmaals toe op het resultaat van $T(x)$. Ik neem dus $T\left(T(x)\right)$ welke gegeven wordt door:

$$T^2(x) = T\left(T(x)\right) = \beta T(x) - b_2(x).$$

Als ik hier $T(x)$ isoleer volgt:

$$T(x) = \frac{T^2(x)}{\beta} + \frac{b_2(x)}{\beta}.$$

Als ik de gevonden vergelijking voor $T(x)$ nu invul in de vergelijking voor x volgt dat:

$$x = \frac{b_1(x)}{\beta} + \frac{b_2(x)}{\beta^2} + \frac{T^2(x)}{\beta^2}.$$

De vervanging van $T^2(x)$ kan weer gedaan worden door $T^3(x)$ te bepalen en hier $T^2(x)$ uit te isoleren en weer in x in te vullen. Na n keer dit proces herhalen volgt de volgende vergelijking voor x:

$$x = \frac{b_1(x)}{\beta} + \frac{b_2(x)}{\beta^2} + \cdots + \frac{b_n(x)}{\beta^n} + \frac{T^n(x)}{\beta^n}. \tag{2.1}$$

Omdat $T^n(x) \in \left[0, \frac{1}{\beta-1}\right]$ geldt dat $\lim_{n\to\infty} \frac{T^n(x)}{\beta^n} = 0$ waardoor de getalsontwikkeling van de willekeurig gekozen x de volgende sommatie wordt:

$$x = \sum_{i=1}^{\infty} \frac{b_i(x)}{\beta^i}.$$

Merk op dat een getalsontwikkeling gegeven kan worden door slechts de waarde van $b_1, b_2, \ldots$ te geven, aangezien voor elke b_i met $i = 1, 2, \ldots$ bekend is wat in de noemer staat.

2.2 Algoritmen

Een andere manier om een getalsontwikkeling te maken is met behulp van een algoritme. Ik introduceer hier twee algoritmen.

- Het Greedy algoritme: Stel $b_1, b_2, \ldots, b_n$ zijn bekend. Dan is $b_{n+1} = 1$ als:

$$\sum_{k=1}^{n} \frac{b_k}{\beta^k} + \frac{1}{\beta^{n+1}} \leq x.$$

  Als dit niet het geval is, wordt $b_{n+1} = 0$.

- Het Lazy algoritme: Stel $b_1, b_2, \ldots, b_n$ zijn bekend. Dan is $b_{n+1} = 0$ als:

$$x \leq \sum_{k=1}^{n} \frac{b_k}{\beta^k} + \sum_{k=n+2}^{\infty} \frac{1}{\beta^k}.$$

  Als dit niet het geval is, wordt $b_{n+1} = 1$.

Om bekend te raken met de algoritmen zal ik van beide algoritmen een voorbeeld uitwerken. Uit dit voorbeeld zal meteen duidelijk worden dat deze twee algoritmen een verschillende ontwikkeling geven.

Voorbeeld 2.1.
Neem $\beta = \frac{1+\sqrt{5}}{2}$. Deze β heeft de fijne eigenschap dat $\frac{1}{\beta-1} = \beta$. Nu ontwikkel ik het getal $\frac{1}{\beta^2}$ op twee manieren.

Greedy:

$$b_1 = 1 \text{ als } \sum_{k=1}^{0} \frac{b_k}{\beta^k} + \frac{1}{\beta} = \frac{1}{\beta} \leq \frac{1}{\beta^2}.$$

Dit is niet het geval, dus $b_1 = 0$.

$$b_2 = 1 \text{ als } \sum_{k=1}^{1} \frac{b_i}{\beta^k} + \frac{1}{\beta^2} = 0 + \frac{1}{\beta^2} \leq \frac{1}{\beta^2}.$$

Dit klopt, dus $b_2 = 1$.

$$b_3 = 1 \text{ als } \sum_{k=1}^{2} \frac{b_k}{\beta^k} + \frac{1}{\beta^3} = \frac{1}{\beta^2} + \frac{1}{\beta^3} \leq \frac{1}{\beta^2}.$$

Dit is niet het geval, dus $b_3 = 0$.
Voor elke volgende n blijft ongelijkheid niet waar en blijft dus elke b_{n+1} voor $n \geq 2$ gelijk aan 0. De ontwikkeling voor $\frac{1}{\beta^2}$ is met het Greedy algoritme dus de volgende geworden:

$$01000000\ldots$$

Lazy:

$$b_1 = 0 \text{ als } \frac{1}{\beta^2} \leq \sum_{k=2}^{\infty} \frac{1}{\beta^k} = 0 + \frac{1}{(\beta-1)\cdot\beta} = \frac{\beta}{\beta} = 1.$$

Dit is het geval, dus $b_1 = 0$.

$$b_2 = 0 \text{ als } \frac{1}{\beta^2} \leq \sum_{k=1}^{1} \frac{b_k}{\beta^k} + \sum_{k=3}^{\infty} \frac{1}{\beta^k} = 0 + \frac{1}{(\beta-1)\cdot\beta^2} = \frac{\beta}{\beta^2} = \frac{1}{\beta}.$$

Dit is het geval, dus $b_2 = 0$.

$$b_3 = 0 \text{ als } \frac{1}{\beta^2} \leq \sum_{k=1}^{2} \frac{b_k}{\beta^k} + \sum_{k=4}^{\infty} \frac{1}{\beta^k} = 0 + \frac{1}{(\beta-1)\cdot\beta^3} = \frac{\beta}{\beta^3} = \frac{1}{\beta^2}.$$

Dit is het geval, dus $b_3 = 0$.

$$b_4 = 0 \text{ als } \frac{1}{\beta^2} \leq \sum_{k=1}^{3} \frac{b_k}{\beta^k} + \sum_{k=5}^{\infty} \frac{1}{\beta^k} = 0 + \frac{1}{(\beta-1)\cdot\beta^4} = \frac{\beta}{\beta^4} = \frac{1}{\beta^3}.$$

Dit is niet het geval, dus $b_4 = 1$.

Ik weet tot zover dat:

$$\frac{1}{\beta^2} = \frac{0}{\beta} + \frac{0}{\beta^2} + \frac{0}{\beta^3} + \frac{1}{\beta^4} + \sum_{k=5}^{\infty} \frac{b_k}{\beta^k}. \tag{2.2}$$

Hier zijn de waarden van b_k onbekend. Ik claim dat de overige b_k allemaal gelijk aan 1 moeten zijn. Er geldt namelijk dat:

$$\frac{1}{\beta^4} + \sum_{k=5}^{\infty} \frac{1}{\beta^k} = \frac{1}{\beta^4} + \frac{1}{\beta^3} = \frac{\beta+1}{\beta^4} = \frac{\beta^2}{\beta^4} = \frac{1}{\beta^2}.$$

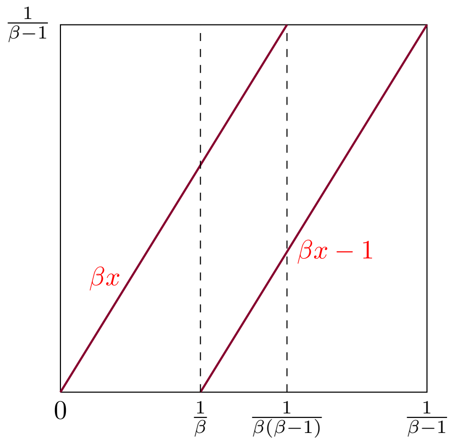
Uit (2.2) volgt dat ik voor geen enkele $k \geq 5$ een $b_k = 0$ mag kiezen, want dan geldt dat $\frac{1}{\beta^4} + \sum_{k=5}^{\infty} \frac{b_k}{\beta^k} < \frac{1}{\beta^2}$. Hieruit volgt dat de ontwikkeling voor $\frac{1}{\beta^2}$ met het Lazy algoritme dus de volgende is:

$$0001111111\ldots.$$

3 Werken met getalsontwikkelingen

3.1 Familie van getalsvoortbrengende functies.

Tot nu toe zijn er twee verschillende ontwikkelingsmethoden gegeven, namelijk de Greedy en de Lazy manier. Het verschil tussen beide is het moment waarop je kiest om een b_i gelijk aan 0 of 1 te maken. Bij Greedy maak je $b_i(x) = 0$ als $x < \frac{1}{\beta}$ en bij Lazy maak je $b_i(x) = 0$ als $x \leq \frac{1}{\beta(\beta-1)}$. De grens, die ik α noem, waar je $b_i(x) = 0$ kiest kan gevarieerd worden, waardoor je verschillende ontwikkelingen krijgt. Er zijn echter grenzen voor α.



Figuur 1: Familie van getalsvoortbrengende functies

Een specifieke functie T hangt van de waarde van α af. Voor onbekende α ziet de functie er als volgt uit:

$$T_{\alpha^-}(x) = \begin{cases} \beta x & \text{als } x < \alpha, \\ \beta x - 1 & \text{als } x \geq \alpha \end{cases} \quad \text{of } T_{\alpha^+}(x) = \begin{cases} \beta x & \text{als } x \leq \alpha, \\ \beta x - 1 & \text{als } x > \alpha. \end{cases} \tag{3.1}$$

Lemma 3.1 legt een eis op het domein waarin α gekozen mag worden.

Lemma 3.1.
Voor iedere $\alpha \in \left[\frac{1}{\beta}, \frac{1}{\beta(\beta-1)}\right]$ bestaan er twee verschillende transformaties, namelijk T_{α^-} en T_{α^+} uit (3.1), die β-ontwikkelingen genereren.

Bewijs Lemma 3.1
Bekend is dat:

$$\beta x - 1 \geq 0 \Leftrightarrow x \geq \frac{1}{\beta} \text{ en } \beta x \leq \frac{1}{\beta - 1} \Leftrightarrow x \leq \frac{1}{\beta(\beta-1)}. \tag{3.2}$$

Als $\alpha \in \left[\frac{1}{\beta}, \frac{1}{\beta(\beta-1)}\right]$, dan geldt dat $T_{\alpha_\pm}(x) \in \left[0, \frac{1}{\beta-1}\right]$ voor alle $x \in \left[0, \frac{1}{\beta-1}\right]$. De twee verschillende ontwikkelingen ontstaan door de verschillende functies

T_{α^-} en T_{α^+} te gebruiken. Als ik T_{α^-} gebruik zal ik zijn ontwikkeling aangeven met een rij $b_i^-(x)$ en als ik T_{α^+} gebruik $b_i^+(x)$. Er geldt dan:

$$b_1^-(x) = \begin{cases} 0 & x < \alpha, \\ 1 & x \geq \alpha \end{cases} \quad \text{en } b_1^+(x) = \begin{cases} 0 & x \leq \alpha, \\ 1 & x > \alpha \end{cases} . \tag{3.3}$$

Voor verdere $n \geq 1$ volgt dan $b_n^{\pm}(x) = b_1^{\pm}\left(T_{\alpha^\pm}^{n-1}(x)\right)$. De twee verschillende ontwikkelingen voor x worden dan net zo bepaald als in hoofdstuk 2.1.

3.2 Bepalen van verschillende getalsontwikkelingen

Figuur 1 heeft drie delen, namelijk:

- $E_0 = [\,0, \frac{1}{\beta}\,)$,
- $S = \left[\frac{1}{\beta}, \frac{1}{\beta(\beta-1)}\right]$,
- $E_1 = (\,\frac{1}{\beta(\beta-1)}, \frac{1}{\beta-1}\,]$.

Voor deze gebieden gelden twee fijne eigenschappen, namelijk:

$$\forall x : x \in E_0 \cup S \Rightarrow \beta x \in \left[0, \frac{1}{\beta - 1}\right] \quad \text{en } \forall x : x \in E_1 \cup S \Rightarrow \beta x - 1 \in \left[0, \frac{1}{\beta - 1}\right].$$

Met behulp van algoritme 3.1 zijn verschillende getalsontwikkelingen te bepalen. In dit algoritme wordt gebruik gemaakt van de gebieden E_0, S, E_1.

Algoritme 3.1.
Stap 1. Laat $x = x_1$ en laat $n = 1$.
Stap 2. Bepaal in welk van de drie gebieden x_n ligt.
Stap 3. Afhankelijk van het gebied waar x_n ligt, volg stap 3a, 3b of 3c.

a Als $x_n \in E_0$, dan $b_n(x) = 0$ en $x_{n+1} = \beta x_n = T_{b_n} \circ T_{b_{n-1}} \circ \cdots \circ T_{b_1}(x_1)$.

b Als $x_n \in E_1$, dan $b_n(x) = 1$ en $x_{n+1} = \beta x_n - 1 = T_{b_n} \circ T_{b_{n-1}} \circ \cdots \circ T_{b_1}(x_1)$.

c Als $x_n \in S$, dan kies:
of $b_n(x) = 0$ en $x_{n+1} = \beta x_n = T_{b_n} \circ T_{b_{n-1}} \circ \cdots \circ T_{b_1}(x_1)$,
of $b_n(x) = 1$ en $x_{n+1} = \beta x_n - 1 = T_{b_n} \circ T_{b_{n-1}} \circ \cdots \circ T_{b_1}(x_1)$.

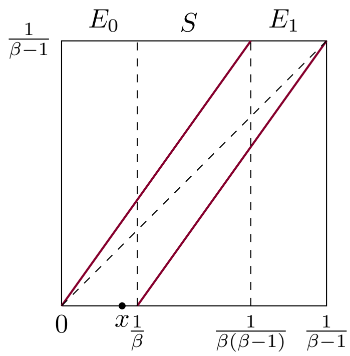
Stap 4. Laat $n := n + 1$ en ga terug naar stap 2.

Ongeacht de keuze van b_n is elke x_{n+1} met dezelfde formule te bepalen, namelijk $x_{n+1} = \beta x_n - b_n$. Met behulp van deze eigenschap valt dan een algemene formule voor de ontwikkeling van x te bepalen, namelijk:

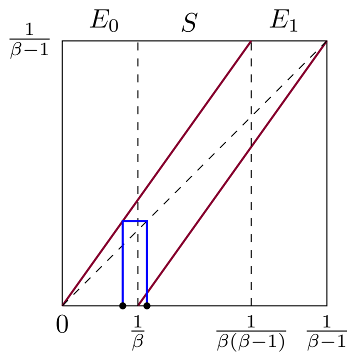
$$\begin{aligned} x = x_1 &= \frac{b_1}{\beta} + \frac{T_{b_1}(x_1)}{\beta} = \frac{b_1}{\beta} + \frac{b_2}{\beta^2} + \frac{T_{b_2} \circ T_{b_1}(x_1)}{\beta^2} \\ &= \cdots = \frac{b_1}{\beta} + \cdots + \frac{b_n}{\beta^n} + \frac{T_{b_n} \circ \cdots \circ T_{b_1}(x_1)}{\beta^n}. \end{aligned}$$

Zoals in het algoritme te zien is zijn er twee keuzes voor b_n als x_n in deel S ligt, namelijk $b_n = 0$ of $b_n = 1$. Ik zoek hier echter niet naar een specifieke ontwikkeling, maar naar meerdere mogelijke getalsontwikkelingen. Hoe deze verschillende ontwikkelingen gevonden worden zal ik nu laten zien met behulp van voorbeeld 3.1.

Voorbeeld 3.1.
Stel ik neem een punt $x = x_1$ zoals gegeven in figuur 2. Dit punt ligt in E_0, dus met stap 3a volgt dat $b_1(x) = 0$. Om b_2 te bepalen moet ik bepalen waar $x_2 = T_0(x_1)$ ligt. Dit doe ik door een verticale lijn vanuit punt x_1 te tekenen tot ik op lijn βx kom, waarna ik horizontaal naar de lijn $y = x$ loop. Het snijpunt van deze horizontale lijn met de lijn $y = x$ geeft de waarde voor x_2. Zoals te zien in figuur 3, ligt $x_2 \in S$.

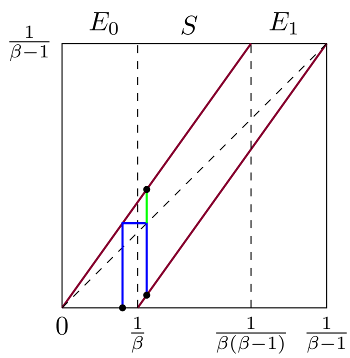


Figuur 2

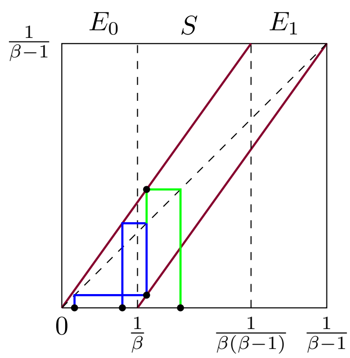


Figuur 3

Omdat $x_2 \in S$ zijn er met stap 3c twee keuzes. Deze keuzes zijn met twee kleuren in figuur 4 gegeven. Als ik $b_2 = 0$ kies, dan kan ik de positie van x_3 bepalen door $x_3 = T_0(x_2)$, oftewel een verticale lijn vanuit punt x_2 tekenen tot ik bij lijn βx kom, daarna horizontaal naar het snijpunt met de lijn $y = x$ waar ik de waarde voor x_3 vind. Als ik $b_2 = 1$ kies, dan kan ik de positie van x_3 analoog bepalen maar dan neem ik de verticale lijn vanuit x_2 tot de lijn $\beta x - 1$. Er ontstaan nu twee verschillende ontwikkelingen. De positie van x_3 zal verschillen afhankelijk van de keuze van b_2. In figuur 5 is te zien welke positie x_3 in beide gevallen aanneemt.



Figuur 4



Figuur 5

Hier is duidelijk te zien dat de waarde voor x_3 verschilt. Als ik $b_2 = 1$ kies, dan ligt $x_3 \in E_0$, maar als ik $b_2 = 0$ kies, dan ligt $x_3 \in S$. In het eerste geval geldt met stap 3a dat $b_3 = 0$. In het andere geval geldt $x_3 \in S$, dus heb ik met stap 3c twee keuzes voor b_3. Dit proces kan herhaald worden, waardoor telkens meerdere verschillende ontwikkelingen ontstaan, omdat ik telkens weer keuzes krijg als een $x_n \in S$.

Als ik de blauwe lijn volg, weet ik dat geldt $b_1 = 0, b_2 = 1, b_3 = 0$, dus deze ontwikkeling van x begint met $010\ldots$. Als ik bij de splitsing de groene lijn volgt weet ik dat geldt dat $b_1 = 0, b_2 = 0$, dus deze ontwikkeling van x begint met $00\ldots$.

Elke keer als een x_n in S komt heb ik dus keuze uit twee opties voor bijbehorende b_n. Een schematische weergave waarin te zien is dat er dankzij deze opties telkens meer ontwikkelingen ontstaan is te zien in sectie 4.3.

4 Hoofdstelling

Dit project heeft als belangrijkste stelling de onderstaande.

Stelling 4.1.
Als $1 < \beta < \frac{1+\sqrt{5}}{2}$, dan heeft elke $x \in \left(0, \frac{1}{\beta-1}\right)$ overaftelbaar oneindig veel β-ontwikkelingen.

Uit paragraaf 3.2 volgt dat als een x_n in S komt, er twee verschillende takken ontstaan. Paragraaf 4.1 zal laten zien waarom $\frac{1+\sqrt{5}}{2} \leq \beta < 2$ uitgesloten is. Paragraaf 4.2 zal laten zien dat er voor elke $x \in \left(0, \frac{1}{\beta-1}\right)$ een n aantal iteraties is zodat $x_n \in S$. Hierna zal paragraaf 4.3 het bewijs van de hoofdstelling geven.

4.1 Fouten bij andere waarden van β

Verschillende ontwikkelingen voor dezelfde waarde x worden gevonden wanneer een iteratie van x in het gebied S komt. Er zijn echter bepaalde waarden voor $1 < \beta < 2$ waarin er geen enkele iteratie van x in S komt. De waarde van β moet gelimiteerd worden, zodat het niet mogelijk is vanuit E_0 naar E_1 te springen zonder in S te komen. Er moet dus gelden dat $\beta \cdot \frac{1}{\beta} < \frac{1}{\beta(\beta-1)}$. Eenvoudige algebra geeft dat deze ongelijkheid geldt als $\beta < \frac{1+\sqrt{5}}{2}$. Er volgen nu twee voorbeelden. Voorbeeld 4.1 laat zien wat gebeurt als $\beta = \frac{1+\sqrt{5}}{2}$ en voorbeeld 4.1 laat zien wat gebeurt als $\beta > \frac{1+\sqrt{5}}{2}$.

Voorbeeld 4.1.
Neem aan dat $\beta = \frac{1+\sqrt{5}}{2}$. Merk op dat voor deze β geldt dat $\frac{1}{\beta(\beta-1)} = 1$. Neem $x = x_1 = \frac{1}{\beta}$. Merk op dat $x_1 \in S$. Nu mag ik met behulp van stap 3c van algoritme 3.1 een keuze voor b_1 maken.
Stel dat ik $b_1 = 1$ kies. Dan geldt dat $x_2 = T_1(x_1) = T_1\left(\frac{1}{\beta}\right) = \beta \cdot \frac{1}{\beta} - 1 = 0$. Omdat $0 \in E_0$ en wegens stap 3a geldt dat $x_3 = T_0(0) = 0$, volgt dat elke $b_n = 0$ voor $n \geq 2$.
Stel dat ik $b_1 = 0$ kies. Dan geldt dat $x_2 = T_0(x_1) = T_0\left(\frac{1}{\beta}\right) = \beta \cdot \frac{1}{\beta} = 1$. Nu geldt dat $x_2 \in S$, dus mag ik weer een keuze maken voor b_2.
Stel dat ik $b_2 = 0$ kies. Dan geldt dat $x_3 = T_0 \circ T_0(x_1) = T_0(1) = T_0\left(\frac{1}{\beta(\beta-1)}\right) = \beta \cdot \frac{1}{\beta(\beta-1)} = \frac{1}{\beta-1}$. Omdat $\frac{1}{\beta-1} \in E_1$ en wegens stap 3b geldt dat $x_4 = T_1\left(\frac{1}{\beta-1}\right) = \frac{1}{\beta-1}$, volgt dat elke $b_n = 1$ voor $n \geq 3$.
Stel dat ik $b_2 = 1$ kies. Dan geldt dat $x_3 = T_1 \circ T_0(x_1) = T_1(1) = T_1\left(\frac{1}{\beta(\beta-1)}\right) = \beta \cdot \frac{1}{\beta(\beta-1)} - 1 = \frac{1}{\beta}$. Nu geldt dat $x_3 = x_1$, dus zijn we terug bij het begin. Eenvoudig zien we hier dat elke iteratie een extra ontwikkeling voor x toevoegd. Het aantal ontwikkelingen groeit dus lineair evenredig met het aantal iteraties. Dit aantal is aftelbaar veel en niet overaftelbaar veel. Dus voor deze waarde van β geldt stelling 4.1 niet.

Voorbeeld 4.2.
Neem aan dat $\frac{1+\sqrt{5}}{2} < \beta < 2$. Merk op dat voor deze waarden van β geldt dat $\beta < \beta^2 - 1$. Deze eigenschap zal ik tweemaal gebruiken.
Ik zoek nu een waarde voor $x = x_1 \in E_0$ zodanig dat $T_0(x_1) \in E_1$ en $T_1 \circ T_0(x_1) = x_1$. Ik claim dat $x = \frac{1}{\beta^2-1}$ hieraan voldoet.
Voor de gekozen grenzen van β geldt dat $x = \frac{1}{\beta^2-1} < \frac{1}{\beta}$. Hieruit volgt dat $x_1 \in E_0$, dus met stap 3a van algoritme 3.1 volgt dat $x_2 = T_0(x_1) = \beta \cdot \frac{1}{\beta^2-1} = \frac{\beta}{\beta^2-1}$.
Voor de gekozen grenzen van β geldt dat $\frac{\beta}{\beta^2-1} > \frac{1}{\beta(\beta-1)}$ (want $\beta < \beta^2 - 1$), dus $x_2 \in E_1$.
Nu geldt dat $x_2 \in E_1$, dus met stap 3b van algoritme 3.1 volgt dat $x_3 = T_1 \circ T_0(x_1) = T_1\left(\frac{\beta}{\beta^2-1}\right) = \beta \cdot \frac{\beta}{\beta^2-1} - 1 = \frac{\beta^2}{\beta^2-1} - \frac{\beta^2-1}{\beta^2-1} = \frac{1}{\beta^2-1} = x$.
Ik heb nu dus een waarde van x gevonden waar maar 1 mogelijke ontwikkeling voor is, namelijk $010101\ldots$.

4.2 Een eindig aantal iteraties

<u>Lemma</u> 4.1.
Voor elke $x \in \left(0, \frac{1}{\beta}\right)$ geldt dat er een $n \in \mathbb{N}$ is zodanig dat $T_0^n(x) \in S$ en $T_0^k(x) \in E_0$ voor alle $0 < k < n$.

Omdat het figuur symmetrisch is, is het bewijzen van dit lemma genoeg om te concluderen dat er voor elke $x \in \left(0, \frac{1}{\beta-1}\right)$ een $n \in \mathbb{N}$ is zodat $x_n \in S$.

<u>Bewijs Lemma 4.2</u>
Er zijn twee manieren waarop $T_0^n(x) \notin S$ voor alle n, namelijk:

1. voor een gegeven x bestaat een n zodat $T_0^n(x)$ van E_0 naar E_1 springt (oftewel $T_0^{n-1}(x) \in E_0$ en $T_0^n(x) \in E_1$),
2. voor een gegeven x blijft $T_0^n \in E_0$ voor alle n.

Ik zal laten zien dat beide gevallen niet mogelijk zijn, waaruit ik kan concluderen dat voor elke x er een eindige n bestaat zodat $T_0^n(x) \in S$.

1. Omdat de waarde van n waar de sprong van E_0 naar E_1 voorkomt niet uitmaakt, kan ik $T_0^{n-1}(x')$ vervangen door x en $T_0^n(x') = T_0(x)$. Stel dat $x \in E_0$ en $T_0(x) \in E_1$. Er geldt dan dat $x < \frac{1}{\beta}$ en $\beta x > \frac{1}{\beta(\beta-1)}$. Ik zoek dus een x die als volgt begrensd is:
$$\frac{1}{\beta^2(\beta-1)} < x < \frac{1}{\beta}$$
Dit kan alleen als $\frac{1}{\beta^2(\beta-1)} < \frac{1}{\beta}$. Deze ongelijkheid geldt als $\beta > \frac{1+\sqrt{5}}{2}$. Dit is in tegenspraak met de aanname dat $\beta < \frac{1+\sqrt{5}}{2}$, dus een dergelijke x bestaat niet.

2. Voor elke willekeurige $x \in \left(0, \frac{1}{\beta}\right)$ geldt dat $T_0^n(x) = \beta^n x$. Elke iteratie die optreedt bij verhogen van de n geeft een strikt stijgende functie, wat betekent dat voor een gegeven n moet gelden dat $T_0^n(x) \geq \frac{1}{\beta}$, oftewel dat $T_0^n(x) \notin E_0$.
   Er is dus geen x waarvoor $T_0^n(x) \in E_0$ voor alle n.

$\square$

Hieruit concludeer ik dat voor elke $x \in \left(0, \frac{1}{\beta}\right)$ geldt dat er een n is zodanig dat $T_0^n(x) \in S$ en voor elke $x \in \left(\frac{1}{\beta(\beta-1)}, \frac{1}{\beta-1}\right)$ geldt dat er een m is zodanig dat $T_1^m(x) \in S$.

4.3 Bewijs van de hoofdstelling

Ik zal nu beginnen met het bewijzen van stelling **??**. Deze stelling luidt:

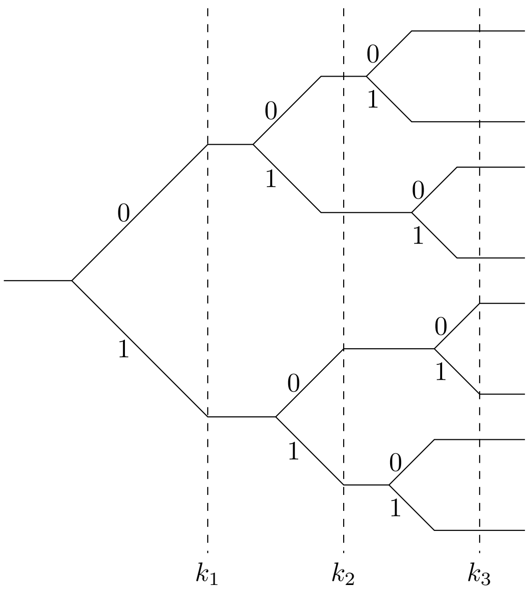
Stelling 4.1.
Als $1 < \beta < \frac{1+\sqrt{5}}{2}$, dan heeft elke $x \in \left(0, \frac{1}{\beta-1}\right)$ overaftelbaar oneindig veel β-ontwikkelingen.

Bewijs Stelling 4.1
Laat β en x binnen de intervallen zoals gegeven liggen. Een ontwikkeling van willekeurige x begint in een van de delen E_0, S, E_1. Ik weet uit Lemma 4.2 dat ongeacht in welk deel x begint, de ontwikkeling van x na een eindig aantal iteraties in gebied S komt. Wanneer dit gebeurt ontstaan twee verschillende ontwikkelingen. Als ik verder ga met een van deze ontwikkelingen, zal er voor deze ontwikkeling weer na een eindig aantal stappen een moment komen waar de ont- wikkeling in S komt. Hier wordt de ontwikkeling weer in twee delen gesplitst. Dit proces herhaalt zich.

Het moment dat elke splitsing zelf weer splitst is niet voor elke tak gelijk. In het algemeen geldt wel dat voor elke $n \geq 1$ er een minimale k_n is zodat, met behulp van algoritme 3.1, na k_n stappen er minstens 2^n verschillende beginstukken van ontwikkelingen van x ontstaan zijn. In de limiet van n naar oneindig zal het aantal verschillende beginstukken overaftelbaar oneindig veel worden.

Dus met deze gekozen x en β ontstaan overaftelbaar oneindig veel verschillende getalsontwikkelingen. □

Figuur 6 laat schematisch zien hoe de minimale k_n bepaald worden voor verschillende waarden van n. Op de horizontale lijnen zijn de bijbehorende $x_i \notin S$. Een splitsing treedt op als een bepaalde $x_i \in S$.



Figuur 6

5 Verdieping: Hoe kunnen β-ontwikkelingen gebruikt worden in een telefoon.

AD-omzetting is een term die gebruikt wordt om analoge naar digitale signalen om te zetten. Dit houdt in dat analoge signalen (denk aan tekst, geluid, spraak) gecodeerd worden naar een bepaalde digitale code (denk aan binaire code, morse code, rook en licht signalen). Tegenwoordig worden de digitale signalen vooral omgezet in een reeks van nullen en enen. Om dit te doen zijn verschillende algoritmen bedacht. Bij ontwerp van zulke algoritmen zijn er twee stromingen waar aandacht naar uitgaat. Als eerste moet het algoritme er mooi uitzien en moet het algoritme eenvoudig werken (het algoritme moet dus niet oneindig lang duren of zoveel stappen uitvoeren dat het te lang duurt voor een computer om het algoritme af te lopen). Ten tweede moet geprobeerd worden een algoritme te maken dat zijn eigen fouten kan herstellen. Dit is handig omdat computers die erg lange codes moeten maken wel eens een foutje kunnen maken. Als dit foutje niet herstelt kan worden zal de code nutteloos zijn. Het herstellen van foutjes wordt ook wel robuustheid van een algoritme genoemd.

Bestaande AD-omzetters kunnen in twee grote klassen opgedeeld worden, namelijk de 'oversampled'-omzetters en de Nyquist-rate omzetters. De oversampled-omzetters zal ik buiten beschouwing laten. De Nyquist-rate omzetter bestaat uit twee delen, een versterker en een zogenaamde weerstand-condensator filter (RC-filter in engels). Wat dit precies inhoudt probeer ik aan de hand van een voorbeeld duidelijk te maken.

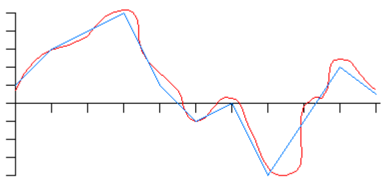
Stel dat je een voicemail wil inspreken. Je praat dan in een telefoon met analoge signalen. Je telefoon kan echter alleen digitale signalen doorsturen naar een antwoordapparaat in een ander huis. Het analoge signaal (de geluidsgolf) moet dus omgezet worden naar een digitaal signaal. De geluidsgolf bestaat uit een frequentie en een amplitude. De frequentie bepaalt de hoogte van de klanken, hoge klanken hebben een grote frequentie en lage klanken een kleine. De amplitude bepaalt het volume van het geluid, oftewel hoe hard je klinkt. Iemand die fluistert geeft dus golven met een zeer lage amplitude af terwijl iemand die schreeuwt een grote amplitude heeft. Deze golven gaan de telefoon in, waar een versterker de amplitude vermenigvuldigt met een factor. Kleine volumeverschillen worden hierdoor groter omdat alle amplituden die in je telefoon komen met dezelfde factor vermenigvuldigd worden. Hierdoor herkent je telefoon kleine volumeverschillen beter.
Op je voicemail wil je natuurlijk niet elk klein geluidje terughoren. Achtergrondgeluiden zoals je broertje die met een pen speelt of de vogels die buiten zingen zijn niet belangrijk. Daarom worden alleen golven met een amplitude binnen een bepaald interval toegelaten. Deze test zit ook verwerkt in de versterker. Omdat geen enkele machine 100% rendement heeft, ontstaat er een beetje ruis tijdens dit proces. Het rendement in dit proces ligt in het meten van de amplitude. De waarde van de amplitude wordt altijd afgerond, dus het geluid blijft niet precies gelijk. Hierdoor klinkt een voicemail niet exact hetzelfde als wanneer je rechtstreeks tegen iemand praat. De hoeveelheid ruis die ontstaat hangt af van de versterker, omdat hier de amplitude gemeten wordt en vermenigvuldigd wordt met een factor wat afrondfouten meebrengt.

De golven (met vergrootte amplituden) die doorgelaten zijn door de versterker worden daarna getest op frequentie. Een voicemail hoeft geen geluiden door te geven die zo hoog of laag zijn dat het menselijk oor ze niet kunnen horen. De frequentie wordt dus ook getest en doorgelaten als deze in een bepaald interval ligt. Dit wordt gedaan door de RC-filter.

Nu is het tijd om deze golven om te zetten naar een reëel getal. Er zijn twee factoren die in te stellen zijn op je telefoon om deze analoge signalen om te zetten, namelijk:

- Hoeveel metingen je telefoon per seconde doet. Dit wordt ook wel 'Sampling rate' genoemd.
- De nauwkeurigheid van de meting. Hierbij moet gedacht worden aan hoe klein de verschillen moeten zijn tussen verschillende amplituden.

Stel dat we bij het inspreken de volgende analoge golf krijgen, waar op de verticale as de amplitude in volt is uitgedrukt en op de horizontale as de tijd:



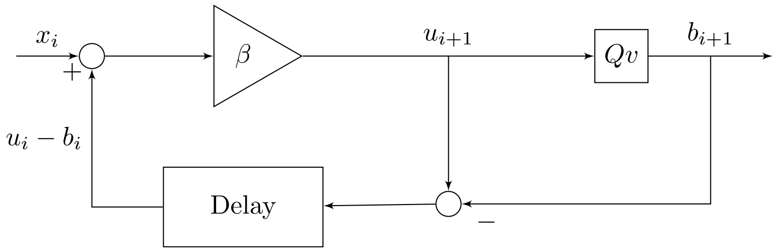
Figuur 7: De blauwe lijn geeft de verdeling in waarden van amplitude aan die bepaald worden uit de golf. [1]

Hier is de sampling rate vrij klein en de nauwkeurigheid ook, er zijn namelijk maar 10 verschillende waarden die de amplitude aan kan nemen. Door verschillen in de dichtsbijzijnde waarde op de y-as met de waarden van de frequentie ontstaat de ruis waar ik het eerder over had. Als ik nu op beide assen kleinere intervallen neem, zal het signaal telkens nauwkeuriger worden. Voor telefoons wordt een sampling rate van 8000 gebruikt (dus er wordt 8000 keer per seconde een meting gedaan) en het aantal intervallen op de amplitude zijn er ongeveer 65000. Bij elke meting wordt nu gekeken bij welk van de 65000 punten de amplitude het meest in de buurt ligt. Een meting krijgt dan een waarde tussen 0 en 65000.

Het hele proces wat nu doorlopen is vanaf een analoog signaal tot een getal tussen 0 en 65000, wordt gedaan door een Nyquist-rate omzetter is dus een analoog signaal omgezet naar een reeks reële getallen. Deze getallen moeten dan nog omgezet worden naar binaire codes. Dit is het punt waarop algoritmes ingezet worden om deze binaire codes te maken.

De β-encoder is een van deze algoritmen. Het fijne aan dit algoritme is dat het zijn eigen fouten herstelt. De precizie van dit algoritme vervalt exponentieel, maar langzamer dan de algoritmen die normaal gebruikt worden. De β-encoder werkt alleen op getallen x in het interval $\left[0, \frac{1}{\beta-1}\right]$. Omdat de getallen x die het systeem inkomen tussen 0 en 65000 liggen, zal de encoder eerst dit interval moeten verkleinen naar een geschikt interval. Om dit te doen kan de encoder

bijvoorbeeld elke inkomende x vermenigvuldigen met de factor $\frac{1}{65000(\beta-1)}$.
Nu ontstaat een reeks getallen op een geschikt interval voor het algoritme. Een elektrisch apparaat werkt echter niet met behulp van een algoritme, maar met behulp van een circuit. In een circuit wordt een elektrische stroom gestuurd, waarna er bepaalde handelingen met het stroompje uitgevoerd worden. De handelingen lijken erg op de handelingen van het algoritme. Het circuit ziet er als volgt uit:



Figuur 8: Het elektrische circuit om ontwikkelingen te maken. [3]

Ik zal nu een toelichting geven van het circuit uit figuur 8. De bepaalde x, nadat x vermenigvuldigd is met $\frac{1}{65000(\beta-1)}$ zodat $x \in \left[0, \frac{1}{\beta-1}\right]$, is een elektrische stroom die in het circuit gestuurd wordt. De waarde van x is de spanning van de stroom. De term x_i is de spanning van de stroom die het circuit ingestuurd wordt. Hier is $x_1 = x$ en $x_2, x_3, \ldots, x_n = 0$. Deze spanning wordt dan naar een knooppunt gestuurd. In dit knooppunt wordt een bepaalde spanning opgeteld. Wat deze spanning is zal ik later verklaren.
De driehoek in het figuur is een vermenigvuldiger. Hier wordt een bepaalde stroom vermenigvuldigd met een gegeven factor, die in mijn geval β is. De waarde van de spanning na het vermenigvuldigen wordt aangegeven met $u_{i+1} = \beta x_i$.
Dit stroompje wordt daarna naar een zogenaamde comparator gestuurd (aangegeven met een vierkant met Q_v erin). Hier wordt de waarde van de spanning van de stroom gemeten en vergeleken met de waarde v, waarna de waarde b_{i+1} gelijk aan 0 of 1 wordt. Je zou verwachten dat de waarde van v in hetzelfde interval ligt als de waarde van α uit sectie 3.1, waardoor b_{i+1} net zo bepaald wordt als in (3.3). Er is echter een belangrijk verschil tussen het moment waarop b_{i+1} bepaald wordt in (3.3) en het moment waarop b_{i+1} bepaald wordt in het circuit. In het circuit wordt x_{i+1} namelijk eerst vermenigvuldigd met β voordat het vergeleken wordt met een bepaalde waarde v, terwijl in (3.3) alleen x vergeleken wordt met een waarde α. De bepaling van b_{i+1} volgt dus uit (3.3) door zowel x als α met β te vermenigvuldigen. Er geldt dan:

$$b_{i+1}(\beta x_{i+1}) = \begin{cases} 0 & \beta x_{i+1} < \beta\alpha, \\ 1 & \beta x_{i+1} \geq \beta\alpha \end{cases}.$$

Als ik nu $v = \beta\alpha$ kies, volgt wegens het interval van α uit 3.1 dat $v \in \left[1, \frac{1}{\beta-1}\right]$.

Omdat $\beta x_{i+1} = u_{i+1}$, kan ik de bepaling van b_{i+1} ook schrijven als:

$$b_{i+1}(u_{i+1}) = Q_v(u_{i+1}) = \begin{cases} 0 & u_{i+1} < v, \\ 1 & u_{i+1} \geq v \end{cases} .$$

Merk op dat de waarde van v ingesteld kan worden door de fabrikant. Deze waarde voor b_{i+1} wordt dan ook de output van het circuit en levert na meerdere iteraties een ontwikkeling voor x op.
Nadat b_{i+1} bepaald is, moet b_{i+2} bepaald worden. Uit algoritme 3.1 weet ik dat $x_{n+1} = \beta x_n - b_n$. In dit circuit is u_{n+1} gelijk aan βx_{n+1}. De waarde van deze twee stromen worden echter niet tegelijkertijd bepaald. Om ze toch gelijk door te sturen is een delay in het circuit gezet. De stroom u_{i+1} bereikt eerder de delay dan de stroom b_{i+1}. De delay zorgt er dan voor dat de stroom u_{i+1} niet wordt doorgegeven voordat de waarde van stroom b_{i+1} ervanaf gehaald is. Nadat dit gedaan is wordt de stroom $u_{i+1} - b_{i+1}$ bij $x_{i+1} = 0$ opgeteld, waarna het circuit weer opnieuw begint.
De fabrikant kan ook instellen hoeveel iteraties er gedaan moeten worden. Eerder hebben we de mogelijke spanning van de analoge signalen in 65000 stukjes gedeeld. Het aantal bits dat per spanning x bepaald moet worden, moet zodanig bepaald worden dat geldt dat #bits $< \log_2(65000)$. Dit geldt als het aantal bits minstens 16 is.

Het hele proces wat nu beschreven is lijkt erg op de manier zoals nu binaire codes gemaakt worden. Het verschil is dat de vermenigvuldiger nu op β staat in plaats van 2. Hierdoor veranderen tevens ook de grenzen van v. Dit verschil is cruciaal voor de robuustheid van het proces. Er geldt namelijk dat een binaire code maar op één manier gemaakt kan worden in het 2-tallig stelsel. Dit houdt in dat er maar 1 mogelijkheid is van een ontwikkeling van een willekeurige input. Het β-tallig stelsel kan echter oneindig veel verschillende ontwikkelingen maken van een willekeurige input. Hierdoor is het niet erg wanneer een computer een foutje maakt, omdat dit foutje doorberekent wordt in het systeem (namelijk het stroompje met waarde u_{i+1}) waarna een nog steeds correcte (maar wel andere) ontwikkeling ontstaat. Het β-tallig stelsel kan dus zijn eigen foutjes verbeteren omdat er oneindig veel verschillende ontwikkelingen per input zijn, waaruit volgt dat het β-tallig stelsel robuuster is dan het 2-tallig stelsel.

Het omzetten van analoge signalen naar digitale signalen is dus een enorm proces. We weten nu dat er per seconde 8000 getallen ontwikkelt worden, elk met minstens 16 bits. Nadat een digitaal signaal is ontstaan bestaan er soortgelijke circuits met DA-omzetting, zodat iemand aan de andere kant van de lijn jouw stem kan horen. Hoe dit proces werkt laat ik buiten beschouwing, daar het niet relateert aan mijn scriptie.

Gebruikte literatuur

Referenties

[1] M. Brain, How Analog and Digital Recording Works, http://electronics.howstuffworks.com/analog-digital.htm, (2011).

[2] K. Dajani, C. Kraaikamp, Ergodic Theory of Numbers, *Mathematical Association of America*, Washington, DC (2002).

[3] I. Daubechies, R.A. DeVore, C.S. Güntürk, V.A. Vaishampayan, A/D conversion with imperfect quantizers, *IEEE Trans. Inform. Theory*, **52**(3):874-885 (2006).

[4] I. Daubechies, C.S. Güntürk, Y. Wang, Ö. Yilmaz, The Golden Ratio Encoder, *IEEE Trans. Inform. Theory*, **56**(10):5097-5110 (2010).

[5] C. Gammel, Switching Regulators and Switching Noise, http://chrisgammell.com/switching-regulators-and-switching-noise/, (2009).

[6] J. Widder, Y. Zhao, Understanding output filters for Class-D amplifiers, http://www.eetimes.com/design/audio-design/4015847/Understanding-output-filters-for-Class-D-amplifiers, (2008).

[7] Digital Audio- Digital Sound Primer, http://www.zytrax.com/tech/audio/digital-sound.html, (2013).