Matematično-fizikalni praktikum 2025/26

5. naloga: Hitra Fourierova transformacija in korelacijske funkcije

Samo Krejan, 28231092

Diskretno Fourierovo transformacijo smo definirali kot

$$H_k = \sum_{n=0}^{N-1} h_n \exp(2\pi \mathrm{i} k n / N), \qquad k = -\tfrac{N}{2}, \ldots, \tfrac{N}{2},$$

oziroma

$$H_k = \sum_{n=0}^{N-1} W_N^{nk} h_n, \qquad W_N = \exp(2\pi \mathrm{i}/N).$$

Ta postopek ima očitno časovno zahtevnost N^2. Račun pa je mogoče izvesti tudi z bistveno manj operacijami. Osnovni premislek je razcep

$$H_k = H_k^{\mathrm{sod}} + W_N^k H_k^{\mathrm{lih}} ,$$

kjer smo transformiranko H izrazili s transformacijama njenih sodih in lihih členov, pri čemer je vsota vsake od transformacij zdaj dolžine N/2. Gornjo relacijo lahko uporabljamo rekurzivno: če je N enak potenci števila 2, lahko rekurzijo razdrobimo do nizov, ki imajo samo še en člen. Zanj je transformacija identiteta. Za obrat pri eni vrednosti frekvence (pri danem m) je potrebno na vsakem koraku rekurzije le eno množenje s potenco W, korakov pa je $\log_2 N$. Skupna časovna zahtevnost je torej le še $N \log_2 N$.

Da ne iščemo pripadnikov niza po vsej tabeli, si podatke preuredimo. Lahko je pokazati, da je v prvotni tabeli treba med seboj zamenjati podatke, katerih vrstna števila v binarnem zapisu so obrnjena: v novem redu jemljemo člene kar po vrsti. Tudi potenc W ne izražamo vedno znova s sinusi in kosinusi, pač pa jih računamo z rekurzijo. Tak ali podoben postopek je osnova vseh algoritmov hitre Fourierove transformacije (FFT).

Z neko transformacijo iz družine FFT bomo izračunali korelacijsko funkcijo dveh signalov. Korelacija periodičnih funkcij $g(t)$ in $h(t)$ s periodo T je definirana kot:

$$\phi_{gh}(\tau) = \frac{1}{T} \int_0^T g(t+\tau)\, h(t)\, \mathrm{d}t ,$$

oziroma diskretno

$$\phi_{gh}(n) = \frac{1}{N} \sum_{k=0}^{N-1} g_{k+n}\, h_k .$$

Računamo torej skalarni produkt funkcij, ki sta časovno premaknjeni za τ oziroma n. Če je za določeno vrednost premika ta funkcija višja kot v okolici, potem to pomeni, da sta si funkciji podobni, le da ju je treba premakniti, da se to vidi.

V primeru, da sta funkciji (signala), ki ju primerjamo, enaki, računamo njuno *avtokorelacijsko funkcijo*: ta je mera za to, ali signal ostaja s pretekanjem časa sam sebi podoben. Če je signal slabo koreliran (sam s sabo), korelacija $\phi_{hh}(n)$ relaksira h kvadratu povprečnega signala $\langle h \rangle^2$, kjer je

$$\langle h \rangle = \frac{1}{N} \sum_{k=0}^{N-1} h_k .$$

Iz lokalnih maksimov v avtokorelacijski funkciji sklepamo na periodičnosti, bodisi popolne ali približne. Pri periodičnih signalih je tudi avtokorelacijska funkcija striktno periodična, za stohastične procese pa je značilna eksponentna avtokorelacijska funkcija. še bolj nas zanima, kako *hitro* se korelacija izgublja: računamo rajši reskalirano obliko avtokorelacije

$$\widetilde{\phi}_{hh}(n) = \frac{\phi_{hh}(n) - \langle h \rangle^2}{\phi_{hh}(0) - \langle h \rangle^2} ,$$

kjer je imenovalec nekakšno merilo za varianco signala,

$$\sigma^2 = \phi_{hh}(0) - \langle h \rangle^2 = \frac{1}{N} \sum_{k=0}^{N-1} (h_k - \langle h \rangle)^2 .$$

Pri zgornjih enačbah moramo še "peš" poskrbeti za periodično zaključenost signala pri $n = N$, torej da je perioda enaka velikosti vzorca. Če tega ne moremo narediti, je bolj pravilna definicija avtokorelacije

$$\phi_{hh}(n) = \frac{1}{N-n} \sum_{k=0}^{N-n-1} h_{k+n}\, h_k .$$

Praktičen račun po zgornji formuli lahko postane za velike vzorce prezamuden. Avtokorelacijo rajši računamo s FFT (DFT) $\mathcal{F}$, saj je korelacija obratna Fourierova transformacija $\mathcal{F}^{-1}$ produkta Fourierovih transformacij $\mathcal{F}$, torej z $G = \mathcal{F}g$ in $H = \mathcal{F}h$ dobimo

$$\phi_{gh}(n) = \frac{1}{N-n} \mathcal{F}^{-1} \left[G \cdot (H)^* \right]$$

oziroma

$$\phi_{hh}(n) = \frac{1}{N-n} \mathcal{F}^{-1} \left[\, |H|^2 \, \right] .$$

Za račun s FTT signale dolžine N najprej prepišemo v dvakrat daljše, periodično zaključene podatkovne nize, $\widetilde{h}_n = h_n$, $\widetilde{h}_{n+N} = 0$ za $n = 0, \ldots, N-1$ in $\widetilde{h}_{n+2N} = \widetilde{h}_n$. Tedaj se avtokorelacija zapiše v obliki

$$\phi_{hh}(n) = \frac{1}{N-n} \sum_{k=0}^{2N-1} \widetilde{h}_{k+n}\, \widetilde{h}_k ,$$

kar lahko izračunamo s FFT.

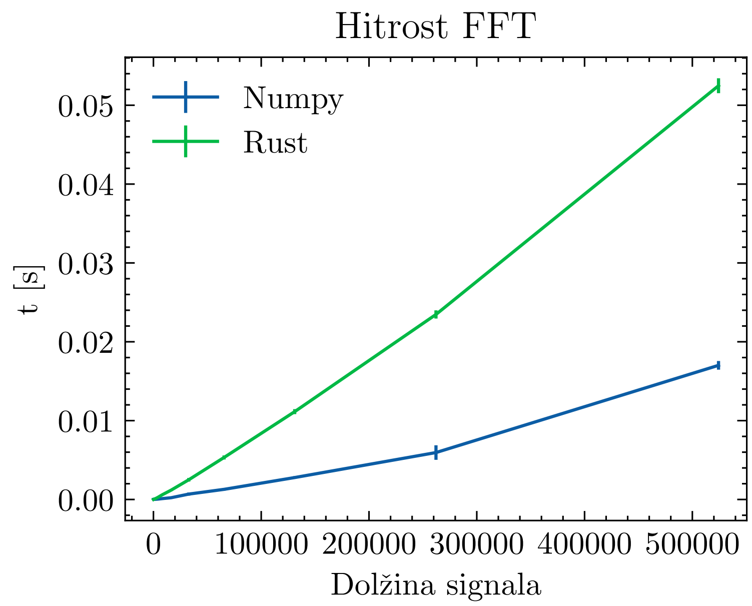
1 Naloga

Na spletni strani MF praktikuma najdeš posnetke oglašanja velike uharice, naše največje sove. Posneti sta dve sovi z minimalnim ozadjem (`bubomono` in `bubo2mono`) in nekaj mešanih signalov, ki zakrivajo njuno oglašanje (`mix`, `mix1`, `mix2` in `mix22`). V signalih `mix2` in `mix22` je oglašanje sove komaj še zaznavno. Izračunaj avtokorelacijsko funkcijo vseh signalov in poskusi ugotoviti, za katero sovo gre pri teh najbolj zašumljenih signalih!

Poglejte si rutine `four1` iz Numerical Recipes ali knjižnice `fftw3`, ki je še dosti hitrejša. V okolju Python so te rutine vključene v 'fft' paket. (Pri tako velikih vzorcih je skorajda nujno uporabiti FFT namesto počasne navadne DFT.)

1.1 FFT

Nalogo sem začel tako, da sem najprej naredil lastno implementacijo hitre Fourierove transformacijev Rust-u. Ta mi sicer ni uspela najbolje saj je med drugim delovala zgolj za sezname katerih dolžina je potenca števila 2 in pa, čeprav je dosegala natančnosti v bližini numerične zmogljivosti dvojne natančnosti, je bila mnogo počasnejša od $numpy.fft.fft$ implementacije (slika 1), tako da sem se v nadaljevanju naloge odločil uporabljati kar $numpy$ implementacijo.



Slika 1: Čas potreben za izvedbo FFT v odvisnosti od velikosti seznama (potence števila 2). Kvalitativno lahko vidimo, da časovna zahtevnost res ustreza $O(n \log n)$ in pa tudi, da je numpy implementacija mnogo boljša.

1.2 Avtokorelacijska funkcija

Ker bomo kasneje naredili avtokorelacijo signalov z hitrim vzorčenjem je pomembno, da poskrbimo, da je algoritem s katerim računamo avtokorelacijo čim bolj učinkovit. Zato sem analiziral tri implementacije avtokorelacije, sicer:

1. Osnovna implementacija v Pythonu po definiciji iz uvoda:

$$\phi_{hh}(n) = \frac{1}{N-n} \sum_{k=0}^{N-n-1} h_{k+n} \, h_k \, .$$

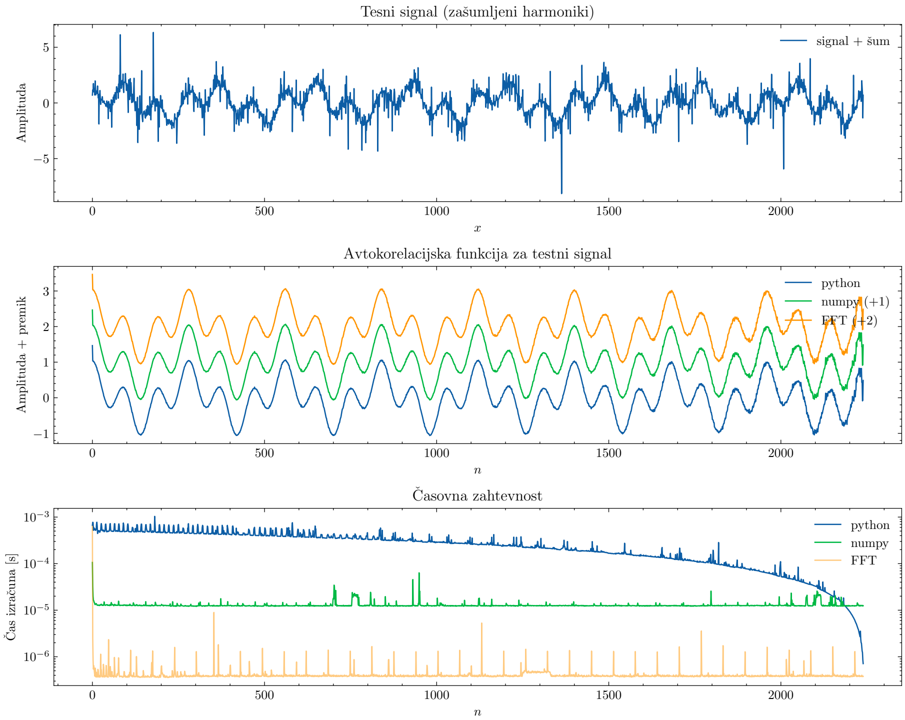
2. Po definiciji iz uvoda z žero paddingža uporabo *numpy*:

$$\phi_{hh}(n) = \frac{1}{N-n} \sum_{k=0}^{2N-1} \widetilde{h}_{k+n} \, \widetilde{h}_k \, ,$$

3. Navsezadnje pa še z uporabo FFT algoritma, kot je opisano v uvodu:

$$\phi_{hh}(n) = \frac{1}{N-n} \mathcal{F}^{-1} \left[\, |H|^2 \, \right] \, .$$

Uspešnost avtokorelacije zašumljenega signala harmonikov in potrebnim časom za vsako iteracijo, sem prikazal na grafu 2



Slika 2: Signal, njegova avtokorelacija (bralec naj bo pozoren na zamike za namen lepšega prikaza), ter časovna zahtevnost