Univerzitet u Sarajevu
Elektrotehnički fakultet
Odsjek za računarstvo i informatiku

Nesterovljev ubrzani gradijentni metod za minimizaciju

Seminarski rad iz Numeričkih algoritama

Studenti:
Aid Mustafić
Zlatan Ljutika

Profesor:
Red. prof. dr Željko Jurić.

Sarajevo, januar 2026.

Sažetak

U ovom radu ćemo se baviti metodom za nalaženje minimuma funkcije, zasnovanu na Nesterovljevom ubrzanom gradijentnom. Na početku rada ćemo se osvrnuti na teoretsku pozadinu navedenog algoritma, područija na kojima je najprimjenjeniji. Sljedeće čime ćemo se baviti je implementacijom navedenog algoritma u Julia programskom jeziku, ćiju ćemo korektnost testirati ručno na nekim proizvoljnim funkcijama, kao i na Rosenbrockovoj funkciji. U zaključku ćemo se osvrnuti na to koliko je naša implementacija algoritma dobra I može li se iskoristiti za svrhe u kojima se ovaj algoritam najviše koristi.

Ključne riječi: Nesterovljeva metoda, ubrzani gradijentni spust, numerička optimizacija, konveksna optimizacija, duboko učenje, FISTA, momentum, metode prvog reda, Julia

Abstract

In this paper, we are going to be discussing a method for finding the minimum of function called Nesterov Accelerated Gradient. Firstly, we will delve into the theory behind the given algorithm and in which fields is it the most used. Secondly, we will implement the algorithm in the Julia programming language, whose correctness we will test by hand on some arbitrary functions, as well as with Rosenbrock's function. In conclusion, we will look back at how our implementation holds up in use cases this algorithm is used the most.

Keywords: Nesterov accelerated gradient, numerical optimization, convex optimization, deep learning, FISTA, momentum, first-order methods, Julia

Sadržaj

Popis slika

Popis tabela

Poglavlje 1

Uvod

1.1 Obrazloženje teme

Problem minimizacije funkcija predstavlja jedan od najvažnijih problema primijenjene matematike. U najopštijem obliku, cilj je odrediti tačku $\mathbf{x} \in \mathbb{R}^n$ u kojoj funkcija $f(\mathbf{x})$ dostiže minimalnu vrijednost. Ovakvi problemi javljaju se u brojnim oblastima primijenjene matematike, fizike, ekonomije i savremenog mašinskog učenja.

Jedna od osnovnih metoda za rješavanje problema minimizacije diferencijabilnih funkcija jeste gradijentni metod. Gradijent funkcije predstavlja vektor parcijalnih izvoda i pokazuje pravac najvećeg rasta funkcije, pa se minimizacija postiže kretanjem u suprotnom smjeru gradijenta. Iako je metoda jednostavna i široko primjenjiva, njena konvergencija može biti spora, naročito kod loše uslovljenih problema.

Radi poboljšanja performansi razvijene su metode koje koriste dodatne informacije iz prethodnih iteracija. Jedan od prvih pristupa je metod momentuma, koji uvodi "inerciju" u proces optimizacije i na taj način ublažava oscilacije i ubrzava kretanje kroz ravnije dijelove funkcije.

Nesterov ubrzani gradijent metod predstavlja unapređenje klasičnog momentum pristupa. Za razliku od standardnog momentuma, kod Nesterovog metoda gradijent se računa u unaprijed predviđenoj tački, čime se dobija efikasniji i stabilniji postupak. Ova modifikacija dovodi do značajnog teorijskog poboljšanja brzine konvergencije.

Cilj ovog rada je da se prikaže teorijska osnova Nesterovog ubrzanog gradijentnog metoda, objasne njegove interpretacije, analiziraju osobine konvergencije, te implementira algoritam i ispita njegovo ponašanje na Rosenbrockovoj funkciji.

1.2 Struktura rada

Rad je organizovan u 4 poglavlja. U prvom poglavlju se opisuje struktura rada i teoretska osnova potrebna za razumjevanje Nesterovog ubrzanog gradijenta, kao i zašto i kada bi se trebao koristiti.

U drugom poglavlju će biti prikazane tri tipa implementacija Nesterovog ubrzanog gradijenta, to jest standardna varijanta, Sutskeverova modifikacija i Bengiova formulacija metode. Zatim će biti prikazan jedan praktičan primjer rada implementiranih funkcija i usporediće se njihovi rezultati.

Treće poglavlje će govoriti o primjenama algoritma u praksi, kao što su u dubokom učenju, modernim softverskim okvirima, optimizatorima te obradi signala i kompresovanog opažanja.

U četvrtom smo oformili zaključak.

1.3 Teoretske osnove

1.3.1 Metod momentuma

Jedno od prvih poboljšanja klasičnog gradijentnog metoda predstavlja metod momentuma, koji je uveo Boris Polyak. Osnovna ideja ovog pristupa jeste da se pri ažuriranju iteracija ne koristi samo trenutna vrijednost gradijenta, već i informacija o prethodnom kretanju algoritma.

$$v_{t+1} = \mu v_t - \varepsilon \nabla f(\theta_t) \tag{1.1}$$

$$\theta_{t+1} = \theta_t + v_{t+1} \tag{1.2}$$

Gdje $\varepsilon > 0$ predstavlja parametar koraka, $\mu \epsilon [0, 1]$ parametar momentuma, a $\nabla f(\theta_t)$ gradijent.[1]

1.3.2 Nesterov ubrzani gradijentni metod

Metod Nesterovog ubrzaniog gradijenta je metod koji izgleda kao metod momentuma, ali nije u potpunosti isti.[2]

$$v_{t+1} = \mu v_t - \varepsilon \nabla f(\theta_t + \mu v_t) \tag{1.3}$$

$$\theta_{t+1} = \theta_t + v_{t+1} \tag{1.4}$$

Razlika je u argumentu gradijenta $+\mu v_t$, omogućuje metodu da brže popravi putanju kretanja od metod momentuma. [1]

Poglavlje 2

Implementacija algoritma

U prethodnom poglavlju izložene su teoretske osnove Nesterovljeve ubrzanog gradijentnog metoda za minimizaciju, uključujući formalnu analizu konvergencije i poređenje sa klasičnim gradijentnim metodama. U ovom poglavlju prelazimo na **konkretnu implementaciju metode** u programskom jeziku *Julia*, uz primjenu na poznatom problemu iz oblasti numeričke optimizacije. Cilj je demonstrirati kako i na koje se načine teoretska formulacija prevodi u funkcionalan kod, te vizualno potvrditi konvergentno ponašanje opisano u teoriji.

Implementacija metode na osnovu teoretskih osnova iz prethodnog poglavlja se može predstaviti sljedećim programskim kodom:

Program 2.1: Standardna Nesterovljeva metoda (NAG)

```
function nag(x0, func, gradient;
             learning_rate  = 0.001,
             momentum_coeff = 0.9,
             max_iter       = 5000,
             eps            = 1e-7)

    v    = zeros(length(x0))
    x    = copy(x0)

    for i in 1:max_iter
        look_ahead = x .+ momentum_coeff .* v
        grad_x     = gradient(look_ahead)
        if norm(grad_x) < eps
            break
        end
        v = momentum_coeff .* v .- learning_rate .* grad_x
        x = x .+ v
    end

    return (x=x, f=func(x))
end
```

Funkcija prima početnu tačku `x0`, funkciju cilja `func` i njen gradijent `gradient`, uz opci-

onalne parametre: stopu učenja α (`learning_rate`), koeficijent momentuma μ (`momentum_coeff`), maksimalni broj iteracija (`max_iter`) i toleranciju zaustavljanja (`eps`).

Unutar petlje, metoda prvo računa *look-ahead* poziciju, zatim evaluira gradijent na toj poziciji, te ažurira vektor brzine spusta v i trenutnu poziciju x. Petlja se prekida kada norma gradijenta padne ispod zadate tolerancije `eps`, ili kada se dostigne maksimalni broj iteracija `max_iter`.

U ovoj i narednim implementacijama pretpostavlja se da je gradijent funkcije analitički poznat, što je u skladu sa uvjetima konvergencije metode. Alternativno, moguće je gradijent funkcije izvesti metodama numeričkog diferenciranja, no to bespotrebno komplicira izvedbu metode, te nije predmet ovog rada.

Relevatno je istaknuti da u praktične primjene, pored standardne formulacije, u literaturi se češće pojavljuju **i druge, ekvivalentne formulacije Nesterovljeve metode**. Među najpoznatijima su formulacije koje su predložili ***Sutskever et al.*** (2013) te ***Bengio et al.*** (2012), koje prilagođavaju originalnu metodu kako bi je učinili pogodnijom za primjenu u **treniranju rekurentnih i dubokih neuronskih mreža *(RNN & DNN)***.

2.1 Sutskeverova modifikacija (SNAG)

Trideset godina nakon Nesterovljeve publikacije, Sutskever i saradnika [1] objavljuju rad u kojem, dotada slaboprimjenutoj, Nesterovljevoj metodi pridaju značaj u kontekstu dubokog učenja *(eng. deep learning)*. Uz jednostavnu reformulaciju originalne metode, autori demonstriraju značajna ubrzanja u treniranju dubokih neuronskih mreža u odnosu na standardni gradijentni spust s momentumom.

Iako se ovaj rad često navodi kao ključni faktor popularizacije i opće primjene NAG metode u oblastima dubokog učenja, postoje slične ideje koje su nezavisno razvijene i primjenjene u drugim oblastima numeričke minimizacije.

Ključna ideja Sutskeverove modifikacije je ***'fazno'* pomjeranje metode za pola iteracije**. Umjesto redoslijeda *„gradijent → momentum“*, granica iteracije se pomjera tako da se dobije redoslijed *„momentum → gradijent“*. Rezultat je matematički ekvivalentna formulacija koja radi nad parametrima ϕ umjesto θ, a koja se pokazuje pogodnijom za potrebe *dubokog učenja* [1].

$$\phi_{t+1} = \phi_t + \mu v_t - \alpha \nabla f(\phi_t + \mu v_t) \tag{2.1}$$

Prednost ove formulacije je ta da parametri ϕ na kraju svake iteracije već su postavljeni na *look-ahead* poziciju. Sljedeća iteracija stoga automatski evaluira gradijent na ispravnom mjestu, te nema potrebe za **eksplicitnim održavanjem dva odvojena vektora parametara** kao u originalnoj formulaciji, ostajući **u skladu sa standardnim gradijentnim metodama**.

Implementacija metode sa ovom modifikacijom se može predstaviti sljedećim programskim kodom:

Program 2.2: Sutskeverova formulacija NAG metode

```
function nag_sutskever(x0, func, gradient;
                       learning_rate  = 0.001,
                       momentum_coeff = 0.9,
                       max_iter       = 5000,
                       eps            = 1e-7)

    v    = zeros(length(x0))
    x    = copy(x0)

    for i in 1:max_iter
        v_prev = copy(v)
        v      = momentum_coeff .* v .- learning_rate .*
           gradient(x .+ momentum_coeff .* v)
        x      = x .+ v
        if norm(grad_x) < eps
            break
        end
    end

    return (x=x, f=func(x))
end
```

Ova modifikacija je danas prisutna u mnogim softverskim okvirima za *duboko učenje*, kao što je **PyTorch**.

2.2 Bengiova modifikacija (BNAG)

Međutim, iako je pogodnija za primjenu u dubokom učenju, Sutskeverova formulacija nije nužno najbolji izbor za sve primjene Nesterovljeve metode, posebno u oblastima gdje je potrebna detaljna analiza stabilnosti i konvergencije, kao što su *rekurentne neuronske mreže*.

Bengiova formulacija [3]. nastaje kao prikladna alternativa Sutskeverovoj. Ključna prednost je što **ne zahtijeva računanje gradijenta na nestandardnoj poziciji** — dovoljno je samo modificirati koeficijente u proračunu spusta, što je znatno jednostavnija izmjena u kodnoj bazi koja već koristi standardni gradijentni spust sa momentumom

Razvijanjem Sutskeverove formulacije i sređivanjem članova, dobija se matematički ekvivalentan izraz:

$$\theta_{t+1} = \theta_t + \mu_{t-1}\mu_t\, b_t - (1+\mu_t)\,\alpha_t \nabla f(\theta_t) \tag{2.2}$$

gdje je:

$$b_{t+1} = \mu_t\, b_t - \alpha_t \nabla f(\theta_t) \tag{2.3}$$

Bengiova formulacija pokazuje da je *look-ahead* perspektiva samo jedan način gledanja te da se NAG ekvivalentno može posmatrati kao momentum s **korigiranim koeficijentima**, gdje metoda primjenjuje **veći efektivni korak gradijenata** i **manji efektivni korak momentuma**, što se predstavlja kao pogodnije za analizu stabilnosti treniranja *RNN*.

Program 2.3: Bengiova formulacija NAG metode

```
function nag_bengio(x0, func, gradient;
                    learning_rate  = 0.001,
                    momentum_coeff = 0.9,
                    max_iter       = 5000,
                    eps            = 1e-7)

    v    = zeros(length(x0))
    x    = copy(x0)

    for i in 1:max_iter
        grad = gradient(x)
        if norm(grad) < eps
            break
        end

        if i == 1
            x = x .- learning_rate .* grad
        else
            x = x .+ momentum_coeff^2 .* v .- (1 +
                momentum_coeff) .* learning_rate .* grad
            v = momentum_coeff .* v .- learning_rate .* grad
        end
    end

    return (x=x, f=func(x))
end
```

U implementaciji Bengiove formulacije uvodi se blago odstupanje od teorijske: proizvod $\mu_{t-1}\mu_t$ aproksimira se kao μ^2, što je ispravno jedino kada je $\mu = \text{const.}$ kroz sve iteracije. U teoretskoj formulaciji μ_t može biti raspoređen *(scheduled)* po iteracijama, što bi zahtijevalo eksplicitno praćenje μ_{t-1}.

Dodatno, vektor b_t iz teoretske formulacije predstavlja razliku parametara $\phi_{t+1} - \phi_t$, dok implementacija koristi v_t koji se ažurira standardnom momentum formulom $v = \mu v - \alpha \nabla f(\theta)$, što je ekvivalentno samo pod pretpostavkom $\mu = \text{const.}$ Za potrebe ovog rada, gdje je μ fiksan, ova aproksimacija ne uvodi grešku. Slučajevi u kojem je greška aproksimacije nezamerljiva su pretežno specifični, te u svrhu bolje vizualizacije nisu razmotreni u sklopu ovog rada, što naravno nije slučaj i u citiranoj literaturi. [3].

No, zbog uzete pretpostavke, posebnu pažnju zahtijeva prva iteracija ($i = 1$): Zbog $\mu_{t-1}\mu_t = 0 \cdot \mu_t = 0$ slijedi da $\mu_{t-1}\mu_t \neq \mu_t^2$

Rješenje ove anomalije preuzeto je od Jamesa Melvillea, autora paketa `mize` za algoritme numeričke minimizacije funkcija u programskom jeziku R [2], gdje se ažuriranje vektora brzine

preskače.

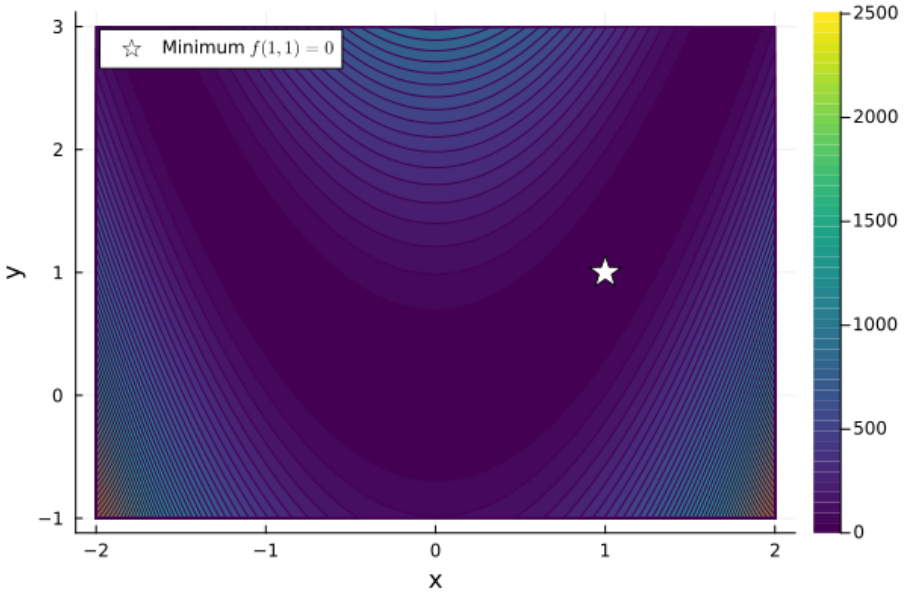
2.3 Primjer i vizualizacija primjene metode

2.3.1 Rosenbrockova funkcija

Rosenbrockova funkcija [4] *(poznata kao i 'banana funkcija')* je klasična testna funkcija u oblasti numeričke optimizacije, definisana kao:

$$f(x,y) = a(1-x)^2 + b(y-x^2)^2, \qquad a = 1,\ b = 100 \tag{2.4}$$

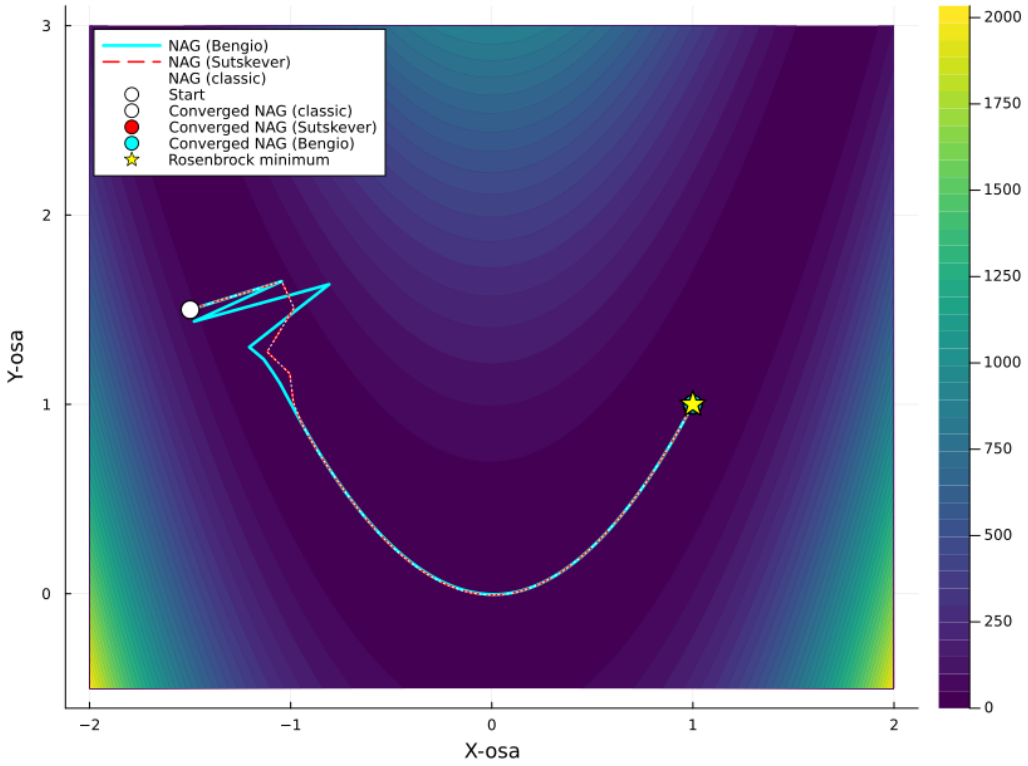
Globalni minimum se nalazi u tački $(x,y) = (1,1)$, gdje je $f(1,1) = 0$. Funkcija je poznata po svojoj **uskoj, zakrivljenoj dolini** koja vodi prema minimumu. Dolina je lako pronađena, ali konvergencija unutar nje je spora zbog zakrivljenosti i loše uvjetovanosti *(ill-conditioning)*. Upravo zbog toga se koristi kao standardni *benchmark* za testiranje metoda minimizacije, posebno gradijentnih metoda s momentumom.



Slika 2.1: Konturni grafik Rosenbrockove funkcije sa označenim globalnim minimumom $f(1,1) = 0$.

2.3.2 Rezultati minimizacije

Primjenom sve tri formulacije NAG metode na Rosenbrockovu funkciju sa startnom tačkom $(x_0, y_0) = (-1.5,\ 1.5)$ i parametrima $\alpha = 0.001$, $\mu = 0.9$, dobijeni su sljedeći rezultati:



Slika 2.2: Putanje konvergencije sve tri formulacije NAG metode na Rosenbrockovima funkciji. Klasična formulacija (bijela), Sutskeverova (crvena), Bengiova (cijan).

Sa Slike 2.2 jasno se vidi da klasična i Sutskeverova formulacija prate gotovo ekvivalentnu putanju konvergencije, dok Bengiova formulacija značajno odstupa u pojedinim iteracijama, no ipak uspijeva konvergirati u očekivanom broju koraka. Kao što se vidi iz tablice 2.1, Sutskeverova formulacija postiže konvergenciju znatno ranije od preostale dvije.

Tabela 2.1: Rezultati minimizacije Rosenbrockove funkcije Nesterovljevom metodom, $(x_0, y_0) = (-1.5,\ 1.5)$, $\alpha = 0.001$, $\mu = 0.9$.

Formulacija	**Broj iteracija**	x^*	f^*
NAG (klasična)	3576	(1.00000, 1.00000)	1.259×10^{-14}
NAG (Sutskever)	1153	(0.99744, 0.99488)	6.556×10^{-6}
NAG (Bengio)	3617	(1.00000, 1.00000)	1.247×10^{-14}

* * *

U ovom poglavlju prikazana je implementacija Nesterovljeve ubrzanog gradijentnog metoda razvijenog u tri općeprimjenute formulacije, klasičnoj, Sutskeverovoj i Bengiovoj, te je svaka demonstrirana na Rosenbrockovima funkciji kao standardnom *benchmark*-u. Implementacije su u skladu sa teoretskim formulacijama iz prethodnog poglavlja, uz eksplicitno naglašene aproksimacije i ograničenja tamo gdje postoje. U narednim poglavljima slijedi diskusija o praktičnoj primjene metode, njenim prednostima i nedostacima, te preporuke za daljnja istraživanja i primjene u različitim oblastima numeričke optimizacije.

Poglavlje 3

Primjene algoritma u praksi

Teoretska analiza i implementacija Nesterovljeve metode iz prethodnih poglavlja postavlja osnovu za razumijevanje njene primjene u realnim problemima. U ovom poglavlju razmotriti ćemo oblasti u kojima NAG metoda pronalazi direktnu i dokumentovanu primjenu, naročito u treniranje dubokih neuronskih mreža te obrada signala i kompresiranog opažanja.

3.1 Primjene metode u dubokom učenju

Najširа i najvažnija praktična primjena Nesterovljeve metode danas je u dubokom učenju, gdje služi kao temelj za treniranje kompleksnih arhitektura.

Rad Sutskevera i saradnika iz 2013. godine predstavlja prekretnicu jer je dokazao da se duboke neuronske mreže (DNN) i rekurentne neuronske mreže (RNN) mogu uspješno trenirati metodama prvog reda do nivoa performansi koji su ranije bili dostižni samo uz Hessian-Free (HF) optimizaciju. [1].

U suštini, treniranje duboke neuronske mreže je rješavanje ogromnog problema optimizacije milijardu parametara, gdje minimiziramo funkciju gubitka L, *koja je nelinearna, neglatka i često loše uvjetovana.* Matematički zapisano: $\theta^* = \arg\min_\theta \mathcal{L}(\theta)$

Metode prvog reda koriste samo informacije o gradijentu funkcije gubitka, što ih čini skalabilnim i efikasnim za velike modele. **Metode drugog reda** koriste informacije o Hessianu (drugom izvodu), što može biti preciznije ali je računski neizvodljivo za modele sa milijardama parametara.

Hessian-Free optimizacija [5] je metoda drugog reda koja koristi aproksimaciju Hessiana, ali i dalje zahtijeva značajne resurse, koja je do primjene Nesterovljeve metode predstavljala standard za treniranje dubokih modela.

Do 2013. godine, treniranje dubokih modela bilo je ograničeno na specifične arhitekture i probleme, gdje su se metode prvog reda smatrale nedovoljno stabilnim za veoma duboke mreže, posebno u slučaju rekurentnih neuronskih mreža koje pate od problema nestajućih i eksplodirajućih gradijenata. Smatralo se da je za efikasno učenje u takvim modelima neophodno

koristiti metode drugog reda koje uzimaju u obzir zakrivljenost funkcije gubitka.

Rad Sutskevera i saradnika pokazao je da pažljiv izbor parametara, incijalizacije i rasporedu učenja *(learning rate schedule)* metode prvog reda, posebno gradijentni spust sa momentumom i Nesterovljevim ubrzanjem, mogu postići performanse uporedive sa Hessian-Free optimizacijom, uz znatno manju računsku složenost i veću skalabilnost. Ovo je predstavljao ogroman iskorak u polju *dubokog učenja*, gdje su modeli postali jednostavniji za implementaciju, efikasni na GPU arhitetkurama i lako skalabilni na milione i milijarde parametara funkcije.

3.1.1 Primjena u modernim softverskim okvirima

Kao što je spomenuto u Poglavlju 2.1. Metoda je implementirana unutar **PyTorch framework-a**, gdje se čak aktivira jednom izmjenom u pozivu optimizatora:

Program 3.1: Aktivacija NAG-a u PyTorch-u

```
import torch.optim as optim

optimizer = optim.SGD(
    model.parameters(),
    lr=0.01,
    momentum=0.9,
    nesterov=True       # NAG umjesto klasicnog momentuma
)
```

Isto važi i za **Keras/TensorFlow okvire**:

Program 3.2: Aktivacija NAG-a u Keras-u

```
from tensorflow.keras.optimizers import SGD

optimizer = SGD(learning_rate=0.01, momentum=0.9, nesterov=
   True)
```

3.1.2 Adam i NAdam optimizatori

Razvoj optimizacionih metoda u dubokom učenju doveo je do potrebe za algoritmima koji istovremeno obezbjeđuju stabilnost momentuma i adaptivno podešavanje brzine učenja *(learning rate-a)*. U tu svrhu, istraživači **Kingma i Ba** su razvili **Adam** (Adaptive Moment Estimation) optimizator [6], koji je **T. Dozat** proširio sa **NAdam** [7], koja integriše ubrzanje Nesterovljeve metode u Adam-ov okvir.

Obje metode pripadaju metodama prvog reda i zasnovane su na procjeni statističkih momenata gradijenta. Neka je $g_t = \nabla \mathcal{L}(\theta_t)$ gradijent funkcije gubitka u trenutku t. Adam održava dvije eksponencijalne pokretne sredine:

$$m_t = \beta_1 m_{t-1} + (1-\beta_1) g_t \tag{3.1}$$

$$v_t = \beta_2 v_{t-1} + (1-\beta_2) g_t^2 \tag{3.2}$$

gdje je:

- m_t procjena prvog momenta (srednja vrijednost gradijenta),
- v_t procjena drugog momenta (srednja vrijednost kvadrata gradijenta),
- $\beta_1, \beta_2 \in (0,1)$ faktori zaborava.

Budući da su m_t i v_t inicijalno pristrasne procjene, uvodi se korekcija pristrasnosti:

$$\hat{m}_t = \frac{m_t}{1-\beta_1^t}, \qquad \hat{v}_t = \frac{v_t}{1-\beta_2^t}. \tag{3.3}$$

$$\theta_{t+1} = \theta_t - \eta \frac{\hat{m}_t}{\sqrt{\hat{v}_t} + \varepsilon}, \tag{3.4}$$

gdje je η brzina učenja *(learning rate)*, a ε mala konstanta uvedena za numeričku stabilizaciju korekcije. Adam se može interpretirati kao kombinacija momentuma (kroz prvi moment) i adaptivnog skaliranja koraka (kroz drugi moment), čime se omogućava automatsko prilagođavanje brzine učenja za svaki parametar pojedinačno.

Nadam (Nesterov-accelerated Adam) predstavlja modifikaciju Adam algoritma u kojoj se standardni momentum član zamjenjuje Nesterovljevim *look-ahead* principom. Ideja je da se u ažuriranju parametara koristi prediktivna korekcija zasnovana na procijenjenom budućem položaju u prostoru parametara. Ažuriranje kod Nadam optimizatora može se zapisati kao:

$$\theta_{t+1} = \theta_t - \eta \frac{\beta_1 \hat{m}_t + \frac{(1-\beta_1)}{1-\beta_1^t} g_t}{\sqrt{\hat{v}_t} + \varepsilon}. \tag{3.5}$$

Na taj način Nadam zadržava adaptivnu prirodu Adam optimizatora, ali dodatno uvodi Nesterovljevo ubrzanje, što može dovesti do brže konvergencije, posebno u slučajevima sa kompleksnom (nelinearnom) dinamikom gradijenata.

3.2 FISTA - Primjena algoritma u obradi signala i kompresiranog opažanja

Dalje, van oblasti dubokog učenja, najznačajnija izvedba NAG metode je zasigurno **FISTA** (*Fast Iterative Shrinkage-Thresholding Algorithm*), koju su razvili **Beck i Teboulle** 2009. godine [8].

3.2.1 Motivacija i problem

FISTA adresira klasu *linearnih inverznih problema* koji se tipično javljaju u obradi signala i slika. U tim problemima posmatrani signal $b \in \mathbb{R}^m$ nastaje kao rezultat sistema linearnih mjerenja

$$b = Ax^* + \varepsilon, \tag{3.6}$$

gdje je:

- $x^* \in \mathbb{R}^n$ nepoznati originalni signal,
- $A \in \mathbb{R}^{m \times n}$ matrica mjerenja (npr. operator zamućenja, projekcioni operator u CT-u),
- ε šum mjerenja.

Cilj inverznog problema jeste rekonstrukcija x^* iz mjerenja b. Međutim, u praksi je matrica A često loše uslovljena ili je problem nedovoljno određen ($m < n$), što znači da direktna inverzija nije moguća ili je numerički nestabilna. Ovakvi problemi nazivaju se *ill-conditioned*, tj. loše uslovljeni problemi.

Da bi se obezbijedila stabilnost rješenja, uvodi se regularizacija, čime se problem formuliše kao optimizacija:

$$\min_{x \in \mathbb{R}^n} F(x) = f(x) + g(x), \tag{3.7}$$

gdje:

- $f(x)$ predstavlja mjeru slaganja sa podacima (data fidelity term), najčešće oblika

$$f(x) = \frac{1}{2}\|Ax - b\|_2^2,$$

- $g(x)$ predstavlja član za regulaciju koji enkodira dotadašnje znanje o strukturi rješenja (npr. rijetkost, glatkoću ili ograničenja).

Tipičan primjer je ℓ_1 regularizacija

$$g(x) = \lambda\|x\|_1,$$

koja promoviše rijetka rješenja i igra ključnu ulogu u kompresiranom opažanju i LASSO regresiji. gdje je f glatka konveksna funkcija (npr. ℓ_2 greška rekonstrukcije), a g neravna konveksna regularizacija (npr. ℓ_1 norma za rijetke signale).

Prethodnik FISTA-e, algoritam ISTA, rješavao je ovaj problem ali s konvergencijom reda $O(1/k)$, što ga čini nepraktičnim za veće probleme.

3.2.2 FISTA kao primjena NAG ubrzanja

Ključna ideja u radu Becka i Teboullea jeste direktna primjena Nesterovljevog principa ubrzanja na ISTA algoritam. FISTA zadržava računsku jednostavnost ISTA-e, ali postiže globalnu stopu konvergencije koja je teorijski i praktično značajno bolja. Konkretno, stopa konvergencije FISTA-e je $O(1/k^2)$, dok je kod algoritama-prethodnika poput ISTA, CGDA i SLA stopa konvergencije $O(1/k)$.

Algoritam se može prikazati sljedećim koracima:

$$x_k = \text{prox}_{\alpha g}(y_k - \alpha \nabla f(y_k)) \tag{3.8}$$

$$t_{k+1} = \frac{1 + \sqrt{1 + 4t_k^2}}{2} \tag{3.9}$$

$$y_{k+1} = x_k + \frac{t_k - 1}{t_{k+1}} (x_k - x_{k-1}) \tag{3.10}$$

gdje $\text{prox}_{\alpha g}$ označava aproksimaciju regulatizatora g, a t_k je koeficijent ubrzanja analogan Nesterovljevom momentumu. **pravo ovaj član donosi $O(1/k^2)$ ubrzanje u odnosu na ISTA-u.**

3.2.3 Primjene FISTA-e

U praksi, FISTA pronalazi neke od vrlo ključnih oblasti i primjena, kao što su:

LASSO regresija, tip linearne regresije gdje se traži rješenje koje je rijetko i stabilno u prisustvu šuma.

Kompresirano opažanje (*compressed sensing*), gdje se rekonstruišu rijetki signali iz ograničenog broja linearnih mjerenja, pri čemu ℓ_1 minimizacija predstavlja efikasnu konveksnu relaksaciju problema.

Rekonstrukcija CT slika, u smislu formulacija problema tomografske rekonstrukcije kao inverznog problema, gdje FISTA omogućava bržu i stabilniju konvergenciju u odnosu na standardne metode.

Restauracija slika, u smislu uklanjanja šuma i zamućenja, što je baš kao primjer opisan u originalnom radu Becka i Teboullea [8].

* * *

U ovom poglavlju prikazane su ključne oblasti praktične primjene Nesterovljeve metode. U dubokom učenju, NAG je odigrao važnu ulogu u tranziciji ka skalabilnim metodama prvog reda, što je omogućilo efikasno treniranje dubokih neuronskih mreža velikih dimenzija.

U oblasti obrade signala i kompresiranog opažanja, Nesterovljevo ubrzanje čini teorijsku i algoritamsku osnovu FISTA metode, koja danas predstavlja standard u rješavanju inverznih problema.

U narednom poglavlju slijedi zaključna analiza rezultata rada, razmatranje ograničenja metode te prepoznavanje i diskusija o potencijalnim pravcima daljneg istraživanja.

Poglavlje 4

Zaključak i diskusija

Cilj ovog rada bio je sistematično predstaviti Nesterovljevu metodu ubrzanog gradijenta, od teorijskih osnova, kroz implementaciju, do praktičnih primjena. U nastavku ćemo rezimirati ključne zaključaka i diskutirati perfomans metode, relativan u odnosu na srodne izbore, te dotaći se otvorenih pitanja, mogućih pravaca daljnjeg istraživanja unutar domene teme.

4.1 Rezime pređenih tema

Nesterovljeva metoda ubrzanog gradijentnog predstavlja jedan od teoretskih najznačajnijih doprinosa u oblasti numeričke minimizacije i optimizacije. Nesterov je 1983. godine dokazao da je stopa konvergencije $O(1/k^2)$ teoretski optimalna za metode prvog reda na klasi glatkih konveksnih funkcija [9], i da njegova metoda tu granicu dostiže. Ovaj rezultat nije prevaziđen u narednih četrdeset godina.

U radu su implementirane i analizirane još dodatne dvije formulacije metode - Sutskeverova i Bengiova, izvedene iz praktičnih potreba. Eksperimentalni rezultati na Rosenbrockavoj funkciji, prikazani u Tablici 2.1, potvrđuju da su sve tri formulacije matematički ekvivalentne u smislu pronalaska minimuma, ali se razlikuju u brzini konvergencije i numeričkim karakteristikama. Sutskeverova formulacija pokazala je najbrže dostizanje kriterija zaustavljanja, dok je Bengiova bila najstabilnija u smislu oscilacija oko minimuma.

Vremenom, praktična primjena metode dostigla je visoku rasprostranjenost u raznim oblastima numeričke minimizacije, gdje smo se u ovom radu fokusirali na dvije najveće direktne kontribucije metode: dubokog učenja, gdje je rad Sutskevera i koautora [1] označio prekretnicu u treniranju dubokih i rekurentnih neuronskih mreža, te obrade signala, gdje Nesterovljevo ubrzanje čini teorijsku osnovu FISTA algoritma [8] i njegovih primjena u CT rekonstrukciji i kompresiranom opažanju.

4.2 Diskusija performansi i ograničenja

Ipak, pored opisanih snažnih odlika metode, NAG u praksi dolazi s nekoliko značajnih ograničenja koja je potrebno razmotriti.

4.2.1 Osjetljivost na hiperparametre

NAG zahtijeva ručno podešavanje dvije ključne hiperparametre: stope učenja α i koeficijenta momentuma μ. Za razliku od adaptivnih metoda poput Adama, koje automatski skaliraju korake učenja za svaki parametar pojedinačno i adaptiraju se prema dinamici gradijenata tokom treniranja, NAG ne implementira nikakvo automatsko prilagođavanje ovih vrijednosti. To znači da korisnik mora pažljivo odabrati α i μ prije početka optimizacije, a taj izbor može značajno uticati na performanse algoritma.

U eksperimentima prikazanim u ovom radu, fiksni parametri $\alpha = 0.001$ i $\mu = 0.9$ pokazali su se zadovoljavajućim za Rosenbrockovu funkciju, međutim ovi parametri nisu prenosivi. Problem sa drugačijom geometrijom funkcije gubitka generalno zahtijeva ponovnu kalibraciju.

Loš odabir ovih parametara može rezultirati divergencijom ili ekstremno sporom konvergencijom. Na ovo su Sutskever i saradnici strogo ukazivali i u svom inicijalnom osvrtu.

4.2.2 Ograničenje na nekonveksnim funkcijama

Nesterovljeve garancije konvergencije $O(1/k^2)$ vrijede isključivo za glatke konveksne funkcije. Funkcije gubitka u modernom dubokom učenju **nisu konveksne, sadrže sedlaste tačke, ravne regije i lokalne minimume**. U tim uvjetima, NAG nema formalnu garanciju konvergencije, a ubrzanje u odnosu na standardni gradijentni spust nije teorijski zagarantovano. Empirijski uspjeh metode u dubokom učenju često je više zasnovan na čistim heurističkim, a ne teoretskim osnovama [1].

4.2.3 Poređenje sa Adamom

U savremenom dubokom učenju, Adam optimizator [6] i njegovi izvodi (AdamW, Nadam) dominiraju u praktičnoj upotrebi, a NAG se rjeđe koristi kao samostalan optimizator. Razlog leži u adaptivnoj prirodi Adama, automatskim podešavanjem efektivne stope učenja po parametru, Adam je otporniji na loš odabir hiperparametara i brže konvergira u ranim iteracijama. Međutim, istraživanja pokazuju da dobro podešen NAG može postići komparabilne ili bolje rezultate od Adama, posebno gdje je moguće alocirati veću računsku moć za podešavanje hiperparametara [1].

Izbor između NAG-a i Adam/Nadam optimizatora u praksi svodi se na sljedeće: NAG je teorijski opravdan i razumni izbor za konveksne i dobro uvjetovane probleme, dok Adam i Nadam dominiraju u dubokom učenju gdje je adaptivnost važnija od teoretskih garancija.

4.3 Pravci daljnjeg istraživanja

Na osnovu analize provedene u radu, mogu se identificirati nekoliko potencijalnih pravaca za daljnje istraživanje i razvoj.

Adaptivno raspoređivanje momentuma (*momentum scheduling*), mjesto fiksnog μ, dinamičko povećanje koeficijenta momentuma tokom iteracija moglo bi poboljšati stabilnost konvergencije, naročito u ranim fazama optimizacije kada su gradijenati zašumljeni i nepouzdani.

Primjena FISTA-e u medicinskom snimanju, kako u zdravstvenom sektoru, CT rekonstrukcija i MRI kompresija postaju sve zahtjevniji s povećanjem rezolucije slike, primjena FISTA i varijanti algoritma predstavljaju obećavajući pravac istraživanja u kojima brzina konvergencije bi znatno uticalo na klinički tok rada.

Veza sa kontinuiranim dinamičkim sistemima — novija istraživanja pokazuju da se NAG može interpretirati kao diskretizacija određene obične diferencijalne jednačine drugog reda [10], što otvara mogućnost analize stabilnosti i dizajna novih optimizatora kroz teoriju dinamičkih sistema.

4.4 Zaključak

Nesterovljeva metoda ubrzanog gradijentnog ostaje jedan od temelja moderne numeričke minimizacije. Unatoč jednostavnosti koncepta nad kojim se oslanja, dovoljna je da podstakne razvoj cijele porodice algoritama opšteprimjene, Sutskeverova i Bengiova formulacija omogućile su primjenu u dubokom učenju, FISTA je prenijela Nesterovljevo ubrzanje u oblast obrade signala, a putem Nadam-a je čak pronašao svrhu i u najpopularnijem modernom optimizatoru.

Razumijevanje njene strukture i ograničenja predstavlja solidnu osnovu za praćenje i doprinos aktuelnim istraživanjima u oblasti optimizacije i mašinskog učenja. Činjenica da je metoda, objavljena 1980. godine, i danas prisutna u vodećim okvirima za duboko učenje i ostvaruje nivo konvergencije koji još uvijek nije značajno prevažiđen, svjedoči o njenoj fundamentalnoj važnosti kroz vremena.

Razumijevanje strukture metode, kao i ograničenja predstavlja solidnu osnovu za praćenje i doprinos aktuelnim istraživanjima u oblasti numeričke minimizacije, mašinskog učenja, te čak i obrade signala. U svakom od tih slučajeva, suštinska ideja ostaje ista, umjesto da reaguje na gradijent tamo gdje se trenutno nalazi, metoda *'osjeti'* budući položaj i ispravlja kurs unaprijed. Ta jednostavna modifikacija, kako je pokazano u ovom radu, ima veoma duboke posljedice u eksponencijalnom razvoju algoritama optimizacije i njihovoj primjeni u savremenim problemima.

Literatura

[1] Sutskever, I., Martens, J., Dahl, G., Hinton, G., "On the importance of initialization and momentum in deep learning", in International conference on machine learning. pmlr, 2013, str. 1139–1147.

[2] Melville, J., dostupno na: https://jlmelville.github.io/mize/nesterov.html Dec 2016.

[3] Bengio, Y., "Practical recommendations for gradient-based training of deep architectures", in Neural networks: Tricks of the trade: Second edition. Springer, 2012, str. 437–478.

[4] Rosenbrock, H., "An automatic method for finding the greatest or least value of a function", The computer journal, Vol. 3, No. 3, 1960, str. 175–184.

[5] Martens, J. *et al.*, "Deep learning via hessian-free optimization.", in Icml, Vol. 27, 2010, str. 735–742.

[6] Kingma, D. P., Ba, J., "Adam: A method for stochastic optimization", arXiv preprint arXiv:1412.6980, 2014.

[7] Dozat, T., "Incorporating nesterov momentum into adam", 2016.

[8] Beck, A., Teboulle, M., "A fast iterative shrinkage-thresholding algorithm for linear inverse problems", SIAM journal on imaging sciences, Vol. 2, No. 1, 2009, str. 183–202.

[9] Nesterov, Y., "A method for solving the convex programming problem with convergence rate o (1/k2)", in Dokl akad nauk Sssr, Vol. 269, 1983, str. 543.

[10] Su, W., Boyd, S., Candès, E. J., "A differential equation for modeling Nesterov's accelerated gradient method", Journal of Machine Learning Research, Vol. 17, No. 153, 2016, str. 1–43.