Kolmogorov-Arnoldove mreže i generalizirane partonske distribucije

Mislav Širac
Mentor: Krešimir Kumerički
Fizički odsjek, Prirodoslovno-matematički fakultet, Bijenička 32, Zagreb

U ovom seminaru ćemo ispitati sposobnost prilagodbe funkcija Kolmogorov-Arnold neuronske mreže (KAN). Uvesti ćemo fizikalnu motivaciju iza prilagodbe generaliziranih partonskih distribucija i ukratko opisati Kolmogorov-Arnoldove mreže (KAN) i višeslojne perceptrone (MLP). Zatim ćemo provesti statističku analizu sposobnosti prilagodbe funkcija KAN mreže s različitim stupnjevima ljudske pomoći.

1 Uvod

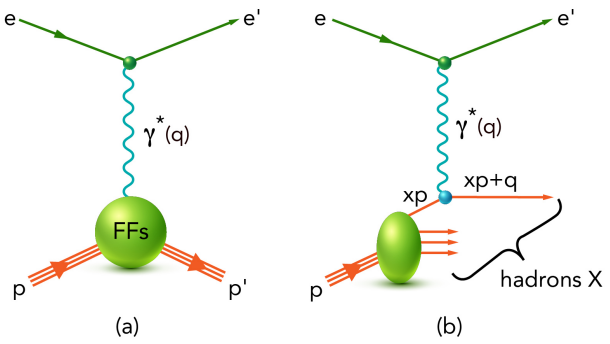
Od dana nastanka računala vidimo njihovu primjenu u znanosti. Računi na koje bi ljudi potrošili dane, tjedne ili više, računala naprave u samo par sekundi. Prvi primjer računa koji se koristio u znanstvene svrhe je bila simulacija detonacije nuklearne bombe na prvom digitalnom računalu ENIAC u sklopu Manhattan projekta. Upravo to računalo označava početak informacijskog doba. Danas preko platformi kao što je ChatGPT se susrećemo s novim probojem u svijetu tehnologije i znanosti, neuronskim mrežama. Neuronske mreže su inspirirane najkompleksnijom stvari na svijetu, ljudskim mozgom, no za razliku od mozga one se ne umaraju te dok god ima struje mogu raditi dan i noć, s uvijek istom učinkovitošću. U ovom seminaru ćemo probati pomoću nove vrste neuronske mreže, Kolmogorov-Arnold neuronske mreže (KAN), provesti račun iz fizike. Račun koji provodimo je prilagođavanje funkcija na eksperimentalno dobivene podatke za generaliziranu partonsku distribucijsku (GPD) funkciju. Dobivanje analitičkog oblika GPD funkcije bi nam dalo znatni uvid u unutarnju strukturu protona, radi čega i istražujemo KAN mrežu.

2 Generalizirana partonska distribucijska funkcija

Prvi uvid u unutarnju strukturu protona koju fizičar dobije je tijekom svog studija, kada obrađuje raspršenje elektrona na protonu. U slučaju elastičnog raspršenja, pomoću argumenata simetrije možemo vidjeti kako rezultat raspršenja ovisi o 2 nezavisna parametra koja mi najčešće biramo kao E_1 (energija snopa) i θ (kut raspršenja) [1]. Ostali parametri kao što su Q^2 (kvadrat 4-impulsa virtualnog fotona) i E_3 (energija raspršenog elektrona) se mogu izraziti pomoću njih. Pri poznatoj energiji ulaznog snopa možemo diferencijalni udarni presjek napisati kao:

$$\frac{d\sigma}{d\Omega} = \frac{\alpha^2}{4E_1^2 \sin^4(\frac{\theta}{2})} \frac{E_3}{E_1} \left(\frac{G_E^2 + \tau G_M^2}{1+\tau} \cos^2(\frac{\theta}{2}) + 2\tau G_M^2 \sin^2(\frac{\theta}{2})\right) [2] \quad (1)$$

tzv. Rosenbluthova formula, gdje G_E i G_M nazivamo form faktorima, no njih ne možemo interpretirati kao Fourierove transformate naboja ili magnetskog momenta kao što je to slučaj za form faktor iz Mott-ove formule. Elastično raspršenje nam može reći nešto o strukturi naboja i magnetskog momenta protona, no ne o sastavu istog. Radi toga se okrećemo prema duboko neelastičnom raspršenju (DIS). To je raspršenje kod kojeg se proton mijenja u pobuđeno



Slika 1: Dijagram elastičnog raspršenja elektrona na protonu iz kojeg saznajemo raspodjelu naboja protona (lijevo). Dijagram duboko neelastičnog raspršenja (desno). [3]

stanje samog sebe, ili razbija na nekoliko čestica. Pošto je proton veoma stabilan potrebne su velike energije za takve procese, zbog čega se ovakvi procesi zovu duboko neelastična raspršenja. Neelastično u biti znači da se mijenja ukupna kinetička energija sustava, te se stoga slama jedna simetrija radi čega je potreban još jedan parametar kako bi opisali proces. Parametar kojeg biramo je Bjorkenov $x = \frac{Q^2}{2p_2 q}$, gdje je p_2 4-impuls ulaznog protona, te q 4-impuls virtualnog fotona. S malo matematičke manipulacije se može vidjeti: $0 \leq x \leq 1$. Uvodeći neelastičnost kao $y = \frac{p_2 \cdot q}{p_2 \cdot p_1}$, za diferencijalni udarni presjek dobivamo:

$$\frac{d\sigma}{dQ^2 dx} = \frac{4\pi\alpha^2}{Q^4} [(1-y)\frac{F_2(x,Q^2)}{x} + y^2 F_1(x,Q^2)], \quad (2)$$

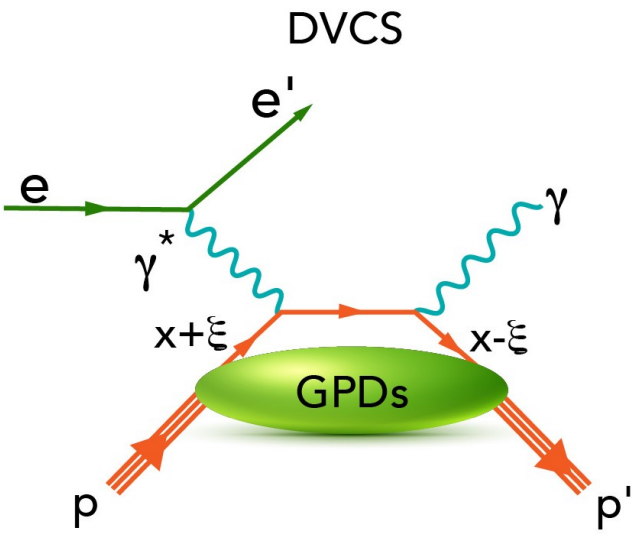
gdje sada naši form faktori F_1 i F_2 ovise o dvije varijable: x i Q^2. U prethodnoj formuli smo kut raspršenja zamijenili s varijablom Q^2. Ono što je primjećeno u DIS eksperimentima je da pri velikim vrijednostima Q^2 nestaje ovisnost F_1 i F_2 o Q^2. Takav prestanak ovisnosti sugerira elastično raspršenje elektrona na točkastim konstituentima protona, partonima tj. kvarkovima. Takav visoko energetski limit nazivamo Bjorkenovo skaliranje te x u režimu Bjorkenovog skaliranja predstavlja frakciju impulsa protona koju nosi parton na kojem se odvija raspršenje elektrona. Još jedna prednost DIS-a je što možemo impulse partona u protonu smatrati paralelnima i u smjeru impulsa protona, pošto je impuls samog protona toliko veći od mogućih transverzalnih impulsa partona, takve partone nazivamo kolinearnima. Pošto znamo diferencijalni udarni presjek za raspršenje elektrona na točkastoj čestici:

$$\frac{d\sigma}{dQ^2} = \frac{4\pi\alpha^2}{Q^4}[(1-y) + \frac{y^2}{2}], \tag{3}$$

pomoću njega možemo konstruirati diferencijalni udarni presjek u slučaju partona kao:

$$\frac{d\sigma}{dQ^2} = \frac{4\pi\alpha^2}{Q^4}\sum_i q_i(x)Q_i^2[(1-y) + \frac{y^2}{2}]dx, \tag{4}$$

gdje je Q_i naboj pojedinog partona, a $q_i(x)$ tzv. Partonska distribucijska funkcija (PDF) koja predstavlja vjerojatnost da nađemo određeni parton s tom frakcijom impulsa x. Uspoređujući formule (2) i (4) možemo vidjeti kako su naši form faktori u biti sume PDF-ova, $F_2(x,Q^2) = 2xF_1(x,Q^2) = x\sum_i Q_i^2 q_i(x)$. Osim DIS-a, za ispitivanje strukture protona možemo koristiti i duboko virtualno Comptonsko raspršenje (DVCS) [3] u kojem također uzimamo da su partoni u protonu kolinearni. DVCS u odnosu na DIS ima dodatnu izlaznu česticu u obliku realnog fotona što nam kompliciira kinematiku procesa i slama dodatne simetrije radi čega moramo uvesti nove parametre za opis DVCS-a. Zbog toga PDF zamjenjujemo sa generaliziranom partonskom distribucijskom funkcijom (GPD). DVCS dijagram se može vidjeti na slici 2. U slučaju DVCS procesa gornji dio dijagrama se može opisati pertrubativnim QCD-om dok se donji dio može opisati GPD-ovima, radi čega je taj proces dobar za ispitivanje strukture GPD funkcija. GPD također opisuje protonsku unutarnju strukturu, no za razliku od PDF-ova koji su funkcije samo x, GPD ovisi o 3 varijable: x, t i ξ. Mandelstamova varijabla t u slučaju DVCS predstavlja prijenos 4-impulsa između ulaznog i izlaznog protona, x je ponovo Bjorkenov faktor te ξ predstavlja longitudinalni prijenos impulsa na pogođeni parton. GPD-ovi u sebi uključuju i elektromagnetske form faktore i PDF-ove, u slučaju $t \to 0$ i $\xi \to 0$ GPD-ovi se pretvaraju u PDF-ove te pri integraciji GPD-ova po x dobivamo elektromagnetske form faktore. Samo parametri ξ i t su opservabilni u DVCS-u, x ulazi u DVCS amplitudu kao integracijski faktor. Stoga će form faktori koje dobijemo iz DVCS eskperimenata i koji nam govore nešto o strukturi GPD-ova biti funkcije 2 varijable, ξ i t. U seminaru se bavimo upravo fitanjem funkcija na eksperimentalne podatke za form faktore iz DVCS-a.



Slika 2: Dijagram duboko virtualnog Comptonskog raspršenja. Možemo vidjeti kako su u ulaznom kanalu proton i elektron, a u izlaznom realni foton, proton i elektron, također se vidi kako je gornji, pertrubativni QCD dio dijagrama, odvojen od GPD dijela omogućavajući izvlačenje podataka o GPD iz DVCS. [3]

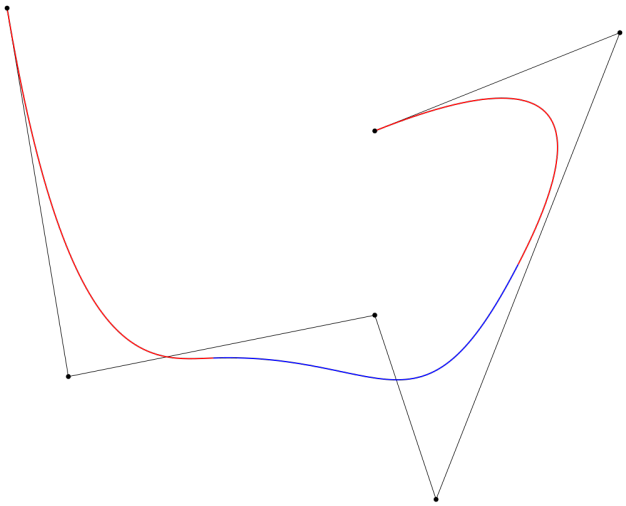
3 Kolmogorov-Arnoldova neuronska mreža

Prvo bi bilo dobro upoznati čitatelja s pojmom neuronskih mreža. Neuronska mreža je mreža vrhova (neurona) i bridova koji spajaju te vrhove. Struktura je takva da na ulazu imamo sloj neurona koji prima naše ulazne podatke, broj ulaznih neurona ovisi o broju ulaznih podataka, na izlazu naše mreže isto tako imamo onoliki broj neurona koliko izlaznih podataka očekujemo. Između ulaznog i izlaznog vrha se nalaze slojevi skrivenih neurona, svaki neuron u pojedinom sloju je povezan preko bridova sa svim neuronima iz prethodnog sloja i sa svim neuronima iz sljedećeg sloja. Način na koji neuronska mreža funkcionira je da uzima podatke iz početnog (ulaznog) sloja, te svaki neuron na svoj podatak primjenjuje neku aktivacijsku funkciju i prosljeđuje putem brida modificirani podatak svim neuronima u sljedećem sloju. Na bridovima postoje težine koje označavaju koliko je značajan za dobro funkcioniranje mreže podatak koji dolazi iz prethodnog neurona. U svakom neuronu se zbrajaju podatci iz svih neurona iz prethodnog sloja. Na početku treniranja klasičnih neuronskih mreža kao što je višeslojni perceptron (MLP) sve težine su nasumično zadane, a aktivacijske funkcije su fiksirane kroz cijeli tijek treninga. Mreža se zatim trenira tako

da prvo propusti podatke unaprijed s ulaznog kraja prema izlaznom te dobije neki izlazni podatak. Pomoću loss funkcije kao što je korijen srednje vrijednosti (RMSE) zatim izračuna koliko je odstupanje dobivenog rezultata od očekivanog. Ako od prve nije savršeno poklapanje očekivanog i dobivenog rezultata (što nikada nije) radi tzv. 'propagaciju unazad', gdje RMSE funkciju propagira unazad kroz mrežu i računa gradijent RMSE s obzirom na svaku težinu u mreži pomoću pravila lančane derivacije. Pomoću tog gradijenta zatim određuje koju težinu treba mijenjati u kojem smjeru i zatim to radi postepeno dok ne minimizira RMSE [4] [5]. U slučaju da je na svakom neuronu aktivacijska funkcija linearna, ovo bi bio samo primjer linearne regresije, no u slučaju MLP-ova aktivacijske funkcije su nelinearne i fiksne, stoga sva nelinearnost modela dolazi iz tih zadanih aktivacijskih funkcija [6]. MLP radi na bazi teorema o univerzalnoj aproksimaciji kojeg možemo ilustrirati na primjeru neuronske mreže s ulaznim slojem od n neurona, skrivenim slojem s m neurona, i izlaznim slojem s jednim neuronom. Teorem nam govori da su u našem primjeru konačne sume oblika $g(x) = \sum_{j=1}^{m} \sigma(\sum_{i=1}^{n} w_{j,i}^{(1)} - b_j)$ [7], gdje su $w_{j,i}^{(1)}$ težine između $j-tog$ neurona u ulaznom sloju i $i-tog$ neurona u skrivenom sloju, a $w_j^{(2)}$ težine između $j-tog$ neurona u skrivenom sloju i neurona u izlaznom sloju, guste na skupu kontinuiranih funkcija $f : I \to \mathbb{R}$, gdje je $I = [0, 1]$. Taj teorem nam u biti garantira da za svaki ϵ iz $|(f(x) - g(x))| < \epsilon$, gdje je f funkcija koju želimo dobiti, a g funkcija koju dobijemo pomoću MLP-a, možemo naći MLP, no on nam ne govori ništa o potrebnoj arhitekturi MLP-a [7] [8]. S druge strane KAN, mreža kojom se bavimo u ovom seminaru, je bazirana na Kolmogorov-Arnoldovom teoremu. Kolmogorov-Arnold teorem nam govori kako se kontinuirane funkcije više varijabli mogu rastaviti na sume konačnih kompozicija funkcija jedne varijable, specifično za glatke funkcije $f : [0, 1]^n \to \mathbb{R}$ vrijedi

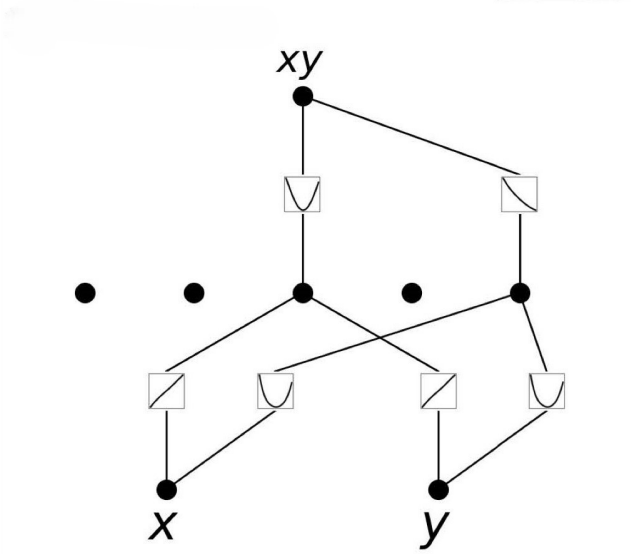
$$f(x) = f(x_1, ..., x_n) = \sum_{q=1}^{2n+1} \phi_q(\sum_{p=1}^{n} \phi_{q,p}(x_p))[6] \quad (5)$$

gdje $\phi_{q,p} : [0, 1] \to \mathbb{R}$ i $\phi_q : \mathbb{R} \to \mathbb{R}$. Ništa nam ne garantira da su funkcije jedne varijable kontinuirane, one mogu biti čak i fraktalne, radi čega se u prošlosti taj teorem nije koristio u neuronskim mrežama. Pošto su sve funkcije funkcije jedne varijable, možemo svaku od njih parametrizirati preko B-spline krivulje prikazane na slici 3. Konstrukcija je takva da uzmemo nekoliko točaka između kojih mreža prilagođava funkcije i spaja ih na tim točkama da bi dobila konačnu funkciju. Iz formule (5) možemo vidjeti kako implementacija Kolmogorov-Arnold neuronske mreže u slučaju $n = 2$ bi učene aktivacijske funkcije prebacila s neurona na bridove tako da se na neuronima samo zbrajaju podatci bez učenih težina, imala bi 2 ulazna neurona, jedan sloj $2n + 1$ skrivenih neurona i jedan izlazni neuron, no znamo da ne možemo ni sa B-spline metodom dobiti fraktalne aktivacijske funkcije što nam govori da trebamo



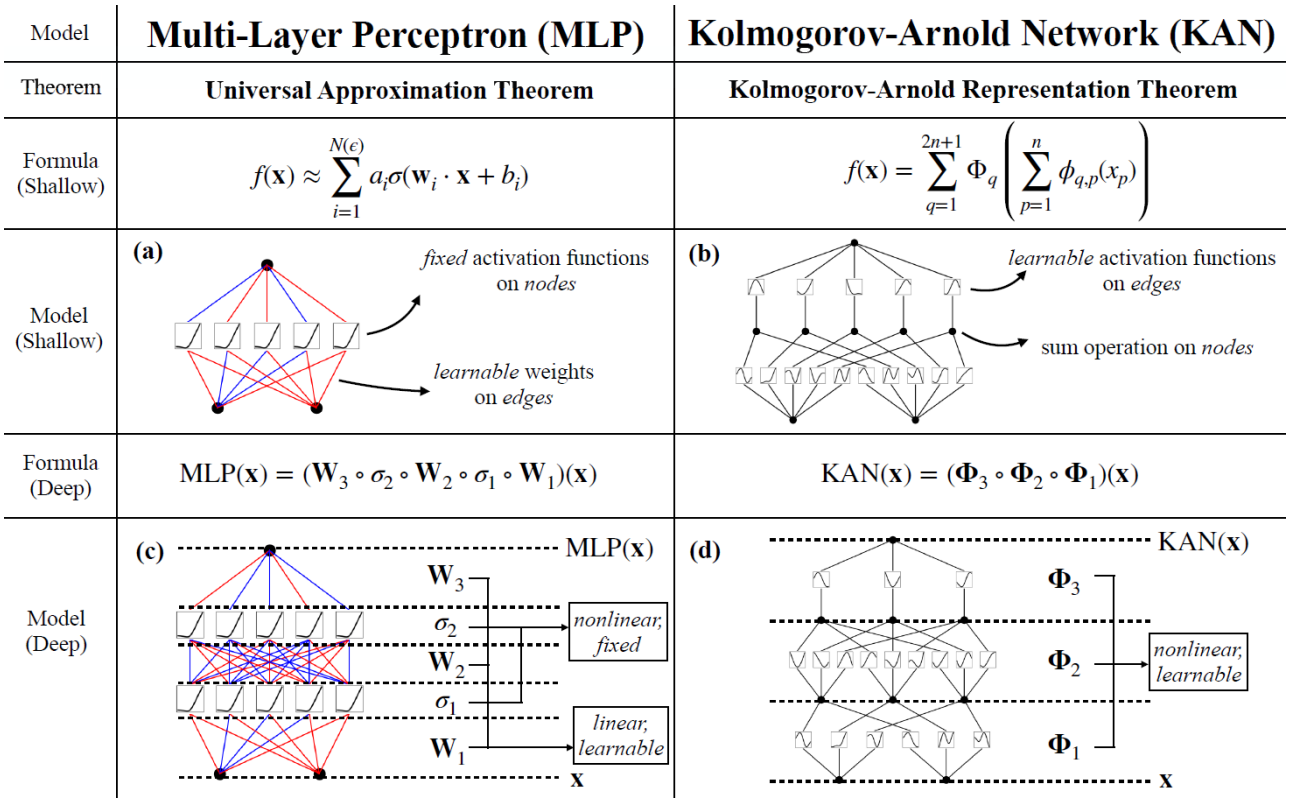
Slika 3: Ilustracija B-spline krivulje. [9]

dublju neuronsku mrežu. Na početku nije jasno kako napraviti dublju KAN mrežu, no spas dolazi u usporedbi s MLP, možemo vidjeti kako su unutarnje funkcije iz kompozicije iz formule (5) mreže s $n_{in} = n$ i $n_{out} = 2n+1$, te vanjske funkcije mreže s $n_{in} = 2n+1$ i $n_{out} = 1$, tako da je formula (5) samo kompozicija dva KAN sloja. Rješenje nam je stoga da samo dodajemo KAN slojeve [6]. Prednost KAN mreže nad



Slika 4: Primjer KAN mreže koja množi dva ulazna broja preko rastava $xy = \frac{(x+y)^2 - x^2 - y^2}{2}$. [6]

MLP je u boljoj nelinearnosti, naime kod MLP nelinearni dio je fiksan, dok kod KAN-a nelinearni dio mreža uči preko B-spline funkcije, stoga je za očekivati kako bi takva mreža trebala biti efikasnija u prilagodbi nelinearnih funkcija kao što je naš GPD. Usporedbu KAN i MLP možemo vidjeti na slici 5. Učena nelinearnost kod KAN-a nam omogućuje da imamo znatno manje mreže nego u slučaju MLP-a što nam povećava interpretabilnost mreže, tj. možemo lagano iščitati



Slika 5: Usporedba KAN i MLP neuronskih mreža. [6]

analitički oblik funkcije. Također bi smanjena veličina mreže trebala povećati brzinu treniranja. Nadamo se i boljem skaliranju preciznosti sa brojem parametara mreže. Od sada na dalje ćemo označavati arhitekturu KAN mreža na sljedeći način: $[a, b, c, d]$, gdje a predstavlja broj neurona u ulaznom sloju, b u sljedećem itd.

4 Metode

U seminaru smo uzeli poznate modele za funkcije koje prilagođavamo te na njih dodali gausijanski šum da simuliramo eksperimentalne podatke. Modeli koje smo koristili nisu bili modeli GPD-ova, nego smo koristili modele Compton Form Faktora (CFF), opservabli koje dobivamo iz DCVS eksperimenata i koje su povezane s GPD-ovima. Također prilagođavamo 2 različite verzije funkcije, dipolnu i eksponencijalnu:

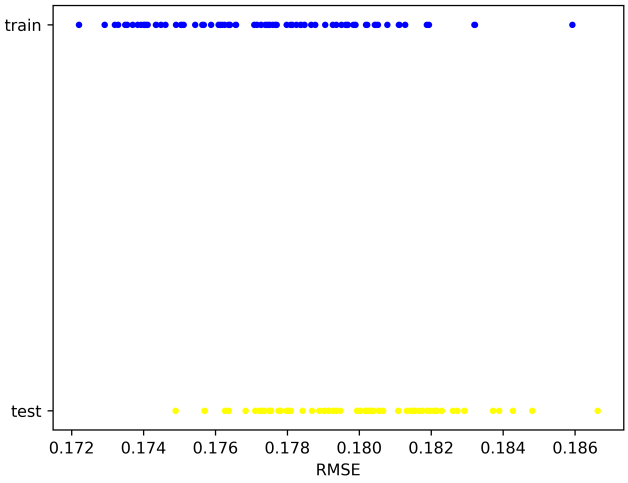
$$\frac{1}{\sqrt{\xi}} \frac{(1-\xi)^3}{(1-t)^2} \tag{6}$$

$$\frac{1}{\sqrt{\xi}} (1-\xi)^3 e^{2t} \tag{7}$$

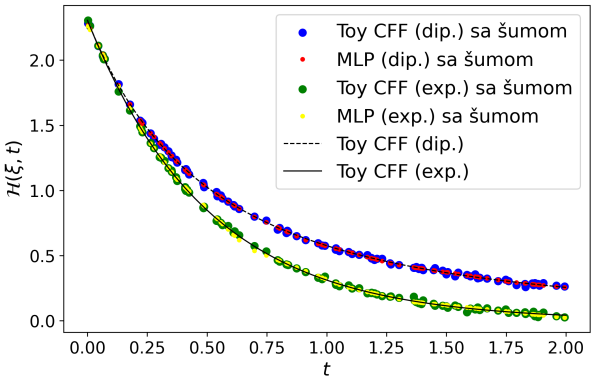
Rasponi vrijednosti ξ i t u kojima razmatramo funkcije su $-2 \leq t \leq 0$ i $0.0001 \leq \xi \leq 0.1$, što su tipični rasponi odgovarajućih kinematičkih veličina u realnim eksperimentima koji se danas izvode. Prvo smo u funkcijama fiksirali varijablu $\xi = 0.1$, dodali šum i prilagodili funkciju kao funkciju samo t te smo isto radili sa zamijenjenim ulogama ξ i $t = -0.7$. Na kraju smo probali prilagoditi funkciju podatcima kao funkciju obje varijable. Svaki navedeni slučaj smo obrađivali u 3 podslučaja: KAN bez ljudske pomoći sa dovoljno dubokom mrežom, KAN s ljudskom pomoći u smislu zadavanja oblika mreže, te KAN s ljudskom pomoći u smislu zadavanja oblika mreže i nekih aktivacijskih funkcija. U podslučaju samog KAN-a bez ljudske pomoći zadajemo dovoljno veliku KAN mrežu te ju treniramo, nakon čega smanjujemo broj neurona u svakom sloju na brojeve koji KAN paket [10] sam misli da su dovoljni i ponovo treniramo. Za sve podslučaje dajemo KAN paketu da sam odabire simboličke izraze za one aktivacijske funkcije u mreži koje nismo mi ručno zadali. Grafički prikazujemo CFF modele u usporedbi s ukupnim simboličkim izrazima koje KAN dobije. Napravili smo prilagodbe i pomoću MLP-a i usporedili smo rezultate dobivene pomoću KAN-a i MLP-a. U prikazanim grafovima podatke dobivene direktno iz poznatih modela funkcija označavamo s 'Toy CFF (dip.)' za dipolnu verziju funkcije, te 'Toy CFF (exp.)' za eksponencijalnu verziju funkcije. Podatke dobivene iz KAN prilagodbe označavamo s 'KAN (dip.)' za KAN treniran s vrijednostima iz 'Toy CFF (dip.)', te 'KAN (exp.)' za KAN treniran s vrijednostima iz 'Toy CFF (exp.)'. Slično označavanje koristimo u MLP slučaju gdje nam 'MLP (dip.)' označava podatke dobivene iz MLP prilagodbe za MLP treniran s vrijednostima iz 'Toy CFF (dip.)', a 'MLP (exp.)' označava podatke dobivene iz MLP prilagodbe za MLP treniran s vrijednostima iz 'Toy CFF (exp.)'. Svi slučaji mogu biti sa dodanim gausijanskim šumom ili bez, što označavamo dodavanjem izraza 'sa šumom' u legendama grafova.

5 Rezultati

Prvo što je potrebno naglasiti je da smo pri radu s KAN mrežom naišli na problem nekonzistentnosti. Pri treniranju mreže s identičnim ulaznim podatcima, te zadavanjem istih seed-ova nasumičnih generatora iz python biblioteka kao što su numpy, torch i random dobivali smo različite izlazne podatke. Navedenu nekonzistentnost možemo vidjeti na grafu na slici 6. Graf prikazuje razmazanost test i train RMSE-ova u 100 različitih treniranja iste mreže sa identičnim početnim uvjetima za autorov primjer [11] prilagodbe funkcije $e^{sin(\pi x_1 + x_2^2)}$ preko KAN mreže veličine [2,5,1]. Razmazanost vrijednosti RMSE upućuje nas na različite izlazne vrijednosti mreže pošto je očekivana izlazna vrijednost u svakom primjeru bila ista. Pri dubljem pogledu u kod KAN paketa nije jasno otkuda ovakva nekonzistentnost, te nam ona onemogućava pravilnu statističku analizu sposobnosti KAN modela da prilagođava funkcije. Također je dobro naglasiti kako smo primjetili da razmazanost RMSE raste s veličinom KAN mreže, no da je uvijek prisutna. Unatoč nekonzistentnosti istrenirali smo KAN mrežu za sve slučaje navedene u 4. odjeljku te prikazujemo rezultate. Na tablici 1 možemo vidjeti numeričke rezultate svih slučajeva. Prvi red tablice je prilagodba na funkciju samo varijable t, možemo vidjeti da su rezultati dosta dobri pošto je RMSE u slučaju KAN i MLP prilagodbe skoro pa jednak standardnoj devijaciji šuma u oba slučaja eksponencijalne i dipolne verzije funkcije, što bi očekivali kod savršene prilagodbe. Međutim primjećujemo manji RMSE u KAN prilagodbi nego što je standardna devijacija šuma, no to pripisujemo malom uzorku od 10 prilagodbi, gdje neke prilagodbe nisu ni provedene radi svojstva KAN mreže da vraća *nan* kao rezultat prilagodbe čak i u slučaju malene standardne devijacije. Skraćenica *nan* stoji za engleski izraz 'not a number' što u prijevodu znači 'nije broj'. Računalni programi vraćaju *nan* vrijednost kada rezultat računalnih operacija ne bude definirani broj, jedan primjer takvog slučaja je $\frac{0}{0}$. Činjenica da KAN prilagodba vraća *nan* znači da se u procesu treniranja provela jedna takva nedefinirana matematička operacija. Radi konstantnog vraćanja *nan* vrijednosti kao rezultata KAN prilagodbe te u nekim slučajevima nemogućnosti da provedemo ijednu prilagodbu radi toga je standardna devijacija šuma postavljena na tako malu vrijednost. U nastavku ćemo koristiti izraz 'oblik' funkcije u usporedbi prilagođenih funkcija s toy modelima, pri čemu mislimo na grafički izgled istih. Slike 7 i 8 prikazuju grafove prilagodbe prvog retka u tablici 1, gdje možemo vidjeti kako MLP u biti veoma dobro prilagođava funkciju u oba slučaja. S druge strane KAN uzima prosjeke, u obliku konstantnih funkcija iznosa konstante 0.830, za dipolnu verziju. Za eskponencijalnu verziju većinom uzima funkcije koje nisu trivijalne, te varira oko 'Toy CFF (exp.)' oblika. Primjer formule KAN prilagođene funkcije za eksponencijalnu verziju je $1.695 \cdot (0.583 \cdot x_1 - 1)^2 + 0.073$, gdje smo sve brojeve zaokružili na treću decimalu te ćemo tako raditi i u svim budućim prikazima prilagođenih formula. Možemo vidjeti kako KAN prilagodba umjesto eksponencijale dobiva kvadratnu funkciju koja je modelu dovoljno slična na zadanom intervalu. Navedeni rezultat KAN-a prikazuje njegovu privrženost jednostavnijim oblicima funkcija kao što je eksponencijala. Na slici 7 primjećujemo i nacrtani gausijanski šum sa standardnom devijacijom iznosa 0.01 te će nam upravo taj graf biti ilustracija svih manjih šumova kroz seminar pošto se smatralo pretjeranim prikazati grafove šumova za svaki podslučaj. Za podslučaj funkcije $f(t)$ sa zadanim oblikom KAN mreže imamo sličnu situaciju kao u prethodnom podslučaju, numerički MLP i KAN imaju slične RMSE te su blizu savršenih fitova, gdje ponovno imamo RMSE manji od standardne devijacije radi još uvijek relativno malog uzorka od 100 prilagodbi. Standardnu devijaciju smo podigli na 0.2 pošto je KAN model stabilniji u slučaju jednostavnijih mreža te ne vraća *nan* vrijednost toliko često.



Slika 6: Graf koji prikazuje nekozistentnost KAN mreže u obliku razmazanosti RMSE.

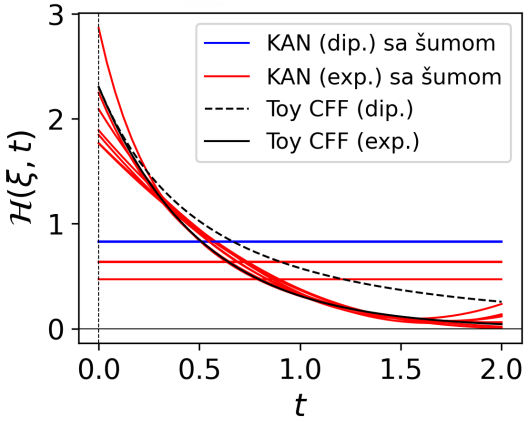


Slika 7: Graf podataka sa šumom standardne devijacije 0.01, bez šuma i MLP prilagodbe $f(t)$ s $\xi = 0.1$ na podatke sa šumom za eksponencijalnu i dipolnu verziju funkcije.

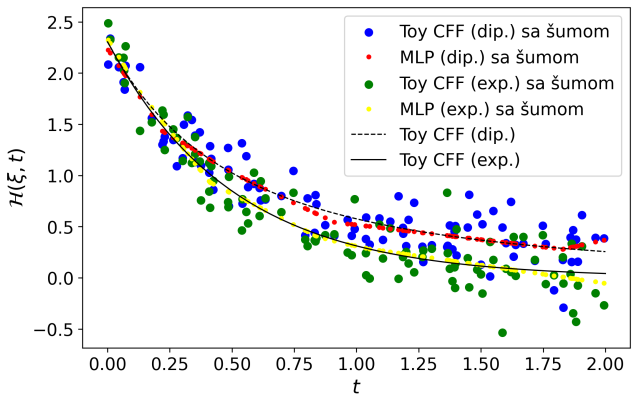
Na slikama 9 i 10 vidimo MLP i KAN prilagodbe grafički te ponovo vidimo kako su oblici MLP funkcija , u usporedbi s KAN-om, sličniji 'Toy CFF' oblicima, dok KAN ponovno za dipolnu verziju funkcije uzima

	OBLIK KAN-a	SEED KAN-a	STD DEVIJACIJA	PROSJEČNI RMSE	OBLIK MLP-a	MLP RMSE
f(t) KAN SAM	[1,4,4,1]	0	0.01	DIP: 0.0089 EXP: 0.0093	[1,50,1]	DIP: 0.01 EXP: 0.01
f(t) KAN ZADAN OBLIK	[1,1]	0	0.2	DIP: 0.1854 EXP: 0.1894	[1,50,1]	DIP: 0.1847 EXP: 0.1847
f(t) ZADANE FUNKCIJE	[1,1]	0	0.2	DIP: 0.1913 EXP: 0.1943	[1,50,1]	DIP: 0.1847 EXP: 0.1847
f(ξ) KAN SAM	[1,4,4,1]	1	0.03	DIP: 0.5414 EXP: 0.7153	[1,50,1]	DIP: 5.5986 EXP: 5.5986
f(ξ) KAN ZADAN OBLIK	[1,2,2,1]	1	0.5	DIP: 0.6349 EXP: 0.5140	[1,50,1]	DIP: 5.7227 EXP: 5.7227
f(ξ) KAN ZADANE FUNKCIJE	[1,2,2,1]	1	0.5	DIP: 0.8740 EXP: 0.6755	[1,50,1]	DIP: 5.7227 EXP: 5.7228
f(t,ξ) KAN SAM	[2,4,4,4,1]	DIP: 1 EXP: 0	0.00001	DIP: 0.38787 EXP: 0.1649	[2,100,100,1]	DIP: 1.7271 EXP: 4.0309
f(t,ξ) KAN ZADAN OBLIK	[2,4,4,1]	DIP: 1 EXP: 0	0.00001	DIP: 0.1938 EXP: 0.0919	[2,100,100,1]	DIP: 1.7271 EXP: 4.0309
f(t,ξ) KAN ZADANE FUNKCIJE	[2,4,4,1]	DIP: 1 EXP: 0	0.00001	DIP: 0.5334 EXP: 0.4402	[2,100,100,1]	DIP: 1.7271 EXP: 4.0309

Tablica 1: Tablica koja prikazuje numeričke rezultate KAN i MLP prilagodbi. Prikazani su RMSE, standardna devijacija šuma te oblici mreža za svaki slučaj. Također je prikazan seed KAN-a za svaki podslučaj. Naslov retka opisuje koji podslučaj promatramo, te u istom retku iznosimo i podatke za MLP mrežu s istim ulaznim podatcima radi lakše usporedbe. Jedini atribut koji KAN i MLP mreže u istom retku dijele je standardna devijacija.

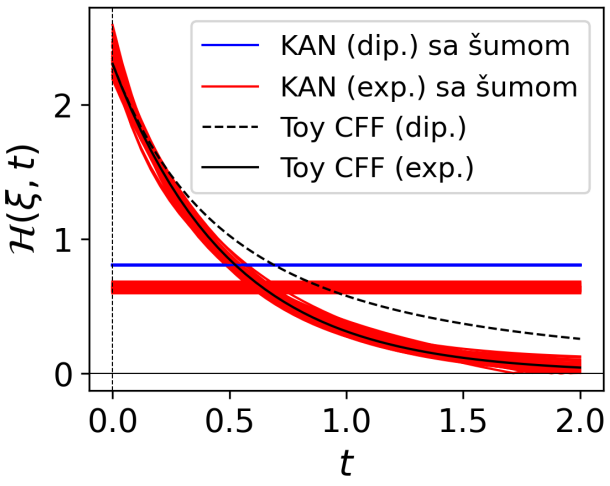


Slika 8: Graf KAN prilagodbe $f(t)$ s $\xi = 0.1$ bez ljudske pomoći na podatke sa šumom standardne devijacije 0.01 i usporedba s pravim funkcijama za dipolnu i eksponencijalnu verziju funkcije.



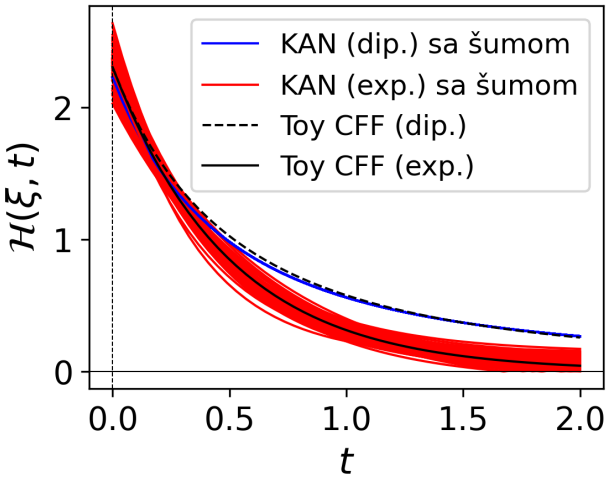
Slika 9: Graf podataka sa šumom standardne devijacije 0.2, bez šuma i MLP prilagodbe $f(t)$ s $\xi = 0.1$ na podatke sa šumom za eksponencijalnu i dipolnu verziju funkcije.

samo prosjek, a za eksponencijalnu verziju u nekoliko slučaja dobije oblik 'Toy CFF (exp.)' s formulom $-0.364 + \frac{2.588}{(-0.888 \cdot x_1 - 1)^2}$ ponovo promašujući eksponencijalu, isto vidimo veću popunjenost pojasa pogreške radi većeg broja prilagodbi. Formula koju smo dali za eksponencijalnu verziju prilagodbe je samo primjer mnoštva različitih formula koje dobijemo prilagodbom. Pogoršanje KAN modela u obliku prilagođenih funkcija s obzirom na prethodni slučaj pripisujemo mnogo većoj standardnoj devijaciji šuma. Na grafu 9 možemo vidjeti jači šum koji će nam služiti kao ilustrativni primjer u slučajevima gdje šum nije zanemariv. Nastavljajući na 3. redak tablice 1 imamo KAN model sa zadanim



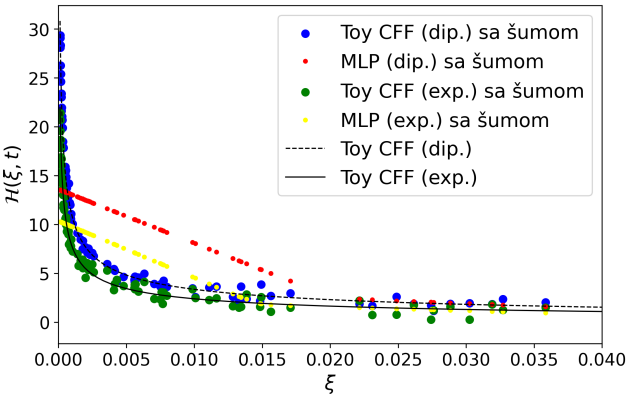
Slika 10: Graf KAN prilagodbe $f(t)$ s $\xi = 0.1$ sa zadanim oblikom KAN mreže na podatke sa šumom standardne devijacije 0.2 i usporedba s pravim funkcijama za dipolnu i eksponencijalnu verziju funkcije.

oblikom mreže i zadanom aktivacijskom fukcijom, sve numeričke vrijednosti su veoma slične slučaju sa samo zadanim oblikom, standardna devijacija je jednaka kao što je i broj prilagodbi, te ponovno imamo RMSE manji od standardne devijacije šuma iz istih razloga kao u prethodnom slučaju. Graf MLP prilagodbe je



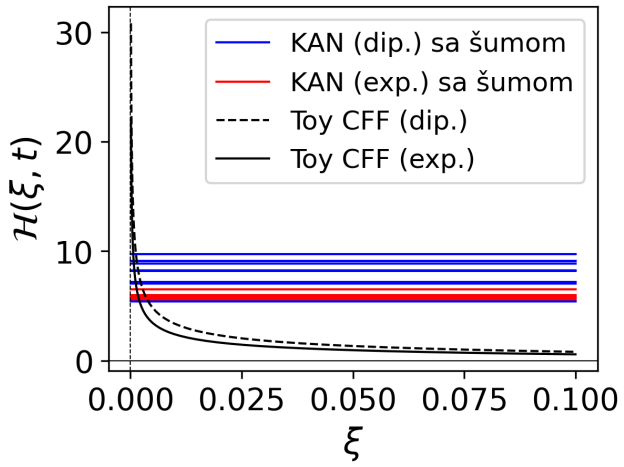
Slika 11: Graf KAN prilagodbe $f(t)$ s $\xi = 0.1$ sa zadanim oblikom KAN mreže i zadanom aktivacijskom funkcijom na podatke sa šumom standardne devijacije 0.2 i usporedba s pravim funkcijama za dipolnu i eksponencijalnu verziju funkcije.

isti kao u prethodnom slučaju te ga se može vidjeti na slici 9. KAN graf, prikazan na slici 11, u ovom slučaju točno pogađa oblike funkcija i za dipolnu i za eksponencijalnu verziju funkcije, s veoma malim pojasom greške u dipolnoj verziji. Ovaj rezultat nije iznenađujuć pošto smo modelu sami rekli koje su funkcije u pitanju pa je on samo morao prilagoditi parametre. Radi ručnog zadavanja aktivacijskih funkcija formula prilagodbi u ovom slučaju je kao i očekivana i za dipolnu $(0.025 + \frac{2.097}{(-x_1-0.970)^2})$ i za eksponencijalnu $(-0.021 + 2.260 \cdot e^{-1.961 \cdot x_1})$ verziju funkcije, gdje su ponovo dani samo primjeri mnoštva različitih formula. Sada prelazimo s $f(t)$ na $f(\xi)$ i gledamo prvo podslučaj potpuno automatiziranog KAN-a. Osvrčući se ponovo na tablicu 1 vidimo da su RMSE vrijednosti za KAN čak red veličine manje nego za MLP u obje verzije funkcije. S druge strane vidimo da je RMSE za KAN za više od red veličine veći od standardne devijacije, što nam govori da je KAN dosta lošiji u prilagodbi CFF kao funkcije ξ nego kao funkcije t. To pogoršanje s obzirom na prošli slučaj dolazi od kompleksnosti ξ dijela CFF-a kao što vidimo u formulama (6) i (7). Naime da bi KAN množio dvije funkcije potreban mu je oblik kao na slici 4 što je svakako mnogo kompliciranije od $[1, 1]$ oblika KAN-a koji je bio dovoljan za slučaj $f(t)$. Možemo i primjetiti kako su RMSE za dipolne i eksponencijalne verzije prilagodbi jednake u MLP slučaju, a različite u KAN slučaju, unatoč prilagodbe istog oblika funkcije. To je posljedica spomenute nekonzistentnosti KAN-a koja će praktički onesposobiti KAN u prilagodbi kompliciranijih funkcija. Grafovi su prikazani na slikama 12



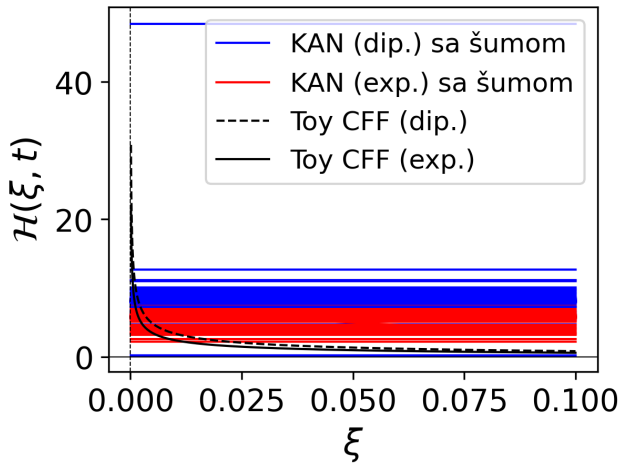
Slika 12: Graf podataka sa šumom standardne devijacije 0.5, bez šuma i MLP prilagodbe $f(\xi)$ s $t = -0.7$ na podatke sa šumom za eksponencijalnu i dipolnu verziju funkcije.

i 13. Treba napomenuti kako graf za MLP je slučaj zadanog KAN oblika koji ima veću standardnu devijaciju iznosa 0.5, no oblik MLP prilagodbe je skoro sasvim isti te jedino što se mijenja je da su podatci sa šumom manje raštrkani. Ponovo vidimo kako je oblik funkcije koju MLP mreža prilagodi u boljem slaganju s 'Toy CFF', no u ovom kompliciranijem slučaju počinje znatno odstupati od pravih CFF-ova, sa druge strane KAN ponovo uzima prosjeke funkcija u obliku konstantnih vrijednosti. Unatoč tome da više strukture, u smislu većeg odstupanja od konstanih vrijednosti, ima MLP prilagođena funkcija, RMSE je manji za KAN prilagodbu radi toga što u području velike gustoće točaka, tj. manjim vrijednostima ξ MLP počinje znatno odstupati od 'Toy CFF' modela što posljedično povećava RMSE, dok KAN uzima optimalnu konstantnu vrijednost za minimizaciju RMSE. Također treba uzeti u obzir da unatoč kompliciranim-



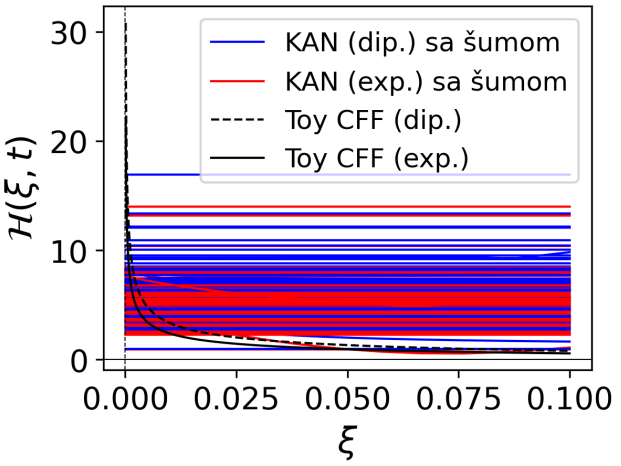
Slika 13: Graf KAN prilagodbe $f(\xi)$ s $t = -0.7$ bez ljudske pomoći na podatke sa šumom standardne devijacije 0.03 i usporedba s pravim funkcijama za dipolnu i eksponencijalnu verziju funkcije.

joj funkciji, oblik MLP mreže ostavljamo istim kao u slučaju $f(t)$ te je moguće da je potrebna veća mreža za ovaj zadatak. Prelazeći na slučaj $f(\xi)$ s zadanim oblikom KAN mreže, numerički RMSE za KAN i za MLP ostaje otprilike isti kao u prethodnom slučaju unatoč mnogo većem šumu. Zaključujemo stoga da nepreciznost MLP-a ne dolazi od šuma nego od same mreže, te da smo za KAN odabrali puno bolji oblik mreže. Ponovno vidimo nekonzistentnost KAN-a u različitim



Slika 14: Graf KAN prilagodbe $f(\xi)$ s $t = -0.7$ sa zadanim oblikom KAN mreže na podatke sa šumom standardne devijacije 0.5 i usporedba s pravim funkcijama za dipolnu i eksponencijalnu verziju funkcije.

RMSE vrijednostima za dipolnu i eksponencijalnu inačicu. MLP graf je već prikazan na slici 12, te KAN graf možemo vidjeti na slici 14, ponašanje grafa je isto kao i u slučaju potpuno autonomnog KAN-a, samo što imamo 100 prilagodbi umjesto 10. Kod podslučaja sa zadanim aktivacijskim funkcijama nismo zadali sve aktivacijske funkcije u mreži nego samo one u koje smo bili razumno sigurni. Na ovom mjestu se ponovo vidi problem s nekonzistentnošću KAN-a, gdje je model mijenjao položaje određenih funkcija u mreži i same aktivacijske funkcije od prilagodbe do prilagodbe što je znatno otežalo ljudsko zadavanje aktivacijskih funkcija radi čega se vidi porast u RMSE-u u 6. retku tablice 1 u odnosu na prethodni redak. KAN graf se može vid-



Slika 15: Graf KAN prilagodbe $f(\xi)$ s $t = -0.7$ sa zadanim oblikom KAN mreže i dijelom zadanim aktivacijskim funkcijama na podatke sa šumom standardne devijacije 0.5 i usporedba s pravim funkcijama za dipolnu i eksponencijalnu verziju funkcije.

jeti na slici 15 gdje vidimo veći pojas pogreške u skladu s povećanjem RMSE, MLP graf je isti kao u prethodnom slučaju te se može vidjeti na slici 12. Konačno prelazimo na prilagodbu $f(\xi, t)$ funkcija, za koju smo uzeli veću MLP mrežu u nadi poboljšanja preciznosti. U svakom podslučaju ovakve prilagodbe standardna devijacija je zanemarivo mala, iznosa 0.00001, radi nestabilnosti na koje smo naišli pri zadavanju većih devijacija u obliku *nan* rezultata, stoga možemo ovo praktički smatrati prilagodbom bez šuma. U sva 3 podslučaja smo radili samo 10 prilagodbi radi vremena koje je potrebno za pojedinu prilagodbu, te se u potpuno automatiziranom KAN-u kao i prije provelo još manje prilagodbi radi vraćanja *nan* vrijednosti, tako da nam je statistički uzorak malen. Također u sva 3 podslučaja možemo vidjeti kako MLP ima više strukture od prosječne KAN prilagodbe, no ponovo na cijenu većeg RMSE. Osvrčući se na grafove CFF-a u ovisnosti o t sa zadanom vrijednošću $\xi = 0.1$, u slučaju potpuno automatiziranog KAN-a većina dipolnih prilagodbi su konstantne vrijednosti osim 2 slučaja koja poprimaju netrivijalni oblik, s primjerom formule: $1.315 \cdot \cos(0.230 \cdot t + 3.298) + 4.897$, dok za eksponencijalnu prilagodbu vidimo samo konstantne vrijednosti. Vidimo kako model ne dobiva traženi dipolni oblik, nego ga zamjenjuje s trigonometrijskom funkcijom. Sve prilagođene funkcije su iznad vrijednosti pravih CFF-ova radi činjenice da smo prilagođavali CFF-ove kao funkcije dviju varijabli, te pošto vrijednost CFF-a u ξ dimenziji brže raste model pokušava uzeti efektivni prosjek. Što više pomažemo modelu to se više može vidjeti odstupanje od konstantnih funkcija, gdje u slučaju zadavanja samo oblika mreže dobivamo prim-

jer dipolne netrivijalne funkcije:

$$5.724 + 0.456 \cdot \cos 1.344 \cdot t - 0.639, \tag{8}$$

kao i eksponencijalne:

$$0.525 + 18.393 \cdot e^{-2.377 \cdot t}, \tag{9}$$

te u slučaju zadavanja oblika i nekih aktivacijskih funkcija nekolicina njih poprima netrivijalne oblike kao što je za dipolnu verziju:

$$-25.806 + \frac{28.693}{\sqrt{-0.025 \cdot \mathrm{Ai}(-0.101 \cdot t - 1.997) - 0.599 \cdot \sin(0.219 \cdot t + 8.922) + 1 - 0.015 \cdot e^{-1.647 \cdot t}}}, \tag{10}$$

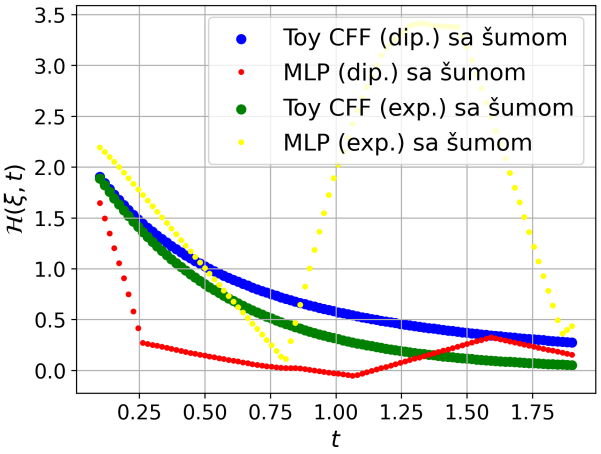
gdje s Ai označavamo Airyjevu funkciju, te za eksponencijalnu verziju:

$$28.157 \cdot \cos(0.125 \cdot t + 0.063 \cdot \cos(2.987 \cdot \xi + 5.767) + 0.007 \cdot \cos(1.388 \cdot t + 9.650) + 0.527 - 0.365 \cdot e^{-10.011 \cdot t}) - 22.068. \tag{11}$$

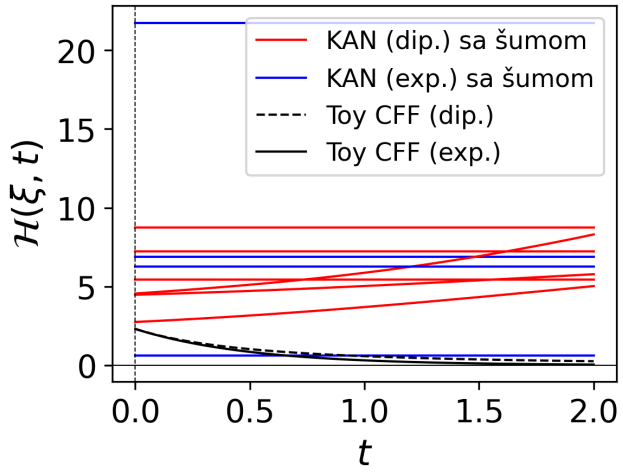
Možemo vidjeti kako u slučaju zadavanja samo oblika mreže uspijemo čak dobiti dobru eksponencijalnu ovisnost o t. Preferenciju KAN modela prema jednostavnijim funkcijama možemo vidjeti preko znatno manjeg RMSE eksponencijalne inačice funkcije naspram dipolnoj inačici u svim podslučajevima. Također vidimo smanjenje RMSE s zadavanjem oblika mreže, no povećanje istog na vrijednosti iznad skroz automatiziranog KAN-a ponovo radi njegove nekonzistentnosti i mijenjanja aktivacijskih funkcija unatoč identičnim ulaznim parametrima, onemogućavajući ručno zadavanje istih. MLP prilagodbe kao što smo rekli imaju više strukture od KAN prilagodbi u smislu odstupanja od konstantnih funkcija, no vidimo kako su prilagodbe puno lošije nego za slučajeve sa funkcijama samo jedne varijable, posebno izraženo u eksponencijalnoj inačici funkcije gdje vidimo skok na $t = 1.3$. Prelazeći na grafove CFF-a u ovisnosti o ξ sa zadanom vrijednošću $t = -0.7$ vidimo da svaka razina ljudske pomoći vodi do konstantnih funkcija osim onog s najvećom razinom ljudske pomoći, gdje vidimo jedan pokušaj prilagodbe netrivijalne funkcije u eksponencijalnoj inačici CFF-a koji se na žalost uopće ne slaže s 'Toy CFF (exp.)'. Ovu neovisnost prilagođenih funkcija o variabli ξ smo mogli vidjeti i u dobivenim formulama za iste, gdje nam u skladu sa grafovima samo u slučaju najveće ljudske pomoći funkcija (11) ovisi o ξ. Vrijednosti konstantnih funkcija su sada prema nižem dijelu grafa kako bi se uskladili sa t dimenzijom. MLP prilagođena funkcija ponovo ima bolju strukturu, no najviše odstupa od prave vrijednosti u najgušćem području ξ vrijednosti čime znatno povećava RMSE. Grafovi svih navedenih slučaja su u prilogu.

6 Zaključak

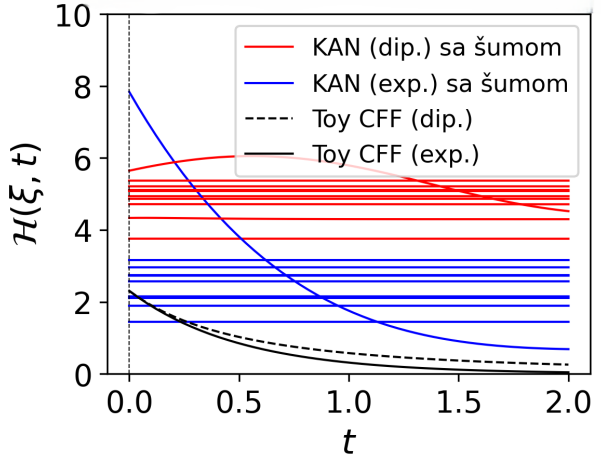
U seminaru smo objasnili fizikalnu motivaciju iza prilagodbe funkcija putem KAN neuronskih mreža, te smo ukratko opisali osnove rada istih. Objasnili smo kako bi, u teoriji, sposobnost KAN-a da uči nelinearne parametre mogla biti veoma korisna u prilagodbi nelinearnih funkcija. Proveli smo analizu sposobnosti KAN mreže da prilagodi funkcije jedne varijable i dvije varijable s različitim razinama ljudske pomoći. Također smo naišli na problem nekonzistentnosti KAN modela koji nam onemogućava pravilnu statističku analizu. Vidjeli smo kako je model funkcionirao samo u slučaju najjednostavnije funkcije $\frac{1}{\sqrt{0.1}}(0.9)^3 e^{2t}$, te u kompliciranijim slučajevima jedino za što je sposoban je vraćanje konstantnih vrijednosti. Glavni problem modela je nekonzistentnost koja onemogućava ljudsku asistenciju modelu, jednu od glavnih prednosti koju su autori naveli. Zaključujemo da unatoč nadi koju smo imali za sposobnost nelinearne prilagodbe KAN modela, implementacija modela u kodu s kojim smo radili je problematična i onesposobljuje KAN model.



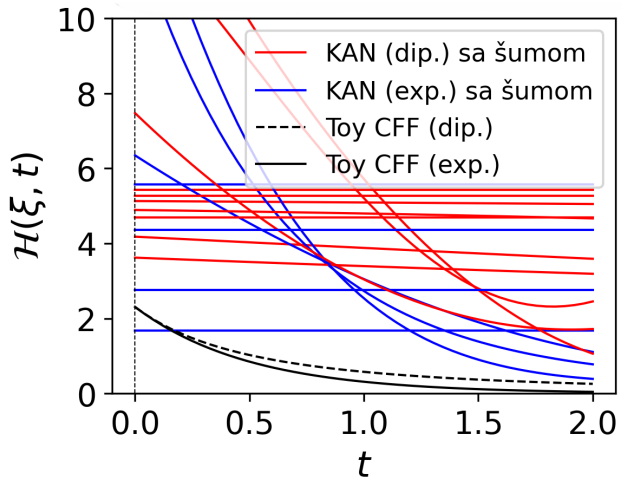
Slika 16: Graf MLP prilagodbe $f(\xi, t)$, na podatke sa šumom, u ovisnosti o t sa zadanim $\xi = 0.1$ i prikazanim podatcima sa šumom standardne devijacije 0.00001



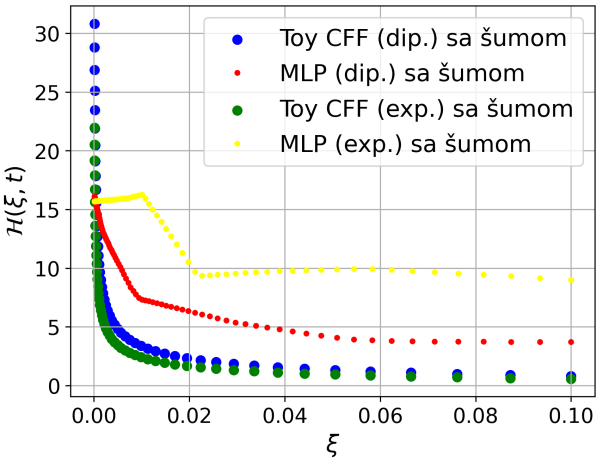
Slika 17: Graf KAN prilagodbe $f(\xi, t)$ bez ljudske pomoći, na podatke sa šumom standardne devijacije 0.00001, u ovisnosti o t sa zadanim $\xi = 0.1$.



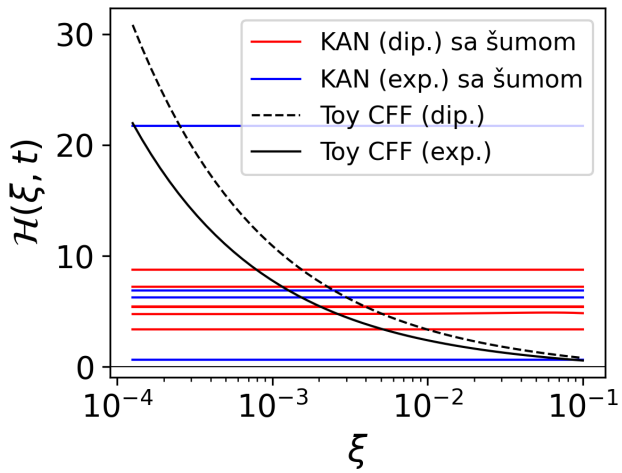
Slika 18: Graf KAN prilagodbe $f(\xi, t)$ s zadanim oblikom mreže, na podatke sa šumom standardne devijacije 0.00001, u ovisnosti o t sa zadanim $\xi = 0.1$.



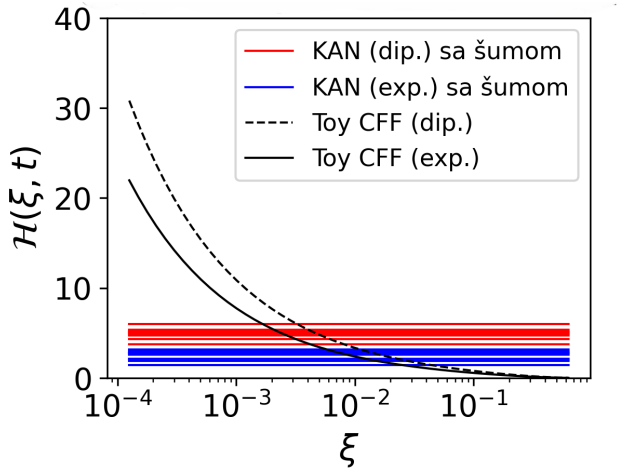
Slika 19: Graf KAN prilagodbe $f(\xi, t)$, s zadanim oblikom mreže i nekim aktivacijskim funkcijama, na podatke sa šumom standardne devijacije 0.00001, u ovisnosti o t sa zadanim $\xi = 0.1$.



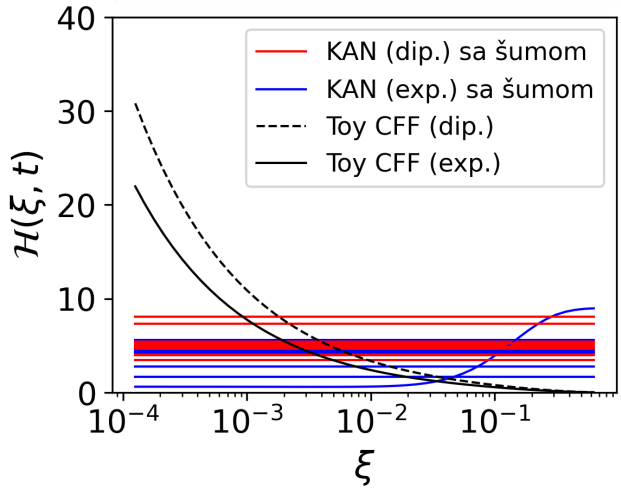
Slika 20: Graf MLP prilagodbe $f(\xi, t)$, na podatke sa šumom, u ovisnosti o ξ sa zadanim $t = -0.7$ i prikazanim podatcima sa šumom standardne devijacije 0.00001



Slika 21: Graf KAN prilagodbe $f(\xi, t)$ bez ljudske pomoći, na podatke sa šumom standardne devijacije 0.00001, u ovisnosti o ξ sa zadanim $t = -0.7$.



Slika 22: Graf KAN prilagodbe $f(\xi, t)$ s zadanim oblikom mreže, na podatke sa šumom standardne devijacije 0.00001, u ovisnosti o ξ sa zadanim $t = -0.7$.



Slika 23: Graf KAN prilagodbe $f(\xi, t)$ s zadanim oblikom mreže i nekim aktivacijskim funkcijama. na podatke sa šumom standardne devijacije 0.00001, u ovisnosti o ξ sa zadanim $t = -0.7$.

Zahvale

Posebnu zahvalnost dugujem mentoru Krešimiru Kumeričkom za njegovo stručno usmjeravanje i podršku tijekom rada na ovom seminaru.

Literatura

[1] K. Kumerički, *Predavanja iz Fizike Elementarnih čestica*, Neobjavljeno

[2] M. Thomson, *Modern Particle Physics*, Cambridge University Press, 2013.

[3] V. D. Burkert, L. Elouadrhiri, F. X. Girod, C. Lorcé, P. Schweitzer, and P. E. Shanahan, *Colloquium: Gravitational Form Factors of the Proton*, arXiv:2303.08347v3

[4] D. Bergmann, C. Stryker, *What is backpropagation?*, [Online], Dostupno: `https://www.ibm.com/think/topics/backpropagation`.

[5] B. Scarff, *Understanding Backpropagation*, [Online], Dostupno: `https://towardsdatascience.com/understanding-backpropagation-abcc509ca9d0`.

[6] Z. Liu, Y. Wang, S. Vaidya, F. Ruehle, J. Halverson, M. Soljačić, T. Y. Hou, and M. Tegmark, *KAN: Kolmogorov–Arnold Networks*, arXiv:2404.19756v4

[7] A. Von Felbert, *The Universal Approximation Theorem*, [Online], Dostupno: `https://www.deep-mind.org/2023/03/26/the-universal-approximation-theorem/#Universal_Approximation_Theorem`.

[8] Suradnici Wikipedije, *Universal approximation theorem*, Wikipedia, The Free Encyclopedia, 04.12.2024, `https://en.wikipedia.org/wiki/Universal_approximation_theorem`, Pristup: 30.12.2024.

[9] Suradnici Wikipedije, *B.spline*, Wikipedia, The Free Encyclopedia, 06.07.2024, `https://en.wikipedia.org/wiki/B-spline#`, Pristup: 30.12.2024.

[10] Z. Liu, pykan 0.2.8, `https://kindxiaoming.github.io/pykan/intro.html`

[11] Z. Liu, pykan documentation, *Hello, KAN!*, `https://github.com/KindXiaoming/pykan`