

## 1 Osnove

### 1.1 Ponovitev logaritmov

- $\log_a x = \frac{\log_b x}{\log_b a}$
- $\log_b \left(\frac{x}{y}\right) = \log_b x - \log_b y$
- $x = b^y \implies \log_b x = y$
- $\log_2 x = \log x$
- $0\log 0 = 0$

### 1.2 Bayesova formula

$$\begin{aligned} P(H_i|A) &= \frac{P(H_i)P(A|H_i)}{P(A)} \\ &= \frac{P(H_i)P(A|H_i)}{\sum_{k=1}^n P(H_k)P(A|H_k)} \end{aligned}$$

### 1.3 Lastna informacija

Opisuje dogodek, ki se je zgodil:

$$I_i = \log_2\left(\frac{1}{p_i}\right) = -\log_2(p_i)$$

### 1.4 Entropija

je povprecie vseh lastnih informacij:

$$H(X) = \sum_{i=1}^n p_i I_i = -\sum_{i=1}^n p_i \log_2 p_i$$

Vec zaporednih dogodkov neodvisnega vira:  $X^l = X \times \cdots \times X \rightarrow H(X^l) = lH(X)$ .

## 2 Kodi

### 2.1 Uvod

Povprečna dolžina k.z.

$$L = \sum_{i=1}^n p_i l_i$$

### 2.2 Tipi kodov

- optimalen** - ce ima najmanjso možno dolžino kodnih zamenjav
- idealen** - ce je povprečna dolžina kod-nih zamenjav enaka entropiji
- enakomeren** - ce je dolžina vseh kodnih zamenjav enaka
- enoznacen** - ce lahko poljuben niz znakov dekodiramo na en sam način
- trenuten** - ce lahko osnovni znak dekodiramo takoj, ko sprejmemo celotno kodno zamenjavo

### 2.3 Kraftova neenakost

obstaja trenutni kod, iff

$$\sum_{i=1}^n r^{-l_i} \leq 1$$

### 2.4 Povp. dolžina, ucinkovitost

Najkrajše kodne zamenjave:

$$H_r(X) = L \rightarrow l_i = \lceil -\log_r p_i \rceil$$

Ucinkovitost:

$$\eta = \frac{H(X)}{L \log_2 r}, \eta \in [0, 1]$$

Kod je **gospodaren**, ce je *L* znotraj:

$$H_r(X) \leq L < H_r(X) + 1$$

kjer je *H<sub>r</sub>*(*X*):

$$H_r(X) = -\sum_{i=1}^n \frac{\log p_i}{\log_r} = \frac{H(X)}{\log_r}$$

### 2.5 Shannonov prvi teorem

Za nize neodvisnih znakov dolžine *n* obstajajo kodi, za katere velja:

$$\lim_{n \rightarrow \infty} \frac{L_n}{n} = H(X)$$

pri cemer je *H*(*X*) entropija vira *X*.

### 2.6 Huffmanov kod

Veljati mora:

$$n = r + k(r - 1), k \geq 0$$

### 2.7 Kod Lempel-Ziv (LZ77)

Gre za kodiranje na osnovi slovarja **Kodiranje**: uporablja drseca okna, znaki se premikajo iz desne na levo. Referenca je podana kot trojček(odmik, dolžina, naslednji znak): npr. (0, 0, A) - ni ujemanja, (4, 3, B) - 4 znake nazaj se ponovi 3 znakovni podniz, ki se nato zaključi s B.  
**dekodiranje**: sledimo kodnim zamenjavam

### 2.8 Kod Lempel-Ziv (LZW)

Osnovni slovar je podan in ga sporti doponju-jemo. Algoritem za **kodiranje**:

```
N = ""
ponavljaj:
    preberi naslednji znak z
    ce je [N,z] v slovarju:
        N = [N, z]
    drugace:
        izpisi indeks k niza N
        dodaj [N, z] v slovar
        N = z
izpisi indeks k niza N
```

Algoritem za **dekodiranje**:

```
preberi indeks k
poisci niz N, ki ustreza indeksu k
izpisi N
L = N
ponavljaj:
    preberi indeks k
    ce je k v slovarju:
        poisci niz N
    drugace:
        N = [L, L(1)]
    izpisi N
    v slovar dodaj [L, N(1)]
    L = N
```

LZW doseže optimalno stiskanje, približa se entropiji.

### 2.9 Verizno kodiranje ali RLE (run length encoding)

Namesto originalnih podatkov, sharnjujemo dolžino verige (fffeef → 3f2e1f).

### 2.10 Kompresijsko razmerje

$$R = C(M)/M$$

## 3 Kanali

### 3.1 Diskretni kanal brez spomina

Kanal je definiran kot množica **pogojnih verjetnosti**

$$p(y_j|x_i).$$

Pogojna verjetnost nam pove verjetnost za do-godek *y<sub>j</sub>* na izhodu iz kanala, ce je na vходу v kanal dogodek *x<sub>i</sub>*.

$$\sum_j p(y_j|x_i) = 1.$$

Kanal popolnoma podamo z *r* × *s* pogojnimi ver-jetnostmi.

#### 3.1.1 Binarni simetrični kanal (BSK)

Napaka kanala je *p*, saj se z verjetnostjo *p* znak prenese v napacnega.

$$P_k = \begin{pmatrix} 1-p & p \\ p & 1-p \end{pmatrix}$$

### 3.2 Pogojna entropija

Pogojna entropija spremenljivke *Y* pri znanem *X* se zapise kot *H*(*Y*|*X*). Vzemimo, da se je zgodil dogodek *x<sub>i</sub>* ∈ *X*. Entropija dogodka *Y* je potem

$$H(Y|x_i) = -\sum_{j=1}^s p(y_j|x_i) \log(p(y_j|x_i)).$$

Velja: 0 ≤ *H*(*Y*|*x<sub>i</sub>*).

Ce pa o dogodku *X* vemo le da se je zgodil, se lahko spomnemo na vis in uporabimo **vezano verjetnost** dogodkov *X* in *Y*, ki pravi:

$$p(x_i, y_j) = p(y_j|x_i)p(x_i)$$

Za entropijo:

$$\begin{aligned} H(Y|X) &= \sum_i p(x_i)H(Y|x_i) \\ &= -\sum_{i=1}^r \sum_{j=1}^s p(x_i, y_j) \log p(y_j|x_i) \end{aligned}$$

Splosno velja: 0 ≤ *H*(*Y*|*X*) ≤ *H*(*Y*), ce pozna-mo spremenljivko *X*, se nedolocenost *Y* ne more povecati (lahko se pomanjša).

#### 3.2.1 Pogojna verjetnost

Verjetnost da se zgodi dogodek A, ce vemo, da se zgodi dogodek B, je

$$P(A|B) = \frac{P(A \cap B)}{P(B)} = \frac{P(A)P(B|A)}{P(B)}$$

Dogodka *A* in *b* sta **neodvisna**, ce velja *P*(*A*|*B*) = *P*(*A*) ali *P*(*A*|*B*) = *P*(*A*)*P*(*B*). Pazi! Za par **nezdružljivih** dogodkov *A* in *B* pa velja *P*(*AB*) = 0, *P*(*A* + *B*) = *P*(*A*) + *P*(*B*), *P*(*A*|*B*) = 0 in *P*(*B*|*A*) = 0.

#### 3.2.2 Popolna verjetnost

Dogodki *H<sub>1</sub>*, *H<sub>2</sub>*, . . . *H<sub>n</sub>* tvorijo **popoln sistem dogodkov**,

$$\sum_{i=1}^{\infty} P(A \cap H_i) = \sum_{i=1}^{\infty} P(H_1)P(A|H_i)$$

### 3.3 Vezana entropija spremenljivk

Vezana entropija naključnih spremenljivk *X* in *Y* je entropija para (*X*, *Y*). Pomembne veze:

- $p(x_i, y_j) = p(y_j|x_i)p(x_i),$
  - $\sum_j p(x_i, y_j) = p(x_i),$
  - $\sum_i p(x_i, y_j) = p(y_j),$
  - $\sum_{i,j} p(x_i, y_j) = 1$  (pazi pri racunskih!)
- Velja: *H*(*X*, *Y*) = *H*(*Y*|*X*) + *H*(*X*).

#### 3.3.1 Obrat kanala

Ker velja tudi *H*(*X*, *Y*) = *H*(*X*|*Y*) + *H*(*Y*), kanal lahko **obrnemo Pogoji**: poznati moramo vhodne verjetnosti. Iz njih lahko določimo izhodne verjetnosti, ki jih lahko uporabimo kot vhodne verjetnosti v obrnjeni kanal. Lastnosti:

- izracun izhodnih verjetnosti  $p(y_j) = \sum_i p(y_j, x_i)p(x_i)$
- obratne pogojne vrjetnosti  $p(x_i, y_j) = p(y_j|x_i)p(x_i) = p(x_i|y_j)p(y_j)$

### 3.4 Medesebojna informacija

Pove nam, koliko o eni spremenljivki izvemo iz druge spremenljivke,

- $I(X;Y) = H(X,Y) - H(X|Y) - H(Y|X)$
- $I(X;Y) = H(X) - H(X|Y)$
- $I(X;Y) = H(Y) - H(Y|X)$
- $I(X;Y) = H(X) + H(Y) - H(X,Y)$
- I*(*X*; *Y*) = simetrična glede na *X* in *Y*
- $I(X;Y) \geq 0$
- $I(X;X) = H(X)$

### 3.5 Kapaciteta kanala

$$C = \max_{P(X)} I(X;Y)$$

#### 3.5.1 Kapaciteta kanala BSK

Lastnosti:

- $C = \max_{P(X)} (H(Y) - H(Y|X))$
- $p(x_0) = \alpha, p(x_1) = 1 - \alpha$
- $I(X;Y) = H(Y) - H(Y|X) = \cdots = H(Y) - H(p, 1 - p)$
- $\frac{dI(X;Y)}{d\alpha} = 0$
- $H(Y) = 1 \Rightarrow C$  je max
- $C = I(X;Y)|_{\alpha=1/2} = 1 - H(p, 1 - p)$

#### 3.5.2 Kapacitata kanala BSK z brisanjem

Definicija:

$$P_k = \begin{pmatrix} 1-p & p & 0 \\ 0 & p & 1-p \end{pmatrix}$$

Lastnosti:

- $C = 1 - p$
- $p(x_0) = \alpha, p(x_1) = 1 - \alpha$
- $p(y_0) = (1 - p)\alpha, p(y_1) = p, p(y_2) = (1 - p)(1 - \alpha)$
- $I(X;Y) = (1 - p)H(\alpha, 1 - \alpha)$
- $\frac{dI(X;Y)}{d\alpha} = 0 \Rightarrow \alpha = 1/2$

### 3.6 Shannonov drugi teorem

Shannon je ugotovil, da nam združevanje znakov v nize daje več možnosti za doseganje zanesli-jvega prenosa.

Naj bo *M* število različnih kodnih zamenjav, ki jih lahko oblikujemo z nizi dolžine *n*. Potem je **hitrost koda** (prenosa) definirana kot:

$$R = \frac{\max H(X^n)}{n} = \frac{\log M}{n} = \frac{k}{n}$$

Hitrost je največja takrat, ko so dovoljene kodne zamenjave na vходу enako verjetne. **Teorem**:

Za **R ≤ C** obstaja kod, ki zagotavlja tako preverjanje informacije, da je verjetnost napake pri dekodiran poljubno majhna. Za **R > C** kod, ki bi omogočal preverjanje informacije s poljubno majhno verjetnostjo napake, **ne** obstaja.

Ce so znaki neodvisni, velja:

$$\log(H(X^n)) = n \log H(X) \Rightarrow R = H$$

Za  $R \leq \frac{\log 2^n C}{n} = C$  je možno najti kodne zamenjave, ki omogočajo zanesljivo komunikacijo.

## 4 Varno kodiranje

### 4.1 Hammingova razdalja

Razdalja med različnimi kodi mora biti vsaj 1, drugace je kod **singularen**. Razdalja je po-dana kot **minimalna** Hammingova razdalja med dvema kodnima zamenjavama. Število napak, ki jih kod zazna:

$$d \geq e + 1 \rightarrow e_{max} = d - 1$$

$$d \geq 2f + 1 \rightarrow f_{max} = \lfloor \frac{d-1}{2} \rfloor$$

#### 4.1.1 Hammingov pogoj

Ce zelimo zagotoviti odpornost na napake, mora biti razdalja *d* > 1. Uporabni kodi imajo st. kodnih zamenjav  $M = 2^k < 2^n$ . Da bi lahko dekodirali vse kodne zamenjave, pri katerih je prislo do *e* ali manj napak mora veljati:

$$M \leq \frac{2^n}{\sum_{i=0}^e \binom{n}{i}}$$

### 4.2 Linearni bločni kodi

Kode označimo kot dvoječek *L*(*n*, *k*). O linearnih bločnih kodih govorimo, kadar:

- je vsota vsakega para kodnih zamenjav spet kodna zamenjava.
- da produkt kodne zamenjave z 1 in 0 spet kodno zamenjavo.
- vedno obstaja kodna zamenjava s samimi ničlami

**Hammingova razdalja** linearnega koda je enaka številu enic v kodni zamenjavi z najmanj enicami.

#### 4.2.1 Generatorska matrika

Generiranje kodne zamenjave lahko opisemo z generatorsko matriko.

$$\tilde{x} = \tilde{E}G$$

V splošnem podatkovni vektor 1 × *k* množimo z generatorsko matriko *k* × *n*, da dobimo kodno zamenjavo 1 × *n*. Kod, cigar generatorska ma-trika ima to obliko, je **sistematični kod** - prvih *k* znakov koda je enakih sporočilu (podatkovnim bitom), ostalih *n* − *k* znakov pa so paritetni biti. Za diskretne kanale brez spomina jo vedno lahko zapisemo v obliki *G* = (*I<sub>k</sub>* | *A*).

#### 4.2.2 Matrika za preverjanje sodosti

Linearne enacbe lahko zapisemo z matriko za preverjanje sodosti Lastnosti:

- $\tilde{x}H^T = 0$
- $G\tilde{H}^T = 0$
- $G = (I_k | A) \Rightarrow H = (A^T | I_{n-k})$
- vsota dveh kodnih zamenjav je nova kodna zamenjava.

### 4.3 Sindrom v kanalu

Predpostavimo da se med posiljanjem v kanalu zgodi napaka:

$$z \rightarrow x = zG \rightarrow err \rightarrow y = x + e \rightarrow s = yH^T$$

Napako pri prenosu preprosto ugotavljamo tako, da pogledamo, ce je *s* = 0. Vendar to nam ne garantira da pri prenosu ni prislo do napake. Sindrom izracunamo na naslednji nacin(vektor velikosti 1 × *n* − *k*):

$$yH^T = (x + e)H^T = eH^T = s$$

Ker je verjetnost za napako obicajno *p* << 1, je niz *s* *t* napakami veliko verjetnejši od niza *s* *t* + 1 napakami.

#### 4.3.1 Standardna tabela

Imejmo ponavljalni kod (0|00) in (1|11). Ses-tavimo matrkli G in H.

Imamo 4 možne sindrome: (00), (01), (10), (11). Na izhodu lahko dobimo 2<sup>*n*</sup> = 8 različnih nizov.

Možne nize na izhodu in njihove sindrome obi-cajno razvrstimo v std. tabelo:

sindrom	popravljalnik	
00	000	111
01	001	110
10	010	101
11	100	011

V isti vrstici so nizi, ki dajo enak sindrom. V prvi vrstici so vedno kodne zamenjave, ki imajo sindrom 0. Skrajno levo je vedno niz, ki ima naj-manj enic, saj je najbolj vrjeten. Imenujemo ga popravljalnik. Ostale nize dobimo tako, da popravljalnik pristevamo k kodnim zamenjavam v prvi vrsti.

### 4.4 Hammingov kod

Hammingovi kodi so družina linearnih bločnih kodov, ki lahko popravijo eno napako. Najlazuje jih predstavimo z matriko za preverjanje so-dosti, v kateri so vsi stolpci nenicelni vektorji. *H*(2<sup>*m*</sup> − 1 = *n*, 2<sup>*m*</sup> − 1 − *m* = *k*).Stolpci v Hammingovem kodu so lahko poljubno razmetani. Pomembno je le to, da nastopajo **vs**a števila od 1 do 2<sup>*m*</sup> − 1.

Hammingov kod je lahko:

- leksikografski** - oznake stolpcev si sledijo po vrsti
- sistematični** - oznake stolpcev so pomesane

V Hammingovem kodu se za varnostne bite obicajno vzamejo tisti stolpci, ki imajo samo **eno** enico.

#### 4.4.1 Dekodiranje

Dekodiranje leksikografskega Hammingovega koda je preprosto:

- izracunamo sindrom *s* = *yH<sup>T</sup>*
- ce je *s* = 0, je *x*' = *y*
- ce *s* ≠ 0, decimalno število *S* pred-stavlja mesto napake.

Za kod, ki pa ni leksikografski pogledamo, na kateri indeks se slika izracunani sindrom.

## 4.5 Ciklični kodi C(n, k)

### 4.5.1 Zapis s polinomi

Imejmo osnovni vektor:

$$\begin{aligned}x &= (x_{n-1}, x_{n-2}, \dots, x_0) \Leftrightarrow \\x(p) &= x_{n-1}p^{n-1} + x_{n-2}p^{n-2} + \dots + x_0\end{aligned}$$

Izvedemo premik za eno mesto:

$$\begin{aligned}x' &= (x_{n-2}, \dots, x_0, x_{n-1}) \Leftrightarrow \\x'(p) &= x_{n-2}p^{n-2} + \dots + x_0p + x_{n-1}\end{aligned}$$

Velja zveza:  $x'(p) = px(p) - x_{n-1}(p^n - 1)$ .  
V mod 2 aritmetiki:

$$\Rightarrow x'(p) = px(p) + x_{n-1}(p^n - 1).$$

V mod( $p^n + 1$ ) aritmetiki:

$$\Rightarrow x'(p) = px(p) \bmod (p^n + 1).$$

**Pozor:** aritmetiko po mod 2 izvajamo na **is-tih** stopnjah polinoma (na bitih), aritmetiko po mod ( $p^n + 1$ ) pa na **polinomu**.  
Izvajanje kroznega prekmika za  $i$  mest:

$$x^i(p) = p^i x(p) \bmod (p^n + 1)$$

### 4.5.2 Generatorski polinomi

Vrstice generatorske matrike lahko razumemo kot kodne zamenjave. Za ciklične kode v splošnem velja: **Generatorski polinom** je stopnje  $m$ , kjer je  $m$  število varnostnih bitov, in ga označimo kot:

$$g(p) = p^m + g_{m-1}p^{m-1} + \dots + g_1p + 1$$

Za sistematični kod velja:  $G = [I_k | A_{k,n-k}]$ . Sistematični lahko dobimo z linearnimi operacijami nad vrsticami. Velja:

$$p^n + 1 = g(p)h(p)$$

Sepravi vsak polinom, ki polinom  $p^n + 1$  deli brez ostanka, je generatorski polinom. Kako narediti kod leksikografski in hkrati sistematični?  $H_L \rightarrow H_S \rightarrow G_S$ .

### 4.5.3 Polinom za preverjanje sodosti

Velja:  $x(p)h(p) \bmod (p^n + 1) = 0 \Rightarrow \sum_{i=0}^n x_i h_{j-i} = 0$   
V matrični obliki:  $\vec{x}H^T = H\vec{x}^T = 0$

### 4.5.4 Kodiranje z množenjem

Kodne zamenjave so večkratniki generatorskega polinoma. Velja:

$$x(p) = z(p)g(p) \bmod (p^n + 1)$$

, kjer je  $z(p)$  polinom, ki ustreza podatkovnemu vektorju  $\vec{z}$  Kod, ki smo ga dobili z množenjem, ustreza generatorski matriki, ki ima v vrsticah koeficiente  $p^{k-1}g(p), \dots, pg(p), g(p)$ , zato ni sistematičen.

### 4.5.5 kodiranje z deljenjem

Kodiranje na osnovi deljenja ustvari sistematičen ciklični kod. Kodna zamenjava je zato sestavljena iz podatkovnega in varnostnega bloka znakov,  $x = (z|r)$ . Polinom podatkovnega bloka je:

$$z(p) = z_{k-1}p^{n-1} + \dots + z_1p^1 + z_0p^0$$

Ce pa polinom pomnožimo s  $p^m$ , dobimo na desni  $m$  nicel.

$$p^m z(p)$$

To ustreza bloku  $z$ , premaknjenem za  $m$  znakov v levo,  $(z_{k-1}, \dots, z_0, 0, \dots, 0)$ .

V splošnem nastavek seveda ne bo deljiv, velja pa  $p^m z(p) = g(p)t(p) + r(p)$ , kjer je  $t(p)$  količnik,  $r(p)$  pa ostanek, s stopnjo manj od  $m$ .

Sepravi delimo  $(p^m * z(p))/g(p)$  in ostanek bodo nasi varnostni biti.  $(z_{k-1}, \dots, z_0|r_{m-1}, \dots, r_0)$ .

### 4.5.6 Strojna izvedba kodirnika

Uporabljeni so trije tipi elementov: pomnilna celica tipa  $D$ , sestevalnik (XOR), množenje s konstanto ( $1 \mid 0$ ). Poznamo kodiranje na osnovi deljenja in na osnovi množenja. (insert pics here). Pri kodiranju se sepravi najprej na izhod pošiljajo kar vhodni znaki, potem v naslednjih korakih se vsebina pomnilnih celic od zadaj naprej.

### 4.5.7 Dekodiranje

Dekodiranje cikličnih kodov sloni na linearnih blocnih kodih. Vzememo, da je pri prenosu prislo do napake  $y = x + e$ , ali pa zapisano v polinomski obliki  $y(p) = x(p) + e(p) = z(p)g(p) + e(p)$ .

- Najprej izračunamo sindrom. Ekvivalent enačbe  $s = yH^T$  v polinomskem zapisu je  $y(p) = q(p) * g(p) + s(p)$ , oz.  $s(p) = y(p) \bmod g(p)$ .

Ce je ostanek deljenja  $y(p) \not\equiv z(p)$  različen od nič, je prislo do napake.

Iz  $s(p) = y(p) \bmod g(p)$  sledi, da je v primeru, ko je napaka na zadnjih  $m$  mestih, stopnja  $e(p)$  manj kot  $m$  in velja kar  $e(p) = s(p)$ . Za ostale napake pa lahko izkoristimo cikličnost kodov:

- Naredimo trik, osnovno enačbo prekamnemo za  $i$  mest:

$$p^i y(p) = p^i x(p) + p^i e(p)$$

- Ce najdemo pravi  $i$ , bo veljalo  $p^i e(p) = s(p)$
- Pravi  $i$  je tisti, pri katerem bo  $e(p)$  imel najmanj enic

## 4.5.8 Klasifikacija napak

Napaki, ki se pojavj na izhodu odposlane kodne zamenjave neodvisno od morebitnih napak na sosednjih znakih, pravimo **posamnica** ali **neodvisna** napaka. Do posamičnih napak pride zaradi motenj, ki so krajše od časa pošiljanja enega znaka.

Povezanim napakam na več zaporednih znakih pravimo **izbruh**. Dolžina izbruha je število znakov med prvim in zadnjim napacno sprejetim znakom. Do izbruha pride, ce je trajanje motenj daljše od časa pošiljanja enega znaka.

Ciklični kodi so posebej primerni za **ugotavljanje izbruhov napak** .

### 4.5.9 Zmoznosti cikličnih kodov

Odkrivanje napak s cikličnimi kodi, kjer velja  $1 < \text{st}(g(p)) < n$ :

- Kod odkrije vsako posamnico napako:  $e(p) = p^i$
- Za določene generatorske polinome odkrije tudi dve posamnici napaki do dolžine bloka  $n = 2^m - 1$
- Odkrije poljubno število lihih napak, ce  $p + 1$  deli  $g(p)$
- Odkrije vsak izbruh napak do dolžine  $m$

- Odkrije vse razen  $2^{-(m-1)}$  izbruhov dožline  $m + 1$
- Odkrije tudi vse razen delez  $2^{-m}$  izbruhov daljših od  $m + 1$

Popravljanje napak s cikličnimi kodi, kjer velja  $1 < \text{st}(g(p)) < n$ :

- Izracun sindroma
- Ciklično prilaganje sindroma prenesemu blok  $y$ .
- Popravijo lahko do  $e = \lfloor \frac{d-1}{2} \rfloor$  posamičnih napak, kjer je  $d$  Hammingova razdalja koda.
- Popravijo lahko tudi izbruhe napak do dolžine  $e = \lfloor \frac{m}{2} \rfloor$

### 4.5.10 CRC

Ali Cyclic Redundancy Check, temelji na cikličnih kodih. Po standardu velja:

- Registri v **LSFR** so na začetku nastavljeni na **1**; osnovni CRC ne loci sporočil, ki imajo različno število vodilnih nicel. Ta sprememba, ki je ekvivalentna negiranjju prvih  $m$  bitov, to težavo odpravi.
- Na koncu sporočila dodamo  $m$  - bitov, odvisno od implementacije LSFR. Pri nasi se to ne dela!
- Operacija XOR** na fiksnem ostanku deljenja, običajno je to kar negacija vseh bitov.
- Vrstni red bitov v bajtu** - nekateri serijski protokoli najprej oddajo najmanj pomembne bite (najmanj pomembni bit ima najvisjo stopnjo polinoma).
- Vrsni red bajtov** - pomnilniska organizacija, odvisna od arhitekture (LE, BE).
- Notacija CRC polinomov - biti označujejo prisotnost faktorja. Večkrat se izpusta en izmed faktorjev  $p^m$  ali 1.

Ciklični kodi so odlični za detekcijo napak. Za popravljanje napak pa danes obstajajo boljši kodi.

### 4.5.11 Prepletanje

Motnje so mnogokrat v obliki izbruhov. V takih primerih pride na določenih kodnih zamenjavah do velikega števila napak, na drugih pa napak ni. S prepletanjem bitov se da napake porazdeliti med več kodnih zamenjav. Resitev:

- Kodne zamenjave v kodirnik vpisujemo vrstico po vrstici, oddaja pa jih stolpec po stolpec. Obratno je na strani dekodirnika.
- Naceloma je vzorec skoraj naključen. Matriko prepletanja poznata kodirnik in dekodirnik.
- Dodamo zakasnitev, izmenično signala potujejo gor/dol, ena veja je zakasnjena.

Dejanske resitve so bolj kompleksne: več vej, zakasnitve tudi do 20 vej.

### 4.5.12 Konvolucijski kodi

Primerni za popravljanje napak. Konvolucijske kode generiramo z linearnimi premikalnimi registri, ki so sestavljeni iz pomnilnih celic  $D$  in vrat XOR. Spadajo pod nelinearne kode.

## 5 Analiza signalov

Pri analizi signalov in sistemov je izjemno pomembna količina frekvenca.

### 5.1 Invariantnost sinusoid

Vzemimo zvezni signal, ki prehaja skozi linearni medij (sistem) kot je na primer električno vezje. V splošnem bo signal na izhodu drugačen od signala na vhodu(zvok, ki ga poslušamo pod vodo je bistveno bolj popačen od tistega, ki ga poslušamo na zraku)

Pomembno pri signalih pa je, da se vhodni signal v obliki sinusoid

$$x(t) = A \sin(2\pi \nu t + \theta)$$

popaci v izhodni signal z drugačno amplitudo in fazo  $\theta$ , vendar ohrani frekvenco  $\nu$ . Razlog, da se frekvenca ohrani je v tem, da linearne sisteme lahko zapisemo v obliki elementarnih operacij, kot so (množenje s konstanto, odvajanje, integracija, zakasnitev, vsota).

## 5.2 Fourierova transformacija

Vsako periodično funkcijo ( ce je dovolj lepa ), lahko zapisemo kot kombinacijo sinusoid. V kombinaciji z invariantnostjo sinusoid to pomeni, da lahko:

- vsako funkcijo razstavimo na sinusoid
- obravnavamo obnasanje vsake sinusoid v sistemu posebej
- na koncu združimo locene rezultate

Ta koncept se danes uporablja pri vsaki analizi signalov.

### 5.2.1 Fourierova vrsta

Funkcija je periodična s periodo  $T$ , ce velja:

$$x(t + T) = x(t), \forall t : -\infty < t < \infty$$

kjer je  $T$  najmanjša pozitivna vrednost s to lastnostjo.

Funkciji  $\sin(t)$  in  $\cos(t)$  sta periodični s periodo  $2\pi \Rightarrow$  Funkciji  $\sin(\frac{2\pi t}{T})$  in  $\cos(\frac{2\pi t}{T})$  sta potem periodični funkciji s periodo  $T$  in frekvenco  $\nu_0 = \frac{1}{T}$ .

Če merimo v sekundah, frekvenco pa v številu cikov na sekundo. Pri analizi signalov zapis večkrat poenostavimo tako, da namesto frekvence uporabimo kotno hitrost

$$\omega_0 = 2\pi \nu_0 = \frac{2\pi}{T}$$

Visji harmoniki sinusoid s frekvenco  $\nu_0$  so sin in cos funkcije s frekvencami, ki so večkratniki osnovne frekvence,  $n\nu_0$ .

Fourier je pokazal, da lahko **vsako** periodično funkcijo  $x(t)$  s periodo  $T$  zapisemo kot:

$$\begin{aligned}x(t) &= \\ \frac{a_0}{2} + \sum_{n=1}^{\infty} a_n \cos(n\omega_0 t) + \sum_{n=1}^{\infty} b_n \sin(n\omega_0 t)\end{aligned}$$

za  $n \geq 1$ .

To velja za vsako funkcijo, ki zadosca Dirichletovim pogojem:

- je enoznačna (za vsak  $t$  ena sama vrednost)
- je končna povsod, oz. njen integral je koncen
- je absolutno integrabilna (ima končno energijo)

$$\int_0^T |x(t)|dt < \infty$$

- mora imeti končno število ekstremov v vsakem območju

- imeti mora knčno število končnih nezveznosti v vsakem območju

Bolj kompaktna predstavitev je z uporabo **Eulerjeve formule**  $e^{i\phi} = \cos(\phi) + i \sin(\phi)$ ,  $i = \sqrt{-1}$ :

$$x(t) = \sum_{n=-\infty}^{\infty} c_n e^{in\omega_0 t}$$

Koeficienti so kompleksni:

$$\begin{aligned}c_n &= \frac{1}{T} \int_0^T x(t) e^{-in\omega_0 t} dt \\ &= \frac{\int_{-T/2}^{T/2} x(t) e^{-in\omega_0 t} dt}{T/2}\end{aligned}$$

Zveza med obema zapisoma:

- $n = 0 : c_0 = \frac{a_0}{2}$
- $n > 0 : c_n = \frac{a_n - ib_n}{2}$
- $n < 0 : c_n = \frac{a_{-n} - ib_{-n}}{2}$

Negativne frekvence so matematični konstrukt, ki nam pride prav pri opisovanju signalov. Vsako sinusoido opišemo z dvema parametroma, prej  $a_n, b_n$ , sedaj pa elegantno s  $c_n$  in  $c_{-n}$ .

### 5.2.2 Fourierova transformacija

Fourierovo vrsto lahko posplošimo tako, da spustimo  $T \rightarrow \infty$  in dobimo Fourierovo transformacijo. Predstavlja jedro vseh frekvenčnih analiz. Enačba:

$$\begin{aligned}x(t) &= \int_{-\infty}^{\infty} X(\nu) e^{-i2\pi \nu t} dt = \\ &= \int_{-\infty}^{\infty} x(t) e^{-i\omega t} dt\end{aligned}$$

Manjši kot je  $T$  v časovnem prostoru, sirsí je signal v frekvenčnem prostoru.

Lastnosti Fourierove transformacije:

- linearost:  $f(t) = ax(t) + by(t) \rightarrow F(\nu) = aX(\nu) + bY(\nu)$
- skaliranje:  $f(t) = x(at) \rightarrow F(\nu) = \frac{1}{|a|} X(\frac{1}{a} \nu)$
- premik:  $f(t) = x(t - t_0) \rightarrow F(\nu) = e^{-i2\pi \nu t_0} X(\nu)$
- modulacija:  $f(t) = e^{i2\pi \nu_0 t} x(t) \rightarrow F(\nu) = X(\nu - \nu_0)$
- konvolucija:  $f(t) = \int_{-\infty}^{\infty} x(t - \tau)y(\tau)d\tau \rightarrow F(\nu) = X(\nu)Y(\nu)$

### 5.2.3 Diskretna Fourierova transformacija - DFT

Frekvenca vzorčenja  $\nu_S$  (sampling) je obratno sorazmerna periodi vzorčenja  $\nu_S = \frac{1}{\Delta}$ . Postopek:

- Ocenimo Fourierovo transformacijo iz  $N$  zaporednih vzorcev.
- Ocenimo Fourierovo transformacijo iz  $N$  vzorcev na vhodu v DFT bomo lahko izračunali natanko  $N$  neodvisnih točk na izhodu.
- Namesto, da bi določili DFT za vse točke od  $-\nu_C$  do  $+\nu_C$ , se lahko omejimo samo na določene vrednosti

$$\nu_n = \frac{n}{N\Delta}, n = -\frac{N}{2}, \dots, \frac{N}{2}$$

spodnja in zgornja meja ustrezata ravno Nyquistovi frekvenci.

- Trenutni zapis vključuje  $N+1$  vrednost. Izkazalo se bo, da sta obe robni vrednosti enaki. Imamo jih zaradi lepšega zapisa.

- Naprej so stvari trivialne

$$\begin{aligned}X(\nu_n) &= \int_{-\infty}^{\infty} x(t) e^{-i2\pi \nu_n t} dt = \\ &= \sum_{k=0}^{N-1} x_k e^{-i2\pi \nu_n k \Delta}\end{aligned}$$

- Ce v zgornji enačbi izpustimo  $\Delta$ , dobimo enačbo za DFT:

$$X_n = \sum_{k=0}^{N-1} x_k e^{\frac{-i2\pi nk}{N}}$$

Povezava s Fourierovo transformacijo je  $X(\nu_n) \approx \Delta X_n$  Iz enačbe za DFT sledi, da je DFT periodična s periodo  $N$ . To pomeni, da je  $X_{-n} = X_{N-n}$  Koeficiente  $X_n$  lahko zato namesto na intervalu  $[-\frac{N}{2}, \frac{N}{2}]$  računamo na intervalu  $[0, N - 1]$ .

Zveza med koeficienti  $X_0, \dots, X_{N-1}$  in frekvencami  $-\nu_C, \dots, \nu_C$ :

indeks	frekvenca
$n = 0$	$\nu = 0$
$1 \leq n \leq \frac{N-1}{2}$	$0 < \nu < \nu_C$
$\frac{N}{2}$	$-\nu_C, +\nu_C$
$\frac{N}{2} + 1 \leq n \leq N - 1$	$\nu_C < \nu < 0$

### 5.2.4 Inverzna DFT

$$x_k = \frac{1}{N} \sum_{n=0}^{N-1} X_n e^{\frac{i2\pi nk}{N}}$$

## 5.3 Resonanca

Do resonance pride, ko je frekvenca vsiljenega nihanja enaka frekvenci lastnega nihanja. Takrat pride do ojačitve amplitud. Resonanca je pomembna lastnost električnih vezij, s katero zagotovimo nihanja, nastavljanje radijskih sprejemnikov na pravo postajo, odstranimo sum.

## 5.4 Modulacija in frekvenčni premik

Iz analize vemo, da nelinearne operacije nad signali (kvadriranje, množenje) privedejo do pomembnih transformacij v frekvenčnem prostoru.

Iz osnovne trigonometrije vemo:

$$\begin{aligned}\sin(2\pi \nu_1 t) \sin(2\pi \nu_2 t) &= \\ \frac{1}{2} [\cos(2\pi (\nu_1 - \nu_2)t) - \cos(2\pi (\nu_1 + \nu_2)t)] \\ \cos(2\pi \nu t) &= \sin(2\pi \nu t + \pi/2)\end{aligned}$$

Produkt sinusoid s frekvencama  $\nu_1$  in  $\nu_2$  lahko torej zapisemo kot vsoto sinusoid s frekvenco  $\nu_1 + \nu_2$  in sinusoid s frekvenco  $\nu_1 - \nu_2$ .

To lastnost izkorisca amplitudna modulacija (radijske postaje AM) in frekvenčni premik, s katerim lahko zagotovimo hkraten prenos več signalov po istem mediju.

## 5.5 Teorem vzorčenja

Signal moramo vzorčiti vsaj s frekvenco  $2\nu_C$ , ce je najvisja opazena frekvenca v signalu  $\nu_C$ . Na tem zaključku sloni vsa danasnja tehnologija.

### 5.5.1 Zajem signalov

Zvezni signal  $x(t)$  je funkcija zvezne spremenljivke  $t$ . Diskreten signal je definiran samo za določene case, ki si najpogosteje sledijo v enaki časovnih intervalih  $x_k = x(k\Delta)$ ,  $\Delta$  je periodo vzorčenja.

Signale danes običajno zajemamo z racuanl-niki. Za to se uporabljajo vezja  $A/D$  pretvorniki. Imajo končno natančnost, na primer 12bit. Signal torej opišemo s končno mnogo različnimi amplitudami  $2^{12}$ .

Diskretnemu in kvantiziranemu signalu recemo tudi digitalni signal. Kvantizacija je običajno tako fina, da jo lahko zanemarimo.

## 5.6 Energija signala

Definicija:

$$E = \int_{-\infty}^{\infty} x(t)^2 dt$$

### Parsevalov teorem

$$\int_{-\infty}^{\infty} x(t)^2 dt = \int_{-\infty}^{\infty} |X(\nu)|^2 d\nu$$

Porazdelitev energije po frekvencah podaja funkcija  $|X(\nu)|^2$ , ki jo imenujemo **energijska spektralna gostota**.

### 5.6.1 Mocnostni spekter diskretnega kanala

Diskretna različica Parsevalovega teorema:

$$\sum_{k=1}^{N-1} |x_k|^2 = \frac{1}{N} \sum_{n=0}^{N-1} |X_n|^2$$

Pri diskretni razlici je PSD vedno v intervalu  $[-\nu_C, \nu_C]$ . Mocnostni spekter je potem:

$$P(0) = \frac{1}{N^2} |X_0|^2$$

$$P(\nu_n) = \frac{1}{N^2} [|X_n|^2 + |X_{N-n}|^2], n = 1, 2, \dots, \frac{N-1}{2}$$

$$P(\nu_C) = \frac{1}{N^2} |X_{\frac{N}{2}}|^2$$