Obrada rezultata merenja

Predrag Pejović

30.10.2022, 14:22

1 Uvod

Obrada rezultata merenja obuhvata numeričke postupke kojima se izračunava vrednost merene veličine kod posrednih merenja, ili se iz ponovljenih merenja dobija bolja procena merene veličine. Širokom dostupnošću digitalnih računara se oblast merenja promenila: obrada rezultata je postala jeftina i lako dostupna. Merenje se danas uglavnom svodi na senzor koji merenu veličinu pretvara u električni oblik, koji se potom digitalizuje i postaje broj koji digitalni računari izuzetno brzo i lako obrađuju. Stoga je oblast obrade rezultata merenja bitno dobila na značaju i proširila primenu, a time je uticala i na metode i svakodnevnu praksu merenja. Danas je multimetar digitalni uređaj, daje rezultat u obliku cifara na displeju. Sa druge strane, svi rezultati merenja se dokumentuju i/ili obrađuju na računaru. Da li je potreban čovek koji podatke sa multimetra rukom prepisuje u svesku, da bi ih kasnije ukucao na tastaturi računara radi dokumentovanja i/ili obrade? Tu je čovek usko grlo u sistemu i izvor grubih grešaka. Merenja su se promenila, mnogi poslovi u toj oblasti nestaju, neki novi nastaju, a ti nastajući poslovi zahtevaju veće obrazovanje, ali i mnogo manje napornog i dosadnog rada.

U ovom tekstu će prvo biti izložena obrada ponovljenih rezultata merenja fizičke veličine koja pretpostavljeno ima konstantnu vrednost. Tu će biti uvedena minimizacija kvadratne greške u optimizaciji rezultata merenja, kao i koncept standardne devijacije za kvantifikovanje rasipanja rezultata merenja. Beselova korekcija će biti prikazana kao optimalna procena varijanse rezultata merenja. Nakon toga, metod minimizacije kvadratne greške će biti primenjen u posrednim merenjima, na optimalnom fitovanju proporcionalne, linearne i polinomske međusobne zavisnosti dve veličine.

Dati prikaz obrade rezultata merenja je veoma elementaran i predstavlja samo uvod u oblast. Budućnost merenja je u digitalizaciji rezultata koje daje senzor i kasnijoj numeričkoj obradi, pa u ovoj oblasti treba očekivati veliki napredak, pre svega u primeni do sada razvijenih metoda.

2 Ponovljena merenja konstantne veličine

Prvi slučaj obrade rezultata merenja će biti merenje nepoznate fizičke veličine ξ za koju se po nekom osnovu smatra da je tokom merenja konstantna. Da bi obrada rezultata imala smisla, rezultati ponovljenih merenja veličine ξ treba da budu prirodno međusobno različiti, bez obzira na to što su uslovi merenja isti. Tipičan primer ovakvog procesa je merenje konstantnog napona analogno/digitalnim konvertorom, na primer onim koji se nalazi u digitalnom osciloskopu. Usled prisustva šuma dobiće se niz od n rezultata merenja

$$x_1,\, x_2,\, \ldots\, x_n \tag{1}$$

među kojima ima međusobno različitih. Rezultate merenja smo numerisali indeksom i, gde $i \in \{1, \ldots n\}$.

Greška svakog pojedinačnog merenja se **uvek** definiše kao razlika izmerene vrednosti x_i i tačne vrednosti ξ (od rezultata merenja se oduzima tačna vrednost)

$$\delta_i \triangleq x_i - \xi \tag{2}$$

gde stalno treba imati u vidu da je ξ nepoznata vrednost za koju je osnovano pretpostavljeno da je konstantna tokom procesa merenja.

Greške pojedinačnih merenja mogu biti i pozitivne i negativne, pa greška celog procesa merenja ne može biti karakterisana njihovim zbirom $\sum_{i=1}^{n} \delta_i$, pošto bi negativne greške potirale pozitivne. Moguća mera greške procesa merenja je zbir apsolutnih vrednosti grešaka pojedinačnih merenja $\sum_{i=1}^{n} |\delta_i|$ [1], ali za analizu i obradu podataka veliki problem predstavlja nediferencijabilnost funkcije apsolutna vrednost u nuli. Stoga se kao mera greške celog procesa merenja koristi zbir kvadrata pojedinačnih grešaka

$$\Delta(\xi) \triangleq \sum_{i=1}^{n} \delta_i^2 = \sum_{i=1}^{n} (x_i - \xi)^2 \tag{3}$$

pošto je kvadratna funkcija glatka i svuda diferencijabilna. Značaj diferencijabilnosti će uskoro postati jasan, a sada treba naglasiti da je ukupna greška procesa merenja $\Delta(\xi)$ zavisna od nepoznate merene vrednosti ξ, tačne vrednosti.

Posle pretpostavke da je merena veličina ξ konstantna tokom procesa merenja, uvešćemo i pretpostavku da su greške merenja slučajne, sa simetričnom gustinom raspodele, što znači da su greške iste apsolutne vrednosti, a suprotnog znaka, jednako verovatne. Pod tom pretpostavkom ima smisla stav da je tačna vrednost merene fizičke veličine ξ bliska vrednosti koja minimizuje ukupnu kvadratnu grešku procesa merenja (3). Stoga, nepoznatu merenu veličinu ξ procenjujemo (estimiramo) iz uslova da ta vrednost minimizuje ukupnu kvadratnu grešku procesa merenja, kada je

$$\frac{d\,\Delta(\xi)}{d\,\xi} = -2 \sum_{i=1}^{n} (x_i - \xi) = 0. \tag{4}$$

Kako je ξ konstanta, gornja suma se može razviti kao

$$\sum_{i=1}^{n} (x_i - \xi) = \sum_{i=1}^{n} x_i - n\,\xi = 0 \tag{5}$$

odakle je vrednost ξ koja minimizuje ukupnu kvadratnu grešku procesa merenja

$$\xi = \frac{1}{n} \sum_{i=1}^{n} x_i \tag{6}$$

pa je procenjena vrednost merene veličine ξ jednaka aritmetičkoj sredini [2] pojedinačnih merenja

$$\boxed{\xi_{opt} = \overline{x} \triangleq \frac{1}{n} \sum_{i=1}^{n} x_i.} \tag{7}$$

Indeks *opt* je izabran kako bi naglasili da je procenjena vrednost rezultata merenja posledica optimizacije po ξ, gde je kriterijum optimizacije minimum kvadratne greške procesa merenja. **Možemo da zaključimo da procenjena vrednost rezultata merenja $\xi = \xi_{opt}$ minimizuje ukupnu kvadratnu grešku procesa merenja $\Delta(\xi)$.**

Zainteresovanim studentima ovde valja napomenuti da aritmetička sredina nije jedini reprezent za „optimalnu vrednost“ rezultata serije ponovljenih merenja, sa različitim argumentima se uvode medijana [3] i mod [4].

3 Rasipanje rezultata merenja

Sledeći problem koji ćemo analizirati je karakterizacija rasipanja rezultata merenja za dati proces merenja. Kao mera ukupne greške u procesu merenja je usvojen zbir kvadrata pojedinačnih grešaka merenja, $\sum_{i=1}^{n} \delta_i^2$. Data mera ne karakteriše sam proces merenja pošto nužno raste sa rastom broja merenja n, pa se za dva puta više rezultata merenja očekuje približno dva puta veća vrednost zbira kvadrata pojedinačnih grešaka merenja. Stoga ima smisla ukupnu grešku merenja podeliti sa brojem merenja, $\frac{1}{n} \sum_{i=1}^{n} \delta_i^2$, čime se dobija vrednost koja karakteriše sam proces merenja. Ta vrednost ima određenu primenu i zove se **varijansa** [5] rezultata merenja. Problem sa ovakvom merom je fizička dimenzija [6] veličine koja karakteriše rasipanje rezultata merenja i koja je jednaka kvadratu dimenzije merene veličine. Kao primer, rasipanje rezultata procesa merenja dužine bi bilo izraženo u kvadratnim metrima, dok bi sam rezultat merenja bio izražen u metrima. Ova činjenica kvari intuitivnu predstavu rasipanja rezultata merenja, bilo bi zgodno da se rasipanje rezultata merenja izražava u istoj jedinici mere kao i sam rezultat merenja. Stoga se za karakterizaciju rasipanja rezultata merenja koristi kvadratni koren srednje kvadratne greške rezultata merenja $\sqrt{\frac{1}{n} \sum_{i=1}^{n} \delta_i^2}$, pa se u skladu sa tim i definiše **standardna devijacija** [7] kao

$$\sigma \triangleq \sqrt{\frac{1}{n} \sum_{i=1}^{n} (x_i - \xi)^2} \tag{8}$$

dok se pod pojmom **varijansa** podrazumeva kvadrat standardne devijacije.

Osnovni problem sa ovakvom definicijom standardne devijacije leži u činjenici da je tačna vrednost merene veličine uglavnom nepoznata, izuzetak je merenje etalona u postupku kalibracije mernog sredstva ili postupka merenja: tada je vrednost etalona ξ tačna vrednost.

Kako je moguće odrediti standardnu devijaciju postupka merenja ako je tačna vrednost nepoznata? Radi pojednostavljenja notacije računaćemo $n\,\sigma^2$ umesto σ prema

$$n\,\sigma^2 = \sum_{i=1}^{n} (x_i - \xi)^2 \,. \tag{9}$$

U cilju zamene ξ sa $\xi_{opt} = \overline{x}$, gornji izraz valja proširiti oduzimanjem i dodavanjem $\overline{x}$ prema

$$\sum_{i=1}^{n} (x_i - \xi)^2 = \sum_{i=1}^{n} (x_i - \overline{x} + \overline{x} - \xi)^2 \,. \tag{10}$$

Na ovom mestu je povoljno uvesti pojmove stvarna greška

$$a_i \triangleq x_i - \xi \tag{11}$$

što je isto što i δ_i, ali je notacija promenjena kako bi pratila standardne tekstove koji se ovom temom bave. Takođe, povoljno je uvesti i procenjene greške merenja (često ih nazivaju i „prividne“ greške merenja)

$$b_i \triangleq x_i - \overline{x}. \tag{12}$$

Valja naglasiti da su vrednosti stvarnih grešaka a_i nepoznate, jer nije poznata tačna vrednost merene veličine ξ, dok su vrednosti b_i poznate, pošto su poznate vrednosti rezultata merenja x_i za $i \in \{1, \ldots n\}$.

Uvođenjem notacije stvarnih i procenjenih grešaka u (10) dobija se

$$\sum_{i=1}^{n} (x_i - \xi)^2 = \sum_{i=1}^{n} a_i^2 = \sum_{i=1}^{n} (b_i + (\overline{x} - \xi))^2 = \sum_{i=1}^{n} \left(b_i^2 + 2\,(\overline{x} - \xi)\,b_i + (\overline{x} - \xi)^2\right) \tag{13}$$

što se razvija u

$$n\,\sigma^2 = \sum_{i=1}^{n} a_i^2 = \sum_{i=1}^{n} b_i^2 + 2\,(\overline{x} - \xi)\,\sum_{i=1}^{n} b_i + n\,(\overline{x} - \xi)^2 \tag{14}$$

nakon izvlačenja neindeksiranih veličina izvan sume. Ovde je povoljno to što je zbir procenjenih grešaka

$$\sum_{i=1}^{n} b_i = \sum_{i=1}^{n} (x_i - \overline{x}) = \sum_{i=1}^{n} x_i - n\,\overline{x} = \sum_{i=1}^{n} x_i - n\,\frac{1}{n}\,\sum_{i=1}^{n} x_i = \sum_{i=1}^{n} x_i - \sum_{i=1}^{n} x_i = 0 \tag{15}$$

jednak nuli, pa se (14) pojednostavljuje na

$$n\,\sigma^2 = \sum_{i=1}^{n} a_i^2 = \sum_{i=1}^{n} b_i^2 + n\,(\overline{x} - \xi)^2\,. \tag{16}$$

Sada je potrebno da se preko stvarnih grešaka izrazi $\overline{x} - \xi$

$$\overline{x} - \xi = \frac{1}{n}\,\sum_{i=1}^{n} x_i - \xi = \frac{1}{n}\,\sum_{i=1}^{n} x_i - \frac{1}{n}\,\sum_{i=1}^{n} \xi = \frac{1}{n}\,\sum_{i=1}^{n} (x_i - \xi) = \frac{1}{n}\,\sum_{i=1}^{n} a_i \tag{17}$$

što dovodi do

$$n\,\sigma^2 = \sum_{i=1}^{n} a_i^2 = \sum_{i=1}^{n} b_i^2 + n\,\frac{1}{n^2}\,\left(\sum_{i=1}^{n} a_i\right)^2 = \sum_{i=1}^{n} b_i^2 + \frac{1}{n}\,\left(\sum_{i=1}^{n} a_i\right)^2. \tag{18}$$

Na ovom mestu nastupa glavni deo izvođenja u kome se analizira član $\left(\sum_{i=1}^{n} a_i\right)^2$ iz (18) koji se razvija u sumu proizvoda

$$\left(\sum_{i=1}^{n} a_i\right)^2 = \left(\sum_{i=1}^{n} a_i\right)\,\left(\sum_{i=1}^{n} a_i\right) = \sum_{i=1}^{n} a_i^2 + \sum_{i=1,\,j=1,\,i\neq j}^{n,\,n} a_i\,a_j \tag{19}$$

odnosno

$$\begin{aligned}\left(\sum_{i=1}^{n} a_i\right)^2 \;=\; & \boxed{a_1^2} \;+\; a_1\,a_2 \;+\; \ldots \;+\; a_1\,a_n \;+ \\ & a_2\,a_1 \;+\; \boxed{a_2^2} \;+\; \ldots \;+\; a_2\,a_n \;+ \\ & \vdots \\ & a_n\,a_1 \;+\; a_n\,a_2 \;+\; \ldots \;+\; \boxed{a_n^2}.\end{aligned} \tag{20}$$

Uokvireni kvadratni članovi su uvek pozitivni i njihov zbir sa povećavanjem broja merenja n raste, dok unakrsni proizvodi, odnosno proizvodi članova sa različitim indeksima, jednako verovatno su manji ili veći od nule, pa teže da se međusobno potru. Stoga je zbir unakrsnih proizvoda približno jednak nuli

$$\sum_{i=1,\,j=1,\,i\neq j}^{n,\,n} a_i\,a_j \approx 0 \tag{21}$$

što je ključni korak u ovom izvođenju, a po svojoj suštini je aproksimacija. Iz ovoga sledi

$$\left(\sum_{i=1}^{n} a_i\right)^2 \approx \sum_{i=1}^{n} a_i^2. \tag{22}$$

Zamenom u (18) se dobija

$$\sum_{i=1}^{n} a_i^2 = \sum_{i=1}^{n} b_i^2 + \frac{1}{n} \sum_{i=1}^{n} a_i^2 \tag{23}$$

odakle je

$$\left(1 - \frac{1}{n}\right) \sum_{i=1}^{n} a_i^2 = \sum_{i=1}^{n} b_i^2 \tag{24}$$

i konačno

$$\sum_{i=1}^{n} a_i^2 = \frac{n}{n-1} \sum_{i=1}^{n} b_i^2 \tag{25}$$

što dovodi do

$$n\,\sigma^2 = \frac{n}{n-1} \sum_{i=1}^{n} b_i^2 \tag{26}$$

i konačno do

$$\sigma = \sqrt{\frac{1}{n-1} \sum_{i=1}^{n} b_i^2} = \sqrt{\frac{1}{n-1} \sum_{i=1}^{n} (x_i - \xi)^2}. \tag{27}$$

Prema datom izvođenju, kada se umesto tačne vrednosti merene veličine, koja je nepoznata, koristi njena procena data srednjom vrednošću rezultata merenja, standardna devijacija se računa iz

$$\boxed{\sigma = \sqrt{\frac{1}{n-1} \sum_{i=1}^{n} (x_i - \overline{x})^2}} \tag{28}$$

dok se u slučajevima kada se meri etalon i kada je tačna vrednost poznata, standardna devijacija postupka merenja se računa po definiciji, iz

$$\boxed{\sigma \triangleq \sqrt{\frac{1}{n} \sum_{i=1}^{n} (x_i - \xi)^2}.} \tag{29}$$

Prikazano izvođenje se naziva Beselova korekcija [8] i ona daje optimalnu procenu (*unbiased estimator*) varijanse rezultata ponovljenih merenja.

4 Fitovanje konstante proporcionalnosti

Pretpostavimo da imamo niz vrednosti nezavisno promenljive

$$x_1,\, x_2, \ldots\, x_n \tag{30}$$

za $i \in \{1, \ldots\, n\}$, i njemu odgovarajući niz vrednosti zavisno promenljive

$$y_1,\, y_2, \ldots\, y_n. \tag{31}$$

Pretpostavljena veza između nezavisno promenljive i zavisno promenljive je u sada razmatranom slučaju proporcionalnost

$$y = \alpha\, x \tag{32}$$

a konstanta proporcionalnosti α je nepoznata i treba je iz raspoloživih nizova vrednosti x_i i y_i odrediti. Ovaj postupak se naziva fitovanje [9]. Kada bi podaci i model zavisnosti bili idealni važilo bi

$$y_i = \alpha\, x_i \tag{33}$$

za svako $i \in \{1, \ldots n\}$, pa bi jedan par vrednosti x_i i y_i bio dovoljan da se odredi α. Međutim, realni podaci se u ovo ne uklapaju u potpunosti, već samo približno, pa bi svaki par vrednosti x_i i y_i određivao svoju konstantu proporcionalnosti $\alpha_i = y_i/x_i$, čime model gubi opštost. Stoga ćemo pretpostaviti jedinstvenu vrednost konstante proporcionalnosti α i definisati pojedinačne greške merenja i/ili modela kao

$$\delta_i \triangleq y_i - \alpha\, x_i. \tag{34}$$

Kako bi karakterisali grešku na nivou celog skupa rezultata, za $i \in \{1, \ldots n\}$, iz istih razloga kao i u slučaju ponovljenih merenja konstantne veličine, definišemo ukupnu grešku kao zbir kvadrata pojedinačnih grešaka

$$\Delta\left(\alpha\right) \triangleq \sum_{i=1}^{n} \delta_i^2 = \sum_{i=1}^{n} \left(y_i - \alpha\, x_i\right)^2. \tag{35}$$

Vrednost konstante proporcionalnosti, parametra α, se određuje postupkom optimizacije tako što se minimizuje ukupna kvadratna greška. U tom cilju se izvod ukupne kvadratne greške po parametru α

$$\frac{d\,\Delta\left(\alpha\right)}{d\,\alpha} = -2 \sum_{i=1}^{n} x_i\ \left(y_i - \alpha\, x_i\right) \tag{36}$$

izjednači sa nulom

$$\frac{d\,\Delta\left(\alpha\right)}{d\,\alpha} = 0 \tag{37}$$

odakle je

$$\sum_{i=1}^{n} x_i\, y_i - \alpha \sum_{i=1}^{n} x_i^2 = 0 \tag{38}$$

pa se za optimalnu vrednost parametra α dobija

$$\boxed{\alpha_{opt} = \frac{\sum_{i=1}^{n} x_i\, y_i}{\sum_{i=1}^{n} x_i^2}.} \tag{39}$$

Fizička dimenzija parametra α je jednaka odnosu fizičkih dimenzija promenljivih y i x, tako da je rezultat optimizacije saglasan sa dimenzionom analizom [6], što je jedan vid provere dobijenog rezultata.

Tipičan primer primene opisanog metoda je merenje električne otpornosti metodom ampermetra i voltmetra, gde ako nepoznata otpornost odgovara parametru α, vrednosti x_i odgovaraju struji, a vrednosti y_i naponu na nepoznatoj otpornosti.

Prvo pitanje za razmišljanje: ako se sa istim skupom podataka određuje električna provodnost, a ne otpornost, kada y_j predstavlja rezultate merenja struje, i_j, a x_j rezultate merenja napona, u_j, da li dobijena optimalna vrednost provodnosti, iz istih rezultata merenja, odgovara dobijenoj vrednosti otpornosti? Možete li da izgenerišete numerički primer? Kada se dobijeni rezultati poklapaju?

Drugo pitanje za razmišljanje: ako svaki par vrednosti u_j, i_j određuje vrednost otpornosti $R_j = u_j/i_j$ i vrednost provodnosti $G_j = i_j/u_j$, a finalna vrednost otpornosti i provodnosti se određuju kao aritmetičke sredine pojedinačnih vrednosti, $R = \frac{1}{n}\sum_{j=1}^{n} R_j$ i $G = \frac{1}{n}\sum_{j=1}^{n} G_j$, kolika je razlika u odnosu na vrednosti koje se dobijaju optimalnim fitovanjem konstante proporcionalnosti? Kom rezultatu treba najviše verovati i zašto?

Treće pitanje za razmišljanje: kako karakterisati varijaciju rezultata opisanih merenja otpornosti i provodnosti i kako proceniti važenje modela da su struja i napon međusobno proporcionalni?

Razmišljajte i pitajte se. Sprovođenje algoritama danas rade računari, a i u budućnosti će to raditi, sve više. Na vama je da smišljate algoritme koji su po nečemu bolji od postojećih. Posao u sprovođenju algoritama nećete naći, sigurno ne na dug rok.

5 Fitovanje parametara linearne funkcije

Sledeći slučaj fitovanja međusobne zavisnosti dve promenljive koji ćemo razmatrati jeste slučaj fitovanja parametara linearne funkcije

$$y = \alpha\, x + \beta \tag{40}$$

pod pretpostavkom da su nam poznate vrednosti nezavisno promenljive

$$x_1,\, x_2,\, \ldots\, x_n \tag{41}$$

u n tačaka, kao i njima odgovarajuće vrednosti zavisno promenljive

$$y_1,\, y_2,\, \ldots\, y_n \tag{42}$$

za $i \in \{1,\, \ldots\, n\}$. Opet, u idealnom slučaju bi važilo

$$y_i = \alpha\, x_i + \beta \tag{43}$$

pa bi iz bilo koja dva para vrednosti y_i, x_i bilo moguće odrediti parametre α i β. Kako sada imamo dva parametra koje treba odrediti, potrebne su bar dve jednačine za njihovo određivanje, $n \geq 2$. Zbog netačnosti u utvrđivanju vrednosti x_i i y_i, kao i zbog nepotpunosti modela, (43) neće nužno biti ispunjeno, već će postojati greške

$$\delta_i \triangleq y_i - \alpha\, x_i - \beta. \tag{44}$$

Iz ranije razmatranih razloga, usled tendencije grešaka da se međusobno potiru, ukupnu grešku na razmatranom skupu podataka definišemo kao zbir kvadrata pojedinačnih grešaka

$$\Delta\,(\alpha,\, \beta) \triangleq \sum_{i=1}^{n} \delta_i^2 = \sum_{i=1}^{n} \left(y_i - \alpha\, x_i - \beta\right)^2 \tag{45}$$

i sada je to funkcija dva nepoznata parametra α i β, pošto su vrednosti x_i i y_i za $i \in \{1,\, \ldots\, n\}$ poznate, to su poznati brojevi. Optimalne vrednosti parametara α i β minimizuju vrednost ukupne greške $\Delta\,(\alpha,\, \beta)$, kada su vrednosti parcijalnih izvoda

$$\frac{\partial\, \Delta\,(\alpha,\, \beta)}{\partial\, \alpha} = -2 \sum_{i=1}^{n} x_i\, \left(y_i - \alpha\, x_i - \beta\right) \tag{46}$$

i

$$\frac{\partial\, \Delta\,(\alpha,\, \beta)}{\partial\, \beta} = -2 \sum_{i=1}^{n} \left(y_i - \alpha\, x_i - \beta\right) \tag{47}$$

jednake nuli

$$\frac{\partial\, \Delta\,(\alpha,\, \beta)}{\partial\, \alpha} = 0 \tag{48}$$

i

$$\frac{\partial \Delta(\alpha, \beta)}{\partial \beta} = 0. \tag{49}$$

Ovaj uslov rezultuje sa dve linearne jednačine

$$\sum_{i=1}^{n} x_i y_i - \alpha \sum_{i=1}^{n} x_i^2 - \beta \sum_{i=1}^{n} x_i = 0 \tag{50}$$

i

$$\sum_{i=1}^{n} y_i - \alpha \sum_{i=1}^{n} x_i - n\beta = 0 \tag{51}$$

koje se po nepoznatim parametrima α i β svode na formu

$$\alpha \sum_{i=1}^{n} x_i^2 + \beta \sum_{i=1}^{n} x_i = \sum_{i=1}^{n} x_i y_i \tag{52}$$

i

$$\alpha \sum_{i=1}^{n} x_i + n\beta = \sum_{i=1}^{n} y_i \tag{53}$$

pošto su vrednosti x_i i y_i poznate, a jedine nepoznate u (52) i (53) su α i β. Na ovaj način je sistem od n potencijalno nekonzistentnih jednačina po α i β (43) je sveden na konzistentan sistem od dve jednačine po dve nepoznate (52) i (53). Ovaj metod rešavanja predefinisanog sistema linearnih jednačina koji minimizuje ukupnu kvadratnu grešku se naziva **metod najmanjih kvadrata** [10] i ima jako široku primenu. Stoga je malo verovatno da ćete ikada morati da pišete program za implementaciju ovog metoda: u svim programskim paketima za numeričku obradu podataka on je već implementiran.

Konkretan primer primene metoda najmanjih kvadrata u slučaju određivanja dva parametra za rezultat daje

$$\boxed{\alpha_{opt} = \frac{n \sum x_i y_i - (\sum x_i)(\sum y_i)}{n \sum x_i^2 - (\sum x_i)^2}} \tag{54}$$

i

$$\boxed{\beta_{opt} = \frac{(\sum x_i^2)(\sum y_i) - (\sum x_i)(\sum x_i y_i)}{n \sum x_i^2 - (\sum x_i)^2}} \tag{55}$$

pri čemu je radi preglednosti jednačina u (54) i (55) $\sum_{i=1}^{n}$ zamenjeno simbolom Σ, dok su granice sumiranja podrazumevane pošto su uvek iste.

Primer primene opisanog fitovanja parametara je određivanje elektromotorne sile i unutrašnje otpornosti ekvivalentnog Tevenenovog [11] ili Nortonovog [12] generatora linearnog električnog kola.

6 Fitovanje koeficijenata polinoma

Sledeća generalizacija razmatranog problema optimizacije je fitovanje skupa podataka koji se sastoji iz niza vrednosti nezavisno promenljivih

$$x_1, x_2, \ldots x_n \tag{56}$$

i njima odgovarajućim nizom vrednosti zavisno promenljivih

$$y_1, y_2, \ldots y_n \tag{57}$$

za $i \in \{1, \ldots n\}$, kada je pretpostavljena zavisnost data polinomom

$$y = \sum_{j=0}^{k} a_j \, x^j = a_0 + a_1 \, x + a_2 \, x^2 + \ldots + a_k \, x^k \tag{58}$$

u kome su koeficijenti a_0, a_1, $\ldots$ a_k nepoznati i treba ih odrediti, a ima ih $k+1$. Opet, u slučaju idealnih podataka i idealne pretpostavljene zavisnosti važilo bi

$$y_i = \sum_{j=0}^{k} a_j \, x_i^j = a_0 + a_1 \, x_i + a_2 \, x_i^2 + \ldots + a_k \, x_i^k \tag{59}$$

za $i \in \{1, \ldots n\}$, pri čemu smatramo da je raspoloživ skup podataka dovoljan da se odrede vrednosti nepoznatih koeficijenata polinoma kojih ima $k + 1$, dakle potrebno je $n \geq k + 1$. U slučaju $n = k + 1$, sistem (59) ima jednak broj jednačina i nepoznatih, a ako su sve vrednosti x_i međusobno različite ima jednoznačno rešenje, o čemu će još biti reči, dok je za $n > k + 1$ predefinisian, oblika

$$\begin{aligned} & a_0 + a_1 \, x_1 + a_2 \, x_1^2 + \ldots + a_k \, x_1^k = y_1 \\ & a_0 + a_1 \, x_2 + a_2 \, x_2^2 + \ldots + a_k \, x_2^k = y_2 \\ & \vdots \\ & a_0 + a_1 \, x_n + a_2 \, x_n^2 + \ldots + a_k \, x_n^k = y_n. \end{aligned} \tag{60}$$

Ovaj sistem jednačina je linearan po koeficijentima polinoma i kao takav se može zapisati u matričnoj formi kao

$$\begin{bmatrix} 1 & x_1 & x_1^2 & \ldots & x_1^k \\ 1 & x_2 & x_2^2 & \ldots & x_2^k \\ \vdots & & & & \\ 1 & x_n & x_n^2 & \ldots & x_n^k \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_k \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} \tag{61}$$

što se imenovanjem matrica redukuje na

$$V_x \, A = Y \tag{62}$$

gde je

$$A = [a_0, \, a_1, \, \ldots \, a_k]^T \tag{63}$$

vektor koeficijenata polinoma

$$Y = [y_1, \, y_2, \, \ldots \, y_n]^T \tag{64}$$

vektor vrednosti zavisno promenljive, dok je

$$V_x = \begin{bmatrix} 1 & x_1 & x_1^2 & \ldots & x_1^k \\ 1 & x_2 & x_2^2 & \ldots & x_2^k \\ \vdots & & & & \\ 1 & x_n & x_n^2 & \ldots & x_n^k \end{bmatrix} \tag{65}$$

Vandermondova matrica [13] generisana nizom vrednosti nezavisno promenljive

$$X = [x_1, \, x_2, \, \ldots \, x_n]^T . \tag{66}$$

U slučaju da je $n = k + 1$, Vandermondova matrica je kvadratna, a njena determinanta je različita od nule ako i samo ako su vrednosti x_i međusobno različite [13], pa sistem jednačina po koeficijentima polinoma nužno ima jedinstveno rešenje, kako je već bilo pomenuto.

U slučaju da je $n > k+1$ sistem jednačina je predefinisan i rešava se već opisanim metodom najmanjih kvadrata, tako što se formiraju pojedinačne greške

$$\delta_i = y_i - \sum_{j=0}^{k} a_j \, x_i^j \tag{67}$$

a potom i ukupna kvadratna greška

$$\Delta \left(a_0, \, a_1, \, \ldots \, a_k \right) = \sum_{i=1}^{n} \left(y_i - \sum_{j=0}^{k} a_j \, x_i^j \right)^2 \tag{68}$$

čiji su parcijalni izvodi po nepoznatim koeficijentima polinoma

$$\frac{\partial \Delta \left(a_0, \, a_1, \, \ldots \, a_k \right)}{\partial \, a_j} = -2 \sum_{i=1}^{n} x_i^j \left(y_i - \sum_{j=0}^{k} a_j \, x_i^j \right) \tag{69}$$

za $j \in \{0, \, \ldots \, k\}$. Izjednačavanjem ovih parcijalnih izvoda sa nulom se dobija sistem jednačina po koeficijentima polinoma koji minimizuje (problem optimizacije) ukupnu kvadratnu grešku

$$\sum_{i=1}^{n} x_i^j \left(y_i - \sum_{j=0}^{k} a_j \, x_i^j \right) = 0 \tag{70}$$

što se svodi na linearan sistem od $k+1$ jednačine po $k+1$ nepoznatoj

$$\sum_{i=1}^{n} x_i^j \left(\sum_{j=0}^{k} x_i^j \, a_j \right) = \sum_{i=1}^{n} x_i^j \, y_i \tag{71}$$

za $j \in \{0, \, \ldots \, k\}$. Imajući u vidu definicije (63), (64) i (65), gornji sistem jednačina se u matričnoj formi može zapisati kao

$$V_x^T \, V_x \, A = V_x^T \, Y \tag{72}$$

što je sistem linearnih jednačina od $k+1$ jednačine po $k+1$ nepoznatoj i predstavlja generalnu formu *linear least squares* optimizacionog problema [10], čiji posebni slučajevi su optimizacija vrednosti ponovljenih merenja konstantne veličine (jednodimenzioni slučaj, $k = 0$) i optimizacija parametara linearne funkcije (dvodimenzioni slučaj, $k = 1$). Rešenje sistema jednačina (72) je

$$A = \left(\left(V_x^T \, V_x \right)^{-1} \, V_x^T \right) \, Y \tag{73}$$

gde je $\left(V_x^T \, V_x \right)^{-1} \, V_x^T$ kvadratna matrica poznata kao *Moore-Penrose inverse* matrice V_x [14], što predstavlja generalizaciju pojma inverzne matrice. Taj pojam je postepeno ulazio u matematiku, sa značajnim koracima 1920, 1951. i 1955. godine [14]. U opštem slučaju (kada V_x nije Vandermondova matrica) i kada je V_x kvadratna invertibilna matrica,

$$\left(V_x^T \, V_x \right)^{-1} \, V_x^T = V_x^{-1} \, \left(V_x^T \right)^{-1} \, V_x^T = V_x^{-1} \, \left(\left(V_x^T \right)^{-1} \, V_x^T \right) = V_x^{-1} \tag{74}$$

Moore-Penrose inverse matrice V_x se svodi na običnu inverznu matricu V_x^{-1}, koja u posebnom slučaju Vandermondove matrice V_x generisane vektorom X postoji pod uslovom da su sve vrednosti u X vektoru međusobno različite [13]. Problem najmanjih kvadrata [10] je toliko opšti da je implementiran u svakom relevantnom programskom paketu za numeričku analizu podataka. Potrebe za programiranjem tog metoda danas nema, optimizovani programi su dostupni kao slobodan softver [15], o čemu će biti reči u poglavlju koje sledi.

7 Programi za obradu rezultata merenja

Opisani postupci obrade rezultata merenja, koji spadaju u osnovne, toliko su uobičajeni da ih paketi za numeričku obradu i analizu podataka već imaju implementirane. Stoga je jako malo verovatno da će korisnik ikada sam pisati programe za obradu rezultata merenja, osim u slučaju da mu treba neka specifična obrada, ali i tada veliki deo posla mogu da reše već implementirane funkcije. U ovom tekstu će biti prikazana obrada rezultata merenja primenom programskog paketa GNU Octave [16, 17] i korišćenjem odgovarajućih modula za programski jezik Python [18, 19, 20, 21]. Obe prikazane opcije su slobodan softver [15]. Podrazumevano je osnovno poznavanje rada sa navedenim programima, a razmatrane su samo funkcije od značaja za obradu rezultata merenja.

7.1 GNU Octave

Funkcije programskog paketa GNU Octave [16, 22, 23] će biti ilustrovane na dva niza koji predstavljaju rezultate hipotetičkih merenja

```
x = [1, 2, 3]
```

i

```
y = [2 4 6]
```

koji su namerno zadati u različitoj formi, pošto GNU Octave dopušta kod zadavanja nizova i blanko karakter kao separator, što je u praksi zgodno.

Srednja vrednost niza se dobija korišćenjem funkcije `mean` za koju se potpuni opis može naći na

`https://octave.sourceforge.io/octave/function/mean.html`

i koja se primenjuje kao

```
xm = mean(x)
```

što za rezultat daje `xm = 2` i

```
ym = mean(y)
```

što za rezultat daje `ym = 4`.

Standardna devijacija rezultata merenja se dobija korišćenjem funkcije `std` za koju se potpuni opis može naći na

`https://octave.sourceforge.io/octave/function/std.html`

i koja računa standardnu devijaciju članova niza prema (28) tako da

```
sx = std(x)
```

i

```
sy = std(y)
```

za rezultat daju `sx = 1` i `sy = 2`.

Fitovanje konstante proporcionalnosti prema (39) se može realizovati primenom vektorskih operacija imlementiranih u GNU Octave kao

```
alpha1 = sum(x .* y) / sum(x.^2)
```

što daje `alpha1 = 2`, kao i rešavanjem predefinisanog sistema linearnih jednačina (33) po metodi najmanjih kvadata (*linear least squares*, levo deljenje, [22]) čiji opis se može naći na

`https://octave.sourceforge.io/octave/function/mldivide.html`

kao

```
alpha2 = x' \ y'
```

što daje `alpha2 = 2.0000`.

Vandermondova matrica se generiše funkcijom `vander` koja je opisana na

`https://octave.sourceforge.io/octave/function/vander.html`

i za vektor rezultata merenja nezavisno promenljive `x` puna Vandermondova matrica se generiše sa

```
Vx0 = vander(x)
```

Valja uočiti da su u implementaciji programa kolone Vandermondove matrice poređane redom suprotnim od redosleda kolona u ovde korišćenoj definiciji (65), kao i u [13], pa će koeficijenti polinoma uz veće stepene imati manje indekse. Ukoliko ovo predstavlja problem, dobijena Vandermondova matrica se može preurediti korišćenjem funkcije `flip` čiji se opis može naći na

`https://octave.sourceforge.io/octave/function/flip.html`

kao

```
Vx1 = flip(Vx0, 2)
```

Polinom reda $k = n - 1$ koji prolazi kroz svih n tačaka određenih rezultatima merenja se dobija kao

```
A = Vx0 \ y'
```

što u našem primeru daje $A_2 = 0$, $A_1 = 2$ i $A_0 = 0$.

Uobičajeno je da merenja sadrže redundantne tačke, da je prikupljeno više rezultata od minimalnog broja tačaka potrebnih da bi se odredili koeficijenti polinoma kojim se modeluje proces, da je $k < n - 1$. Tada se Vandermondova matrica (ili redukovana Vandermondova matrica) određuje kao

```
Vx = vander(x, k + 1)
```

što se slučaju $k = 1$, kada fitujemo pravu liniju, svodi na

```
Vx2 = vander(x, 2)
```

pa su koeficijenti polinoma

```
a = Vx2 \ y'
```

što daje $a_1 = 2$ i $a_0 \approx 0$.

Problem računanja koeficijenata polinoma koji optimalno fituje rezultate merenja je toliko uobičajen da je moguće direktno doći do koeficijenata primenom funkcije `polyfit` čiji je opis dat na

`https://octave.sourceforge.io/octave/function/polyfit.html`

i koja je implementirana baš zbog toga što je problem toliko čest u praksi. Funkcija ima formu

```
polyfit(x, y, k)
```

gde su `x` i `y` vektori sa rezultatima merenja nezavisno i zavisno promenljive, dok je `k` red fitovanog polinoma. Na datom primeru podataka, koeficijenti polinoma drugog reda koji prolazi kroz sve tačke određene rezultatima merenja se dobija iz

```
p2 = polyfit(x, y, 2)
```

što za rezultat daje približno `p2 = [0 2 0]`, dok se fitujući polinom prvog stepena, odnosno prava linija koja najbolje fituje date podatke dobija iz

```
p1 = polyfit(x, y, 1)
```

što za rezultat daje približno `p1 = [2 0]`.

7.2 Python

Problemi čije je rešavanje prethodno prikazano primenom programa GNU Octave, moguće je rešiti i u programskom jeziku Python uz korišćenje modula NumPy [24]. Ovde će biti prikazano korišćenje funkcija koje implementiraju metode obrade rezultata merenja opisane u tekstu u PyLab okruženju, koje se iz komandne linije može pokrenuti koristeći IPython [25] pozivom

```
ipython3 --pylab
```

a moguće je koristiti i Jupyter okruženje [26] na različitim platformama. U okviru Python programa, PyLab okruženje se dobija importovanjem modula `pylab` sa

```
from pylab import *
```

nakon čega je moguće koristiti njegove funkcije.

Nizovi koji predstavljaju rezultate merenja u našim primerima se zadaju sa

```
x = array([1, 2, 3])
```

i

```
y = array([2, 4, 6])
```

gde treba uočiti da je zapeta neophodan separator između elemenata, za razliku od GNU Octave gde je blanko karakter dovoljan separator.

Srednja vrednost se dobija primenom funkcije `mean` čiji opis se može naći na

`https://numpy.org/doc/stable/reference/generated/numpy.mean.html`

i na našim primerima `xs = mean(x)` rezultuje sa `xs = 2.0`, dok `ys = mean(y)` rezultuje sa `ys = 4.0`.

Standardna devijacija elemenata niza se može odrediti pomoću funkcije `std` čiji je opis dat na

`https://numpy.org/doc/stable/reference/generated/numpy.std.html`

ali za razliku od istoimene funkcije u GNU Octave, u ovom slučaju se normalizacija podrazumevano (*by default*) vrši sa n, ne sa $n-1$, dakle prema (29). Tako

```
sx = std(x)
```

rezultuje sa `sx = 0.816496580927726`, dok

```
sy = std(y)
```

rezultuje sa `sy = 1.632993161855452`, što je bitno različito u odnosu na vrednosti koje je istoimena funkcija dala na osnovu istih argumenata u programu GNU Octave.

Jedan (loš) način za primenu Beselove korekcije je skaliranje rezultata koji daje funkcija `std` faktorom koji zavisi od dužine niza, a koja se dobija funkcijom `len`

```
n = len(x)
```

pa je uz Beselovu korekciju vrednost standardne devijacije

```
ssx = sqrt(n / (n - 1)) * std(x)
```

što daje `ssx` ≈ 1 i

```
ssy = sqrt(n / (n - 1)) * std(y)
```

što daje `ssy` ≈ 2.

Bolji način za primenu Beselove korekcije je da pročitate uputstvo za primenu funkcije `std` u kome se opisuje dejstvo argumenta koji se zadaje preko parametra (*keyword argument*) `ddof` (*delta degrees of freedom*) kojim se umanjuje imenilac normalizujućeg faktora za vrednost parametra `ddof`, pa se za

```
sssx = std(x, ddof = 1)
```

dobija `ssx = 1.0`, dok se za

```
sssy = std(y, ddof = 1)
```

dobija `ssy = 2.0`. Na ovaj način se program brže izvršava u odnosu na prethodno opisan metod, a uz to zadavanje parametra za normalizaciju omogućava i primenu drugih vrednosti, onih koje optimizuju procenu standardne devijacije [27], a ne varijanse kako je dalo naše izvođenje [8]. Ovo je napredna tema koja ilustruje koliko je prikazana analiza uvodna i elementarna.

U modulu NumPy je vektorizacija drugačije sintaksno rešena u odnosu na GNU Octave, operacije nad nizovima su podrazumevano po elementima (*elementwise*), pa se fitovanje konstante proporcionalnosti jednostavno vrši sa

```
alpha1 = sum(x * y) / sum(x**2)
```

što daje `alpha1 = 2.0`, dok se rešavanje sistema linearnih jednačina u smislu najmanjeg kvadratnog odstupanja vrši pozivom funkcije `lstsq` opisane na

`https://numpy.org/doc/stable/reference/generated/numpy.linalg.lstsq.html`

sa

```
alpha2 = lstsq(x.reshape((n, 1)), y, rcond = None)[0][0]
```

što za rezultat daje `alpha2 = 2.0000000000000004`. Ovaj rezultat se malo razlikuje od prethodno dobijene vrednosti usled numeričke greške postupka.

U PyLab okruženju se Vandermondova matrica dobija pokretanjem funkcije `vander` koja je opisana na

`https://numpy.org/doc/stable/reference/generated/numpy.vander.html`

pa se puna Vandermondova matrica po vektoru `x` dobija kao

```
Vx0 = vander(x)
```

gde su elementi matrice dati u obrnutom redosledu kolona u odnosu na (65) i na [13]. Ukoliko je redosled kolona od značaja, matrica se može preurediti primenom funkcije `flip`, slično kao u programu GNU Octave, sa

```
Vx1 = flip(Vx0, 1)
```

ili primenom funkcije `fliplr` sa

```
Vx2 = fliplr(Vx0)
```

Posle formiranja Vandermondove matrice se koeficijenti punog fitujućeg polinoma koji prolazi kroz sve merne tačke dobijaju kao

```
A = lstsq(Vx0, y)[0]
```

što daje `A` $\approx [0, 2, 0]$. Redukovana Vandermondova matrica na prve dve kolone (u skladu sa u ovom tekstu korišćenom notacijom prve dve kolone) se dobija sa

```
Vx3 = vander(x, 2)
```

pa se koeficijenti optimalne fitujuće prave dobijaju preko

```
a = lstsq(Vx3, y)[0]
```

što daje `a` $\approx [2, 0]$.

Kao i kod programa GNU Octave, fitovanje polinoma je standardna funkcija, već implementirana, opis funkcije se može naći na

```
https://numpy.org/doc/stable/reference/generated/numpy.polyfit.html
```

i ima opšti oblik

```
numpy.polyfit(x, y, k)
```

gde je `X` vektor vrednosti nezavisno promenljive, a `Y` vektor vrednosti zavisno promenljive. U slučaju

```
p2 = polyfit(x, y, 2)
```

dobija se `p2` $\approx [0, 2, 0]$, dok se za

```
p1 = polyfit(x, y, 1)
```

dobija `p1` $\approx [2, 0]$, što u oba slučaja daje vezu između promenljivih $y = 2\,x$.

8 Zaključak

U ovom tekstu je prvo razmatrana obrada ponovljenih rezultata merenja konstantne veličine, gde je pokazano da je njihova aritmetička sredina optimalna vrednost koja minimizuje najmanje kvadratno odstupanje. Potom je razmatran pojam standardne devijacije kojim se karakteriše rasipanje rezultata merenja, kao i Beselova korekcija koja optimalno estimira vrednost varijanse rezultata merenja kada tačna vrednost nije poznata. U daljem tekstu je razmatrano fitovanje međusobne zavisnosti dve veličine polinomskim funkcijama, prvo proporcionalnom vezom, potom linearnom funkcijom i na kraju polinomom proizvoljnog reda. Rešavanje predefinisanog sistema linearnih jednačina u smislu minimizacije srednje kvadratne greške je prikazano, uvedena je Vandermondova matrica i pojam pseudoinverzne matrice (*Moore-Penrose pseudoinverse*) koji generalizuje pojam inverzne matrice na matrice koje nisu kvadratne, a svodi se na pojam inverzne matrice u slučaju invertibilne kvadratne matrice. Svim navedenim problemima je pristupljeno kao posebnim slučajevima opšteg problema optimizacije, koji ima mnogo širi značaj od obrade rezultata merenja, pa će stečena znanja i iskustva imati širu upotrebnu vrednost.

Na kraju, ali ne najmanje važno (*last, but not least*), je prikazano kako se opisani postupci implementiraju na digitalnom računaru opšte namene isključivom primenom slobodnog softvera. Ovde je naglašena činjenica da su opisani postupci do te mere uobičajeni i standardni da su već implementirani kao funkcije u programskim jezicima visokog nivoa namenjenim za obradu numeričkih podataka, pa da sam korisnik nema nikakvu potrebu da programira dok god koristi standardne postupke. Stoga, u slučaju primene poznatih i široko rasprostranjenih metoda obrade podataka ima smisla proučiti već dostupne programe, pošto ima dobrih izgleda da su potrebni metodi već implementirani, a implementacija optimizovana.

Literatura

[1] https://en.wikipedia.org/wiki/Average_absolute_deviation

[2] https://en.wikipedia.org/wiki/Arithmetic_mean

[3] https://en.wikipedia.org/wiki/Median

[4] https://en.wikipedia.org/wiki/Mode_(statistics)

[5] https://en.wikipedia.org/wiki/Variance

[6] https://en.wikipedia.org/wiki/Dimensional_analysis

[7] https://en.wikipedia.org/wiki/Standard_deviation

[8] https://en.wikipedia.org/wiki/Bessel%27s_correction

[9] https://en.wikipedia.org/wiki/Curve_fitting

[10] https://en.wikipedia.org/wiki/Linear_least_squares

[11] https://en.wikipedia.org/wiki/Th%C3%A9venin%27s_theorem

[12] https://en.wikipedia.org/wiki/Norton%27s_theorem

[13] https://en.wikipedia.org/wiki/Vandermonde_matrix

[14] https://en.wikipedia.org/wiki/Moore%E2%80%93Penrose_inverse

[15] https://www.fsf.org/

[16] https://octave.org/

[17] http://tnt.etf.bg.ac.rs/~oe4sae/12th-2020.pdf

[18] https://www.python.org/

[19] http://tnt.etf.bg.ac.rs/~oe4sae/9th-2020.pdf

[20] http://tnt.etf.bg.ac.rs/~oe4sae/10th-2020.pdf

[21] http://tnt.etf.bg.ac.rs/~oe4sae/11th-2020.pdf

[22] https://docs.octave.org/octave.pdf

[23] https://www.wcc.vccs.edu/sites/default/files/Introduction-to-GNU-Octave.pdf

[24] https://numpy.org/

[25] https://ipython.org/

[26] https://jupyter.org/

[27] https://en.wikipedia.org/wiki/Unbiased_estimation_of_standard_deviation

Dodatak

GNU Octave source code

```
# rezultati merenja
x = [1, 2, 3]
y = [2 4 6]

# srednje vrednosti
xs = mean(x)
ys = mean(y)

# standardne devijacije
sx = std(x)
sy = std(y)

# fitovanje konstante proporcionalnosti
alpha1 = sum(x .* y) / sum(x.^2)
alpha2 = x' \ y'

# Vandermondova matrica
Vx0 = vander(x)
Vx1 = flip(Vx0, 2)

# fitovanje polinoma
A = Vx0 \ y'

# fitovanje linearne funkcije
Vx2 = vander(x, 2)
a = Vx2 \ y'

# direktno fitovanje polinoma
p2 = polyfit(x, y, 2)
p1 = polyfit(x, y, 1)
```

Python source code

```
from pylab import *

# rezultati merenja
x = array([1, 2, 3])
y = array([2, 4, 6])

# srednje vrednosti
xs = mean(x)
ys = mean(y)

# standardne devijacije
sx = std(x)
sy = std(y)

# standardne devijacije, Beselova korekcija
n = len(x)
ssx = sqrt(n / (n - 1)) * std(x)
ssy = sqrt(n / (n - 1)) * std(y)

# standardne devijacije, Beselova korekcija, implementirano
sssx = std(x, ddof = 1)
sssy = std(y, ddof = 1)

# fitovanje konstante proporcionalnosti
alpha1 = sum(x * y) / sum(x**2)
alpha2 = lstsq(x.reshape((n, 1)), y, rcond = None)[0][0]

# Vandermondova matrica
Vx0 = vander(x)
Vx1 = flip(Vx0, 1)
Vx2 = fliplr(Vx0)

# fitovanje polinoma
A = lstsq(Vx0, y, rcond = None)[0]

# fitovanje linearne funkcije
Vx3 = vander(x, 2)
a = lstsq(Vx3, y, rcond = None)[0]

# direktno fitovanje polinoma
p2 = polyfit(x, y, 2)
p1 = polyfit(x, y, 1)
```