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?, kao I njegovo izvođenje?. 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: predložak, LaTeX, ETF

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?, as well as how is the algorithm derived?. 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: template, LaTeX, ETF

Sadržaj

Popis slika

Popis tabela

Poglavlje 1

Uvod

U skladu sa dobrom istraživačkom praksom, uvodno poglavlje rada prvog ciklusa studija bi trebao sadržavati bar sljedeće elemente:

- obrazloženje teme,
- opis strukture rada.

U narednom tekstu će detaljnije biti obrazložena svaka od tačaka.

1.1 Obrazloženje teme

U ovoj sekciji autor je dužan da obrazloži koja tema ili problem će biti analizirani ili istraživani, te zbog čega je upravo ova tema pogodna i bitna za istraživanje. Pohvalno je napraviti pregled literature sa odgovarajućim referenciranjem na istu.

1.2 Struktura rada

U ovoj sekciji je najbolje dati po jedan paragraf o svakom poglavlju iz rada. Potencirajte koji su glavni doprinosi svakog poglavlja, te kako su poglavlja medjusobno povezana.

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 opcionalne 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 koautori [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 formulacija metode (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 [2]. 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. [2].

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 [3], gdje se ažuriranje vektora brzine preskače.

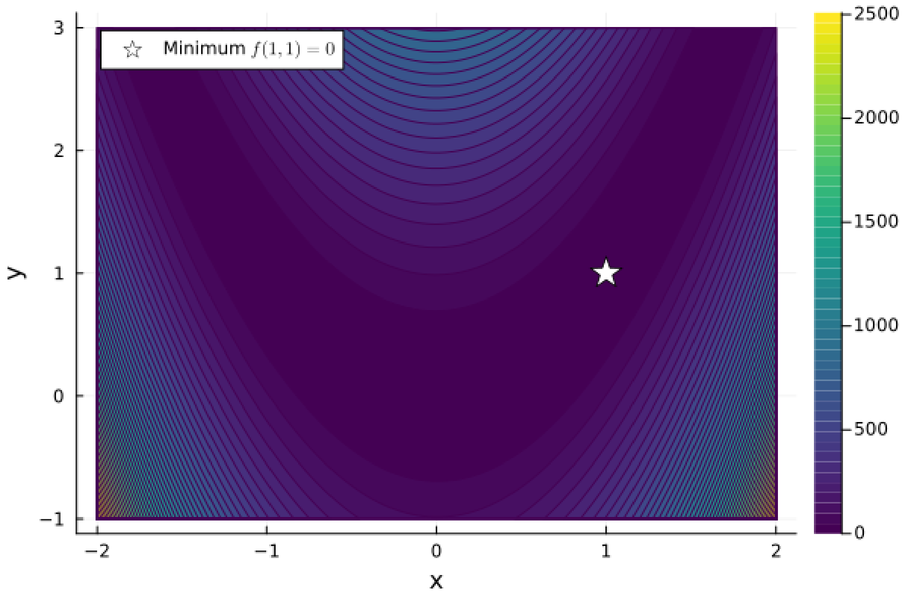
2.3 Primjer i vizualizacija primjene metode

2.3.1 Rosenbrockova funkcija

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

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

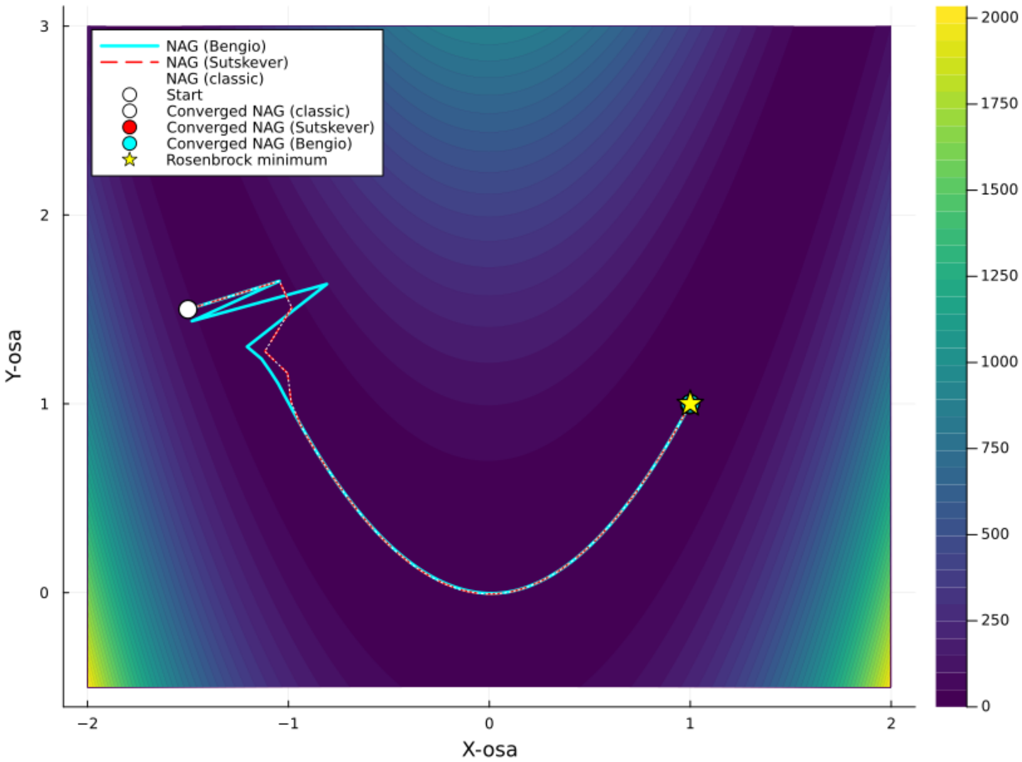
Globalni minimum se nalazi u tački $(x,y) = (1,1)$, gdje je $f(1,1) = 0$. Funkcija je poznata po svojoj **uskoj, zakrivljenoj dolini** koja vodi prema minimumu. Dolina je lako pronađena, ali konvergencija unutar nje je spora zbog zakrivljenosti i loše uvjetovanosti *(ill-conditioning)*. Upravo zbog toga 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 narednom poglavlju slijedi komparativna analiza dobijenih rezultata, diskusija prednosti i nedostataka svakog pristupa, te poređenje sa rezultatima iz relevantne literature.

Poglavlje 3

Primjene algoritma u praksi

Istraživanje se izlaže organizirano, koncizno i konzistentno kroz dva ili više odvojenih poglavlja, sa pregledom teoretskih osnova na kojima su bazirana i kraćom diskusijom dobijenih rezultata. Ono što je izuzetno važno jeste da se dobiveni rezultati istraživanja konstantno objektivno porede sa postojećim rezultatima u literaturi ili oblasti istraživanja, te sistematično ukazuje na prednosti i nedostatke autorskog pristupa. Izostanak komparacije rezultata istraživanja dobivenih od strane autora sa konkurentnim algoritmima, metodama i pristupima pokazuje nepostojanje akademske zrelosti, nedovoljnu posvećenost u istraživanju odgovarajuće naučne oblasti i vrlo često ukazuje na loš kvalitet disertacije/rada.

U radu se za formiranje poglavlja koriste sekcije, podsekcije, podpodsekcije, paragrafi i podparagrafi kao u primjerima koji slijedi.

3.1 Primjer sekcije

Ovo je primjer sekcije. Ovo je primjer sekcije. Ovo je primjer sekcije. Ovo je primjer sekcije. Ovo je primjer sekcije. Ovo je primjer sekcije. Ovo je primjer sekcije. Ovo je primjer sekcije. Ovo je primjer sekcije. Ovo je primjer sekcije. Ovo je primjer sekcije. Ovo je primjer sekcije. Ovo je primjer sekcije. Ovo je primjer sekcije. Ovo je primjer sekcije. Ovo je primjer sekcije. Ovo je primjer sekcije.

3.1.1 Primjer podsekcije

Ovo je primjer podsekcije. Ovo je primjer podsekcije. Ovo je primjer podsekcije. Ovo je primjer podsekcije. Ovo je primjer podsekcije. Ovo je primjer podsekcije. Ovo je primjer podsekcije. Ovo je primjer podsekcije. Ovo je primjer podsekcije. Ovo je primjer podsekcije. Ovo je primjer podsekcije. Ovo je primjer podsekcije. Ovo je primjer podsekcije. Ovo je primjer podsekcije. Ovo je primjer podsekcije.

Primjer podpodsekcije

Ovo je primjer podpodsekcije. Ovo je primjer podpodsekcije. Ovo je primjer podpodsekcije. Ovo je primjer podpodsekcije. Ovo je primjer podpodsekcije. Ovo je primjer podpodsekcije. Ovo je primjer podpodsekcije. Ovo je primjer podpodsekcije. Ovo je primjer podpodsekcije. Ovo je primjer podpodsekcije. Ovo je primjer podpodsekcije. Ovo je primjer podpodsekcije. Ovo je primjer podpodsekcije.

Primjer paragrafa Ovo je primjer paragrafa. Ovo je primjer paragrafa. Ovo je primjer paragrafa. Ovo je primjer paragrafa. Ovo je primjer paragrafa. Ovo je primjer paragrafa. Ovo je primjer paragrafa. Ovo je primjer paragrafa. Ovo je primjer paragrafa. Ovo je primjer paragrafa. Ovo je primjer paragrafa. Ovo je primjer paragrafa. Ovo je primjer paragrafa. Ovo je primjer paragrafa.

Primjer podparagrafa Ovo je primjer podparagrafa. Ovo je primjer podparagrafa. Ovo je primjer podparagrafa. Ovo je primjer podparagrafa. Ovo je primjer podparagrafa. Ovo je primjer podparagrafa. Ovo je primjer podparagrafa. Ovo je primjer podparagrafa. Ovo je primjer podparagrafa. Ovo je primjer podparagrafa. Ovo je primjer podparagrafa. Ovo je primjer podparagrafa. Ovo je primjer podparagrafa. Ovo je primjer podparagrafa.

*　　*　　*

Na kraju svakog poglavlja, najbolje je dati jedan kraći zaključak koji ukratko objedinjuje sve najvažnije zaključke iz tog poglavlja. Taj kraći zaključak treba da služi kao poveznica između poglavlja koje se upravo završilo, i narednog poglavlja koje tek treba da počne. Ovaj zaključak je poželjno odvojiti bilo kao odvojenu podsekciju poglavlja nazvanu "Zaključak", bilo kao jednostavno izdvojeni dio teksta razmaknut zvjezdicama.

Kratak primjer zaključka za ovo poglavlje: U ovom poglavlju je pokazano kako se formiraju centralna poglavlja u radu/disertaciji. U nastavku će biti pokazano kako se piše konačan zaključak, te dati određeni tehnički podaci oko formatiranja teksta, slika i formula.

Zaključak i diskusija

Preporučuje se da se poglavlje "Uvod" i "Zaključak", te odgovarajuće sekcije i podsekcije ne numerišu. Ovo poglavlje bi trebalo na izvjestan način objediniti sve "kraće" zaključke date na kraju pojedinih poglavlja.

Ostvareni ciljevi završnog rada

U ovoj sekciji je potrebno dati jasan sumarni pregled obavljenih istraživanja i dobijenih rezultata. Rezultati trebaju biti struktuirani i prikazani prema okvirima i ciljevima postavljenim u uvodnom poglavlju. Potrebno je i provesti poredjenja dobivenih rezultata sa literaturom navedenom u uvodnom poglavlju, te dati diskusiju kako se dobijeni rezultati uklapaju, potvrdjuju, nadopunjuju ili su kontradiktorni onim koji su prikazani u uvodnom poglavlju.

Prilozi

Prilog A

Korištenje funkcija u Tex-u

Sadržaji koji se mogu uključiti u Priloge su: izvođenje jednačina i formula, detalji važnijih softverskih programa, razne tabele i dijagrami, karkateristike i performanse ili opisi opreme i komponenti koje su korištene u disertaciji/radu. Mogu se takodjer uključiti konstrukcioni crteži ili električne sheme.

U ovom prilogu prikazane su neke od funkcije koje se mogu koristiti prilikom oblikovanja rada i prikaza rezultata istraživanja korištenjem LaTeXa.

A.1 Matematički izraz

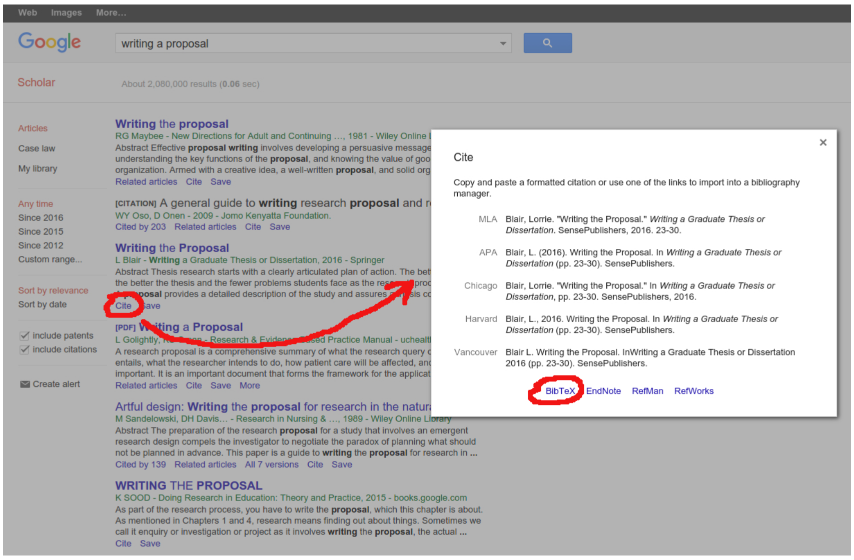
Primjer matematičke formule prikazan je izrazom

$$T : \mathbf{x}_B \mapsto \mathbf{x}_A \Leftrightarrow T(\mathbf{x}_B) = \mathbf{x}_A. \tag{A.1}$$

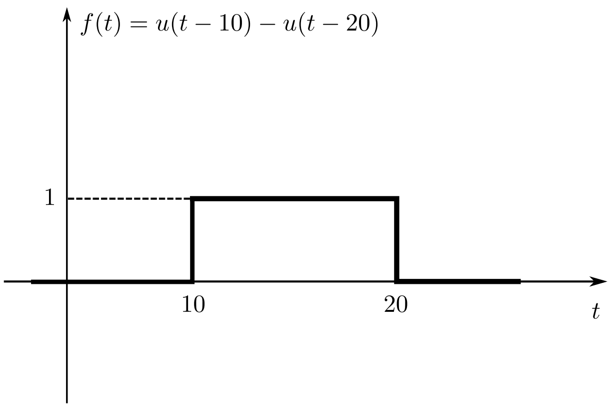
Matematičke relacije se u LaTeXrazvojnom okruženju automatski numeriraju. Međutim, da bi se pojedina relacija (slika, tabela) referencirala u tekstu, potrebno je da se svakoj relaciji (slici, tabeli) dodijeli pogodna labela npr. `\label{MojaRelacija}`, a potom referencira u .tex fajlu sa `\ref{MojaRelacija}`. Na taj način će se LaTeXpobrinuti za odgovarajuće kros-referenciranje.

A.2 Slika

Slika A.1 služi kao primjer uključivanja slike u tekst. Relacije, slike i tabele se automatski numeriraju u LaTeXu, i to sa oznakom brojpoglavlja.brojslike (npr. u Poglavlju 3 se numeriraju sa 3.1, 3.2, ... neovisno od toga u kojoj sekciji ili podsekciji se nalaze).



Slika A.1: Primjer naslova slike - uputstvo za traženje bibiografskih referenci na Google Scholar.



Slika A.2: Primjer dijagrama - veličina i tip fonta na slici bi trebao odgovarati veličini i tipu fonta u tekstu

A.3 Tabela

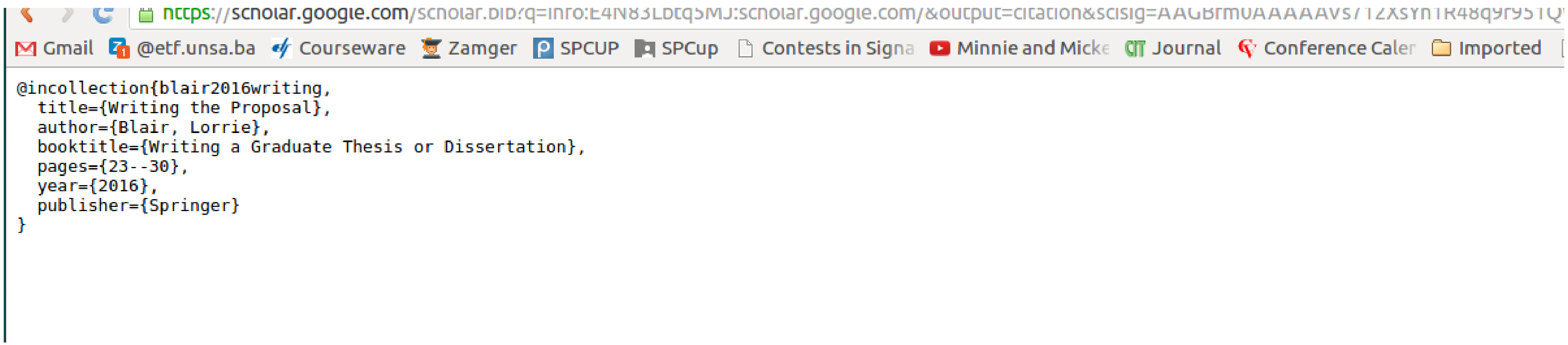
Formiranje tabele prikazano je na primjeru u Tabeli A.1. Za razliku od naslova slika, naslov tabela stoji iznad odgovarajućih tabela u tekstu.

Tabela A.1: Naslov tabele

Oznaka reda	Kolona 1	Kolona 2	Kolona 3
red 1	1	2	3
red 2	3	2	1
red 3	$E = mc^2$	2	3

A.4 *Landscape*

Postavljanja stranice u prikaz *landscape* prikazano je umetanjem izduzene Slike A.3 u *landscape* format papira.



Slika A.3: Primjer ekstrakcije bibliografskih stavki za kopiranje u .bib fajl, sa Google Scholara

A.5 Indeks pojmova i Popis oznaka

Ukoliko je u radu neophodno uvesti i indeks, odnosno popis oznaka, onda se to radi na sljedeći način. Prilikom definiranja indeksa koristi se `\index{ime}`. Npr. `\index{uključivanja slike}`.

Kod dodavanja pojmova u Popis oznaka u .tex fajlu se koristi `\nomenclature{simbol}{opis}`, npr. `\nomenclature{ETF}{Elektrotehnički fakultet}`. Generiranje indeksa i Popisa oznaka se pravi korištenjem naredbi `\makeindex` i `\makenomenclature` u preambli, odnosno `\printindex` i `\printnomenclature` na mjestu generiranja popisa. Osim toga, potrebno je i kompajlirati dokument sa MakeIndex.

A.6 Korištenje literature

Popis literature navodi se na kraju rada. Da bi uz LaTeXefikasno koristila literatura, potrebno je da se generira fajl sa bibliografskim jedinicama. Fajl `literatura.bib` je sastavni dio ovog rada, i može poslužiti kao primjer kako se pišu pojedine bibliografske jedinice. Svaki unos (referenca) sadrži labelu na tu referencu, putem koje se bilo gdje u radu može citirati npr. sa `\cite{Hajn01}` .

Dobar trik za popunjavanje bibliografskih unosa u .bib fajlu je korištenje Google Scholara `https://scholar.google.com/`. Osim što je baza naučnih radova, Google Scholar omogućava i kopiranje zapisa referenci na ispravan način. Podržani su svi najpopularniji formati citiranja (MLA, Chicago, Harvard itd.), kao što se vidi na Slici A.1. Osim toga, klikom na dugme "BibTeX", moguće je izabrati i zapis reference razumljive razvojnom okruženju LaTeX, a nakon toga je jednostavno kopirati u bibliografski fajl `literatura.bib` (vidjeti Sliku A.3).

A.7 Programski kodovi

Programski kodovi se LaTeXu navode korištenjem okruženja `lstlisting`. Primjer koda je dat ispod.

Program A.1: Primjer programa

```
// program u C++
#include <iostream>

int main ()
{
  std::cout << "Dobar␣Dan!␣";
  std::cout << "Prvi␣program␣u␣C++";
}
```

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] 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.

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

[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.

Indeks pojmova