1 Osnove

1.1 Ponovitev logaritmov

- $log_a x = \frac{log_b x}{log_b a}$
- $log_b(\frac{x}{y}) = log_b x - log_b y$
- $x = b^y \implies log_b x = y$
- $log_2 x = logx$
- $0log0 = 0$

1.2 Bayesova formula

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

1.3 Lastna informacija

Opisuje dogodek, ki se je zgodil:

$$I_i = \log(\tfrac{1}{p_i}) = -\log(p_i)$$

1.4 Entropija

je povprecje vseh lastnih informacij:

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

Lastnosti: je zvezna, simetricna funckija (vrsni red p_i ni pomemben, sestevanje je komutativno). Je vedno vecja od 0 ($p_i \geq 0 \rightarrow -p_i \log p_i \geq 0 \rightarrow H(X) \geq 0$) in navzgor omejena z $\log n$.
Ce sta dogodka **neodvisna** velja aditivnost: $H(X, Y) = H(X) + H(Y)$.
Vec zaporednih dogodkov neodvisnega vira: $X^l = X \times \cdots \times X \rightarrow H(X^l) = lH(X)$.

2 Kodi

2.1 Uvod

Kod sestavljajo *kodne zamenjave*, ki so sestavljene iz znakov **kodne abecede**. Stevilo znakov v kodni abecedi oznacujemo z **r**.
Ce so $\{p_1, \ldots, p_n\}$ verjetnosti znakov $\{s_1, \ldots, s_n\}$ osnovnega sporocila in $\{l_1, \ldots, l_n\}$ dolzine prejetih kodnih zmanjav, je povprecna dolzina kodne zamenjave

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

2.2 Tipi kodov

- **optimalen** - ce ima najmanjso mozno dolzino kodnih zamenjav
- **idealen** - ce je povprecna dolzina kodnih zamenjav enaka entropiji
- **enakomeren** - ce je dolzina vseh kodnih zamenjav enaka
- **enoznacen** - ce lahko poljuben niz znakov dekodiramo na en sam nacin
- **trenuten** - ce lahko osnovni znak dekodiramo takoj, ko sprejmemo celotno kodno zamenjavo

2.3 Kraftova neenakost

Za dolzine kodnih zamenjav $\{l_1, \ldots, l_n\}$ in r znaki kodne abecede obstaja trenutni kod, iff

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

2.4 Povp. dolzina, ucinkovitost

Najkrajse kodne zamenjave imamo, ce velja:

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

Ucinkovitost koda:

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

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

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

kjer je $H_r(X)$:

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

2.5 Shannonov prvi teorem

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

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

pri cemer je $H(X)$ entropija vira X.
Postopek kodiranja po Shannonu:

1. znake razvrstimo po padajocih verjetnostih
2. dolocimo stevilo znakov v vsaki kodni zamenjavi (l_k)
3. za vse simbole izracunamo komulativne verjetnosti ($P_k = \sum_{i=1}^{k-1} p_i$)
4. P_k pretvorimo v bazo r. Kodno zamenjavo predstavlja prvih l_k znakov necelega dela stevila

2.6 Fanojev kod

Postopek kodiranja:

1. znake razvrstimo po padajocih verjetnostih
2. znake razdelimo v r cim bolj enako verjetnih skupin
3. Vsaki skupini priredimo enega od r znakov kodne abecede
4. Deljenje ponovimo na vsaki od skupin. Postopek ponavljamo, dokler je mogoce

2.7 Huffmanov kod

Huffmanov postopek kodiranja poteka od spodaj navzgor (Pri Fanoju je ravno obratno). Pri huffmanovem kodu imamo dve fazi:

1. Zdruzevanje
   (a) Posici r najmanj verjetnih znakov in jih zdruzi v sestavljeni znak, katerega verjetnost je vsota verjetnosti vseh znakov
   (b) Preostale znake skupaj z novo sestavljenim znakom spet razvrsti
   (c) Postopek ponavljaj dokler ne ostane samo r znakov
2. Razdruzevanje
   (a) Vsakemu od preostalih znakov priredi po en znak kodirne abecede
   (b) Vsak sestavljeni znak razstavi in mu priredi po en znak kodirne abecede
   (c) Ko zmanjka sestavljenih znakov, je postopek zakljucen

Pred kodiranjem, je vedno pametno preveriti, ce imamo zadostno stevilo znakov. Veljati mora:

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

Ce imamo premalo znakov, jih po potrebi dodamo s verjetnostjo $p = 0$.
Huffmanov kod lahko razsirimo tako, da vec osnovnih znakov zdruzujemo v sestavljene znake → bolj ucinkoviti kodi. Vendar naletimo na nevarnost kombinacijske eksplozije.

2.8 Aritmeticni kod

Je **hiter** in **blizu optimalnemu** kodu, ter manj ucinkovit kot Huffmanov, vendar se izogne kombinacijski eksploziji. Vsak niz je predstavljen kot realno stevilo $0 \leq R < 1$, kar nam pove, da daljsi kot bo niz, bolj natancno mora biti podano naravno stevilo R.
Postopek kodiranja(znakov ni potrebno razvrstiti):

1. Zacnemo z intervalom $[0, 1)$
2. Izbrani interval razdelimo na n podintervalov, ki se ne prekrivajo. Sirine podintervalov ustrezajo verjetnostim znakov. Vsak podinterval predstavlja en znak
3. Izberemo podinterval, ki ustreza iskanemu znaku
4. Ce niz se ni koncan, izbrani podinterval ponovno razdelimo (bne 2.tocka)

5. Niz lahko predstavimo s poljubnim realnim stevilom v zadnjem podintervalu

Ko dobimo realni interval, ga samo se pretvorimo v binarnega s pomocjo klasicnega pretvarjanja iz dec v bin stevilski sistem.

2.9 Kod Lempel-Ziv (LZ77)

Stiskanje temelji na osnovi slovarja, tako, da ne potrebujemo racunati verjetnosti za posamezne znake. **Kodirnik** med branjem niza gradi slovar, in **dekodirnik** med branjem kodnih zaamenjav rekonstruira slovar in znake.
Kodiranje: uporablja drseca okna, znaki se premikajo iz desne na levo. Referenca je podana kot trojcek:

- odmik - razdalja do zacetka enakega podniza v medpomnilniku
- dolzina enakega podniza
- naslednji znak

npr. (0, 0, A) - ni ujemanja, (4, 3, B) - 4 znake nazaj se ponovi 3 znakovni podniz, ki se nato zakljuci s B.
dekodiranje: sledimo kodnim zamenjavam

2.10 Deflate

Gre za predelan LZ77. Uporablja pare (odmik, dolzina). Ce ujemanja v kodni tabeli ni, zapise kar znak. Uporablja dve kodni tabeli:

- **tabela za znake in dolzine** - 285 simbolov (0-255 za osnovne znake, 256 konec bloka, 257-285 kodira dolzine) Kodne zamenjave brez dodatnih bitov, se zakodira s Huffmanom.
- **tabela odmikov**

Niz znakov se razdeli na bloke(64k) vsak blok se kodira na enega od treh nacinov:

1. **brez stiskanja** osnovni znaki se prepisejo
2. **stiskanje s staticnim Huffmanom** (verjetnosti podane vnaprej), Huffmanovo drevo ni zakodirano v bloku
3. **stiskanje s Huffmanom** izracunamo verjetnosti za vsak blok

Glava posameznega bloka: 1bit - zadnji/ni zadnji blok + 2bita tip stiskanja + pri (3) se Huffmanovo drevo Ker Huffmanovo drevo ni enolicno, uvedemo kanonicni Huffmanov kod.
Postopek:

1. znake razvrstimo najprej po dolzinah kodnih zamenjav in nato po abecedi
2. prvi simbol ima same nicle
3. vsakemu naslednjemu znaku dodelimo naslednjo binarno kodo (prejsnji + 1)
4. ce je kodna zamenjava daljsa od binarne kode stevila, na koncu pripnemo niclo
5. ponavjlaj (3) do konca

Na taksen nacin dosezemo, da je potrebno kodirati samo dolzine kodnih zamenjav.

2.11 Kod Lempel-Ziv (LZW)

Osnovni slovar je podan in ga sporti doponjujemo. Alogritem 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 doseze optimalno stiskanje, pribliza se entropiji.

2.12 Verizno kodiranje ali RLE (run lenght encoding)

Namesto originalnih podatkov, sharnjujemo dolzino verige (fffeef → 3f2e1f). Problemu, ko se podatki ne ponavljajo, se izognemo tako, da izvedemo kombinacijo direktnega kodiranja in kodiranja RLE.

2.13 Stiskanje z izgubami

S taksnim nacinom stiskanja lahko dosezemo veliko boljsa kompresijska razmerja, vendar izgubimo podatke. Zato ga uporabljamo samo s formati, kjer se ne ukvarjamo z integriteto podatkov(MP3, MPEG, JPEG, . . .). Postopki kodiranj znanih formatov:

- **JPEG**
  1. priprava slike → ker je svetlost bol pomembna, je barvna resolucija obicajno zmanjsana (YC_RC_B)
  2. aproksimacija vsake od treh komponent s 2D DCT
  3. kvantizacija → podatki ki bolj izstopajo so shranjeni manj natancno kot tisti ki so staticni
  4. kodiranje blokov s pomocjo entropije
  5. RLE cik-cak po sliki
  6. RLE kodiramo z Huffmanom ali Aritmenticnim kodom
- **MP3**
  1. Modified DCT
  2. odstranitev za cloveka neslisnih frekvenc
  3. stereo, ce sta si L in R pretvorimo v mono
  4. Huffman na koncu
- **MPEG**
  1. uvodno kodiranje → celotna slika JPEG
  2. nato pa kodiramo samo spremembe, ki so se zgodile v sliki JPEG s pomocjo vektorja premika. V primeru, da je prevec razlik, se ponovno kodira JPEG slika.

2.14 Kompresijsko razmerje

Izracunamo ga po formuli → stisnjeni binarni zapis C(M) / binarni zapis dokumenta (M):

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

3 Kanali

3.1 Uvod

Kanali so strukture, ki opisujejo medsebojno povezanost. Kanal prenasa informacijo o spemenljivki X do spremenljivke Y. Matematicno ga opisemo s **pogojnimi verjetnostmi**, ki povezujejo izhodne verjetnosti z vhodom.

3.2 Diskretni kanal brez spomina

Povezuje diskretni nakljuicni spremenljivki, s koncno mnozico stanj $X = \{x_1, \ldots, x_r\}$ in $Y = \{y_1, \ldots, y_s\}$. Obema nakljicnima spremenljivkama pripadajo tudi posamezne verjetnosti $P_X = \{p(x_1), \ldots, p(x_r)\}$ in $P_Y = \{p(y_1), \ldots, p(y_s)\}$. Kjer velja, da je vsota posamezne mnozice verjetnosti enaka 1. Kanal je definiran kot mnozica **pogojnih verjetnosti**

$$p(y_j|x_i).$$

Pogojna verjetnost nam pove verjetnost za dogodek y_j na izhodu iz kanala, ce je na vhodu v kanal dogodek x_i. Brez spomina je zato, ker so pogojne vrjetnosti konstantne in torej neodvisne od prehodnih simbolov, velja

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

Kanal popolnoma podamo z $r \times s$ pogojnimi verjetnostmi.

3.2.1 Binarni simetricni kanal (BSK)

Gre za poseben primer diskretnega kanala brez spomina. Napaka kanala je p, saj se z verjetnostjo p znak prenese v napacnega.

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

3.3 Pogojna entropija

Pogojna entropija spremenljivke Y pri znanem X se zapise kot $H(Y|X)$. Vzemimo, da se je zgodil dogodek $x_i \in X$. 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: $0 \leq H(Y|x_i)$.
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:

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

Splosno velja: $0 \leq H(Y|X) \leq H(Y)$, ce poznamo spremenljivko X, se nedolocenost Y ne more povecati (lahko se pomanjsa).

3.4 Vezana entropija spremeljivk

Vezana entropija nakljucnih spremenljivk X in Y je entropija para (X, Y). Pomembne zveze:

- $p(x_i, y_j) = p(y_j|x_i)p(x_i)$,
- $\sum_j p(x_i, y_j) = p(x_i)$,
- $\sum_i p(x_i, y_j) = p(y_j)$,
- $\sum_{i,j} p(x_i, y_j) = 1$

Velja: $H(X,Y) = H(Y|X) + H(X)$, kar nam pove, da ce najprej izvemo, kaj se je zgodilo v dogodku X in potem dobimo se dodatne informacije od dogodku Y, vemo vse.

3.4.1 Obrat kanala

Ker velja tudi $H(X,Y) = H(X|Y) + H(Y)$, kanal lahko **obrnemo** (sepravi vhod Y in izhod X). Pri tem ne obracamo fizicnega procesa, ampak samo verjetnostno strukturo, ki definira kanal. **Pogoj:** poznati moramo vhodne verjetnosti. Iz njih lahko dolocimo 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 $p(x_i, y_j) = p(y_j|x_i)p(x_i) = p(x_i|y_j)p(y_j)$

Za sprejemnika sporocila so obratne pogojne verjetnosti zelo pomembne, saj z njimi lahko iz prejetih znakov doloci verjetnost za vhodne znake.

3.5 Medesebojna informacija

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

$$I(X;Y) = H(X) - H(X|Y)$$

Lastnosti:

- $I(X;Y) = H(X,Y) - H(X|Y) - H(Y|X)$
- $I(X;Y) = H(X) - H(X|Y)$
- $I(X;Y) = H(Y) - H(Y|X)$
- $I(X;Y) = H(X) + H(Y) - H(X,Y)$
- $I(X;Y) =$ simetricna glede na X in Y
- $I(X;Y) = -\sum_i\sum_j p(x_i, y_j)\log\frac{p(x_i)p(y_j)}{p(x_i,y_j)}$
- $I(X;Y) \geq 0$
- $I(X;X) = H(X)$

3.6 Kapaciteta kanala

Kapaciteta kanala je najvecja mozna medsebojna informacija, ki jo lahko prenesemo od vhoda na izhod.

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

3.6.1 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
- $C = I(X;Y)|_{\alpha=1/2} = 1 - H(p, 1-p)$

3.6.2 Kapacitata kanala BSK z brisanjem

Definicija:

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

Lastnosti:

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

3.7 Shannonov drugi teorem

Shannon je ugotovil, da nam zdruzevanje znakov v nize daje vec moznosti za doseganje zanesljivega prenosa.
Naj bo M stevilo razlicnih kodnih zamenjav, ki jih lahko oblikujemo z nizi dolzine n. Potem je **hitrost koda** (prenosa) definirana kot:

$$R = \frac{maxH(X^n)}{n} = \frac{logM}{n}$$

Hitrost je najvecja takrat, ko so dovoljene kodne zamenjave na vhodu enako verjetne. Shannonov teorem pravi, da je mozna skoraj popolna komunikacija s hitrostjo, enako kapaciteti kanala. **Teorem:**

Za $\mathbf{R} \leq \mathbf{C}$ obstaja kod, ki zagotavlja tako preverjanje informacije, da je verjetnost napake pri dekodiran poljubno majhna. Za $\mathbf{R} > \mathbf{C}$ kod, ki bi omogocal 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 $R \leq \frac{\log 2^{nC}}{n} = C$ je mozno najti kodne zamenjave, ki omogocajo zanesljivo komunikacijo.

4 Varno kodiranje

4.1 Uvod

Omejili se bomo na enostavne linearne blocne kode za BSK. Dolzina bloka je k znakov, abeceda je enaka abecedi kanala, torej imamo $M = 2^k$ blokov $x_1, \ldots x_k$, $x_i \in \{0,1\}$. Za potrebe varovanja dodamo se nekaj varnostnih znakov, celotna dolzina vsake od M kodnih zamenjav je potem n. Namesto enega posljemo n enakih znakov. Boljsi pristop pa je, da naredimo kode, kjer se povecujeta n in k hitreje od razilke $n-k$.

4.2 Kontrolne vsote

Varnost komunikacije povecamo tako, da dodamo nekaj bitov za preverjanje parnosti(paritetni biti). Nastavljeni so tako, da je vsota bitov v aritmetiki po modulu 2 fiksna vrednost (0 ali 1).
Spomnimo se arsa

+/-/XOR	0	1
0	0	1
1	1	0

AND $\sim \times$.

npr. $00|0, 01|1, 10|1, 11|1$(detektiramo samo eno napako).

4.2.1 Pravokotni kodi

Zapisemo ga v obliki pravokotnika, gledamo sodost po vrsticah in po stolpcih.

1	0	1
0	1	1
0	1	0

4.2.2 Trikotni kodi

Vsota elementov v stolpcu in vrstici s paritetnim bitom vred mora biti soda. (ravno tako vsota paritetnih bitov)

$$\begin{matrix} 1 & 0 & \mathbf{1} \\ 0 & \mathbf{0} & \\ \mathbf{1} & & \end{matrix}$$

4.3 Hammingova razdalja

Hammingova razdalja med kodnima zamenjava nam pove stevilo znakov, na katerih se razlikujeta. Kodni zamenjavi sta enaki, ce je razdalja 0, razdalja med razlicnimi kodi mora biti vsaj 1, drugace je kod **singularen**. Razdalja je podana kot **minimalna** Hammingova razdalja med dvema kodnima zamenjavama. Stevilo napak, ki jih kod zazna:

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

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

4.3.1 Hammingov pogoj

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

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

4.4 Linearni blocni kodi

Kode oznacimo kot dvojcek (n, k). n predstavlja stevilo vseh bitov, k podatkovnih, $n - k$ pa st. paritetnih. O linearnih blocnih 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 niclami

Oznacimo jih z $L(n, k)$. **Hammingova razdalja** linearnega koda je enaka stevilu enic v kodni zamenjavi z najmanj enicami. Naj bodo podatkovni biti oznaceni kot z_1, z_2 in z_3, varnostni pa kot s_1, s_2 in s_3:

$$\begin{matrix} z_1 & z_2 & s_3 \\ z_3 & s_2 & \\ s_1 & & \end{matrix}$$

Potem vrednosti zlozimo v vektor, in opravimo kodno zamenjavo.

$$\vec{x} = (x_1, x_2, x_3, x_4, x_5, x_6) = (z_1, z_2, z_3, s_1, s_2, s_3)$$

Velja:

$$\begin{aligned} z_1 + z_2 + s_1 = 0 = x_1 + x_2 + x_4 \\ z_3 + s_2 + z_2 = 0 = x_2 + x_3 + x_5 \\ s_3 + s_3 + z_1 = 0 = x_1 + x_3 + x_6 \end{aligned}$$

4.4.1 Generatorska matrika

Generiranje kodne zamenjave lahko opisemo z generatorsko matriko.

$$\vec{x} = \vec{z}G = \begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 \end{bmatrix}$$

V splosnem podatkovni vektor $1 \times k$ mnozimo z generatorsko matriko $k \times n$, da dobimo kodno zamenjavo $1 \times n$. Matrika mora imeti linearno neodvisne vrstice. Kod, cigar generatorska matrika ima to obliko, je **sistematicni kod** - prvih k znakov koda je enakih sporocilu (podatkovnim bitom), ostalih $n - k$ znakov pa so paritetni biti.

Za diskretne kanale brez spomina jo vedno lahko zapisemo v obliki $G = (I_k|A)$.

4.4.2 Matrika za preverjanje sodosti

Linearne enacbe lahko zapisemo z matriko za preverjanje sodosti:

$$H = \begin{bmatrix} 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 1 \end{bmatrix}$$

Lastnosti:

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

4.5 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 $1 \times n - k$):

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

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

4.5.1 Standardna tabela

Imejmo ponavljalni kod (0|00) in (1|11). Sestavimo matrki G in H.

$$G = [1|11] \text{ in } H = \begin{bmatrix} 1| & 1 & 0 \\ 1| & 0 & 1 \end{bmatrix}$$

Imamo 4 mozne sindrome: (00), (01), (10), (11). Na izhodu lahko dobimo $2^n = 8$ razlicnih nizov.

Mozne nize na izhodu in njihove sindrome obicajno 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 najmanj enic, saj je najbolj verjeten. Imenujemo ga popravljalnik. Ostale nize dobimo tako, da popravljalnik pristevamo k kodnim zamenjavam v prvi vrsti. Popravljanje je sedaj enostavno: izracunamo sindrom, popravljalnik odstejemo(pristejemo) od prejetega niza.

4.6 Hammingov kod

Hammingovi kodi so druzina linearnih blocnih kodov, ki lahko popravijo eno napako. Najlazje jih predstavimo z matriko za preverjanje sodosti, v kateri so vsi stolpci nenicelni vektorji. Spadajo med **popolne kode** - sfere z radijem 1 okrog kodnih zamenjav ravno napolnijo prostor z 2^n tockami.

Kod z m varnostnimi biti ima kodne zamenjave dolzine $2^m - 1$. Oznaka koda je potem $H(2^m - 1, 2^m - 1 - m)$. Ce stolpce v matriki H interpretiramo kot stevila v binarni obliki, nam oznaka stolpca doloca polozaj napake. Stolpci v Hammingovem kodu so lahko poljubno razmetani. Pomembno je le to, da nastopajo **vsa** stevila od 1 do $2^m - 1$.

Hammingov kod je lahko:

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

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

4.6.1 Dekodiranje

Dekodiranje leksikografskega Hammingovega koda je preprosto:

1. izracunamo sindrom $s = yH^T$
2. ce je $s = 0$, je $x' = y$
3. ce $s \neq 0$, decimalno stevilo S predstavlja mesto napake.

Za kod, ki pa ni leksikografski potrebujemo tabelo povezav med indeksi sindromov in stolpci(sepravi pogledamo, na kateri indeks se slika izracunani sindrom).

4.7 Ciklicni kodi

Ciklicni kod $C(n,k)$ je linearni blocni kod, v katerem vsak krozni premik kodne zamenjave da drugo kodno zamenjavo. Zapisemo jih s polinomi po padajocih potencah (ravno tako jih sestevamo po mod 2).

4.7.1 Zapis s polinomi

Imejmo osnovni vektor:

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

Izvedemo premik za eno mesto:

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

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 **istih** stopnjah polinoma (na bitih), aritmetiko po mod $(p^n + 1)$ pa na **polinomu**.

Izvajanje kroznega prekmika za i mest:

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

4.7.2 Generatorski polinomi

Vrstice generatorske matrike lahko razumemo kot kodne zamenjave. Za ciklicne kode v splosnem velja: **Generatorski polinom** je stopnje m, kjer je m stevilo varnostnih bitov, in ga oznacimo kot:

$$g(p) = p^m + g_{m-1}p^{m-1} + \dots + g_1 p + 1$$

Za sistematicni kod velja: $G = [I_k | A_{k,n-k}]$. Generatorska matrika:

$$G = \begin{bmatrix} 1 & g_{m-1} & \dots & g_1 & 1 & 0 & \dots & 0 & 0 \\ 0 & 1 & g_{m-1} & \dots & g_1 & 1 & 0 & \dots & 0 \\ \vdots & & & & & & & & \vdots \\ 0 & 0 & \dots & 0 & 1 & g_{m-1} & \dots & g_1 & 1 \end{bmatrix}$$

Sistematicni 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.

4.7.3 Polinom za preverjanje sodosti

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

V matricni obliki: $\vec{x}H^T = H\vec{x}^T = 0$

$$\begin{bmatrix} h_0 & \dots & h_k & 0 & \dots & 0 & 0 \\ 0 & h_0 & \dots & h_k & 0 & \dots & 0 \\ \vdots & & & & & & \vdots \\ 0 & 0 & \dots & 0 & h_0 & \dots & h_k \end{bmatrix} \begin{bmatrix} x_{n-1} \\ \vdots \\ x_0 \end{bmatrix} = 0$$

4.7.4 Kodiranje z mnozenjem

Kodne zamenjave so veckratniki generatorskega polinoma. Velja:

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

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

4.7.5 kodiranje z deljenjem

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

$$z(p) = z_{k-1}p^{n-1} + \dots + z_1 p^1 + z_0 p^0$$

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

$$p^m z(p) = z_{k-1}p^{k-1} + \dots + z_1 p^{m+1} + z_0 p^m$$

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

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

Ostanek lahko zapisemo v obliki niza, kot $(0, \dots, 0, r_{m-1}, \dots, r_0)$.

Polinom $p^m z(p) + r(p) = g(p)t(p)$ je deljiv z $g(p)$ in je zato ustrazna kodna zamenjava. Kodno zamenjavo tako dobimo, ce ostanek deljenja z generatorskim polinomom pristejemo k osnovnemu nastavku, $(z_{k-1}, \dots, z_0 | r_{m-1}, \dots r_0)$.

4.7.6 Strojna izvedba kodirnika

Uporabljeni so trije tipi elementov: pomnilna celica tipa D, sestevalnik (XOR), mnozenje s konstanto (1 | 0). Poznamo kodrianje na osnovi deljenja in na osnovi mnozenja. (insert pics here). Pri kodiranju se sepravi najprej na izhod posiljajo kar vhodni znaki, potem v naslednjih korakih se vsebina pomnilnih celic od zadaj naprej.

4.7.7 Dekodiranje

Dekodiranje ciklicnih kodov sloni na linearnih blocnih kodih. Vzemimo, 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 izracunamo sindrom. Ekvivalent enacbe $s = yH^T$ v polinomskem zapisu je $y(p) = q(p) * g(p) + s(p)$, oz. $s(p) = y(p) \bmod g(p)$.
- Ce je ostanek deljenja $y(p)$ z $g(p)$ razlicen od nic, 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 ciklicnost kodov:

- Naredimo trik, osnovno enacbo premaknemo 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

4.7.8 Klasifikacija napak

Napaki, ki se pojavi na izhodu odposlane kodne zamenjave neodvisno od morebitnih napak na sosednjih znakih, pravimo **posamicna** ali **neodvisna** napaka. Do posamicnih napak pride zaradi motenj, ki so krajse od casa posiljanja enega znaka.

Povezanim napakam na vec zaporednih znakih pravimo **izbruh**. Dolzina izbruha je stevilo znakov med prvim in zadnjim napacno sprejetim znakom. Do izbruha pride, ce je trajanje motenj daljse od casa posiljanja enega znaka.

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

4.7.9 Zmoznosti ciklicnih kodov

Odkrivanje napak s ciklicnimi kodi, kjer velja $1 <$ st $(g(p)) < n$:

- Kod odkrije vsako posamicno napako: $e(p) = p^i$
- Za dolocene generatorske polinome odkrije tudi dve posamicni napaki do dolzine bloka $n = 2^m - 1$
- Odkrije poljubno stevilo lihih napak, ce $p + 1$ deli $g(p)$
- Odkrije vsak izbruh napak do dolzine m
- Odkrije vse razen $2^{-(m-1)}$ izbruhov dolzine $m + 1$
- Odkrije tudi vse razen delez 2^{-m} izbruhov daljsih od $m + 1$

Popravljanje napak s ciklicnimi kodi, kjer velja $1 <$ st $(g(p)) < n$:

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

4.7.10 CRC

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

- Registri v **LSFR** so na zacetku nastavljeni na **1**; osnovni CRC ne loci sporocil, ki imajo razlicno stevilo vodilnih nicel. Ta sprememba, ki je ekvivalentna negiranju prvih m bitov, to tezavo odpravi.
- Na koncu sporocila dodamo m - bitov, odvisno od implementacije LSFR. Pri nasi se to ne dela!
- **Operacija XOR** na fiksnem ostanku deljenja, obicajno 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 oznacujejo prisotnost faktorja. Veckrat se izpusca en izmed faktorjev p^m ali 1.

Ciklicni kodi so odlicni za detekcijo napak. Za popravljanje napak pa danes obstajajo boljsi kodi.

4.7.11 Prepletanje

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

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

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

4.7.12 Konvolucijski kodi

Primerni za popravljanje napak. Konvolucijske kode genriramo 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 kolicina frekvenca.

5.1 Invariantnost sinusoid

Vzemimo zvezni signal, ki prehaja skozi linearni medij (sistem) kot je na primer elektricno vezje.

V splosnem bo signal na izhodu drugacen od signala na vhody(zvok, ki ga poslusamo pod vodo je bistveno bolj popacen od tistega, ki ga poslusamo na zraku)

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

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

popaci v izhodni signal z drugacno amplitudo in fazo θ, vendar ohrani frekvenco ν. Razlog, da se frekvenca ohrani je v tem, da linearne sisteme lahko zapisemo v obliki elementarnih operacij, kot so (mnozenje s konstanto, odvajanje, integracija, zakasnitev, vsota).

5.2 Fourierova transformacija

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

- vsako funkcijo razstavimo na sinusoide
- obravnavamo obnasanje vsake sinusoide v sistemu posebej
- na koncu zdruzimo locene rezultate

Ta koncep se danes uporablja pri vsaki analizi signalov.

5.2.1 Fourierova vrsta

Funkcija je periodicna s periodo T, ce velja:

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

kjer je T najmanjsa pozitivna vrednost s to lastnostjo.

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

Cas merimo v sekundah, frekvenco pa v stevilu ciklov na sekundo. Pri analizi signalov zapis veckrat poenostavimo tako, da namesto frekvence uporabimo kotno hitrost

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

Visji harmoniki sinusoid s frekvenco ν_0 so sin in cos funkcije s frekvencami, ki so veckratniki osnovne frekvence, $n\nu_0$.

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

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

za $n \geq 1$.

To velja za vsako funkcijo, ki zadosca Dirichletovim pogojem:

- je enoznacna (za vsak t ena sama vrednost)
- je koncna povsod, oz. njen integral je koncen
- je absolutno integrabilna (ima koncno energijo)

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

- mora imeti koncno stevilo ekstremov v vsakem obmocju
- imeti mora kncno stevilo koncnih nezveznosti v vsakem obmocju

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:

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

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 matematicni konstrukt, ki nam pride prav pri opisovanju singalov. Vsako sinusoido opisemo 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 posplosimo tako, da spustimo $T \to \infty$ in dobimo Fourierovo transformacijo. Predstavlja jedro vseh frekvencnih analiz. Enacba:

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

Manjsi kot je T v casovnem prostoru, sirsi je signal v frekvencnem prostoru.

Lastnosti Fourierove transformacije:

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

5.2.3 Diskretna Fourierova transformacija - DFT

Frekvenca vzorcenja ν_s (sampling) je obratno sorazmerna periodi vzorcenja $\nu_s = \frac{1}{\Delta}$. Postopek:

- Ocenimo Fourierovo transformacijo iz N zaporednih vzorcev.

$$x_k = x(k\Delta), k = 0, 1, \dots, N-1$$

- Iz N vzorcev na vhodu v DFT bomo lahko izracunali natanko N neodvisnih tock na izhodu.

- Namesto, da bi dolocili DFT za vse tocke od $-\nu_C$ do $+\nu_C$, se lahko omejimo samo na dolocene vrednosti

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

  spodnja in zgornja meja ustrezata ravno Nyquistovi frekvenci.

- Trenuten zapis vkljucuje $N+1$ vrednost. Izkazalo se bo, da sta obe robni vrednosti enaki. Imamo jih zaradi lepsega zapisa.

- Naprej so stvari trivialne

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

- Ce v zgornji enacbi izpustimo Δ, dobimo enacbo 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 enacbe za DFT sledi, da je DFT periodicna s periodo N. To pomeni, da je $X_{-n} = X_{N-n}$ Koeficiente X_n lahko zato namesto na intervalu $[-\frac{N}{2}, \frac{N}{2}]$ racunamo na intervalu $[0, N-1]$.

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

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

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 ojacitve amplitud. Resonanca je pomembna lastnost elektricnih vezij, s katero zagotovimo nihanja, nastavljanje radijskih sprejemnikov na pravo postajo, odstranimo sum.

5.4 Modulacija in frekvencni premik

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

Iz osnovne trigonometrije vemo:

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

Produkt sinusoid s frekvencama ν_1 in ν_2 lahko torej zapisemo kot vsoto sinusoide s frekvenco $\nu_1 + \nu_2$ in sinusoide s frekvenco $\nu_1 - \nu_2$.

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

5.5 Teorem vzorcenja

Signal moramo vzorciti vsaj s frekvenco $2\nu_c$, ce je najvisja opazena frekvenca v signalu ν_c. Na tem zakljucku sloni vsa danasnja tehnologija.

5.5.1 Zajem signalov

Zvezni signal $x(t)$ je funkcija zvezne spremenljivke t. Diskreten signal je definiran samo za dolocene case, ki si najpogosteje sledijo v enakih casovnih intervalih $x_k = x(k\Delta)$, Δ je perioda vzorcenja.

Signale danes obicajno zajemamo z racuanlniki. Za to se uporabljajo vezja A/D pretvorniki. Imajo koncno natancnost, na primer 12bit. Signal torej opisemo s koncno mnogo razlicnimi amplitudami 2^{12}.

Diskretnemu in kvantiziranemu signalu recemo tudi digitalni signal. Kvantizacija je obicajno 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 funckija $|X(\nu)|^2$, ki jo imenujemo **energijska spektralna gostota**.

5.6.1 Mocnostni spekter diskretnega kanala

Diskretna razlicica Parsevalovega teorema:

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

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

- $P(0) = \frac{1}{N^2}|X_0|^2$
- $P(\nu_n) = \frac{1}{N^2}[|X_n|^2 + |X_{N-n}|^2]$, $n = 1, 2, \dots, \frac{n}{2-1}$
- $P(\nu_C) = \frac{1}{N^2}|X_{\frac{N}{2}}|^2$