Optimalno upravljanje nelinearnim sustavima primjenom neuronskih mreža

Kasać, Josip

SVEUČILIŠTE U ZAGREBU
Poslijediplomski studij
VOĐENJE I UPRAVLJANJE POKRETNIM OBJEKTIMA

Josip Kasač

Optimalno upravljanje nelinearnim sustavima primjenom neuronskih mreža

Magistarski rad

Zagreb, studeni 1998.

Ovaj rad je izrađen na sveučilišnom poslijediplomskom studiju
Vođenje i upravljanje pokretnim objektima

Voditelj rada:

Prof. dr. sc. Branko Novaković

Fakultet strojarstva i brodogradnje, Zagreb

Rad sadrži :

- 132 stranice
- 61 sliku
- 0 tablica

Predgovor

Želio bih se na ovom mjestu zahvaliti mentoru Prof. dr. sc. Branku Novakoviću na stručnoj pomoći i vođenju tokom izrade ove radnje.
Također bih se zahvalio Ireni Bakoč na savjesnoj korekturi teksta.

S A D R Ž A J

1. Uvod

Temelji teorije optimalnog upravljanja postavljani su krajem pedesetih godina radovima Pontryagina [1] i Bellmana [2], da bi do kraja šesdesetih teorija poprimila svoj konačni oblik [3], [4]. Paralelno s razvojem teorije razvijale su se i različite numeričke metode za rješavanje problema optimalnog upravljanja. Iako teorija optimalnog upravljanja pruža univerzalni okvir za sintezu upravljanja multivarijabilnim dinamičkim sustavima, ona danas nije zastupljena u praktičnim primjenama s punim opsegom svojih mogućnosti, izuzev u slučajevima gdje postoje, uvjetno rečeno, analitička rješenja, kao npr. problem regulacije linearnih sustava s kvadratnim kriterijem optimalnosti. Razlog tome leži u matematičkoj kompleksnosti nužnih uvjeta optimalnosti i neophodnosti primjene numeričkih metoda. S druge strane, u odnosu na neke metode sinteze upravljanja multivarijabilnim dinamičkim sustavima, teorija optimalnog upravljanja može razmatrati i problem ograničenja varijabli stanja i upravljanja.

Iako je teorija optimalnog upravljanja u početku svog razvoja imala gotovo isključivu primjenu na problemima vođenja tehničkih sustava, primjetan je zadnjih godina sve veći trend njene primjene u ekonomiji i menadžmentu [5]-[7]. Pomoću teorije optimalnog upravljanja tretirani su mnogi problemi ekonomije i menadžmenta kao npr. optimalni ekonomski rast i ekonomsko planiranje, ekonomska stabilizacija, međunarodna trgovina, regionalna ekonomija, urbana ekonomija, kontrola populacije, kontrola zagađenja okoliša, kontrola naoružanja, dinamička teorija organizacije. Centralni problem ekonomije je optimalna raspodjela ograničenih sredstava u nekom vremenskom trenutku (statička optimizacija) ili tokom određenog vremena (dinamička optimizacija). Izbor između manje potrošnje sada i veće poslije, izbor između kratkoročnog napora i dugoročnog profita, problem je dinamičke optimizacije, itd. Drugim riječima, problem se sastoji u raspodjeli investicija po različitim sektorima tijekom cijelog perioda ekonomskog planiranja.

S druge strane, teorija optimalnog upravljanja ima ključnu ulogu u formulaciji opće teorije sustava, odnosno, opće teorije upravljanja, koja bi vrijedila kako za tehničke, tako i za ekonomske, biološke sustave i ljudske organizacije. Danas se principi, metode, tehnike i postupci ove teorije pokušavaju prenijeti i na druge oblasti, tako da tehnička teorija upravljanja dobiva širi značaj. Ta tendencija je posebno izražena u jednoj grani opće teorije sustava; teoriji hijerarhijskih sustava s više nivoa [8]. Osnovna motivacija formulacije teorije hijerarhijskih sustava proisticala je iz područja automatizacije i upravljanja kompleksnim industrijskim sustavima, da bi kasnije prerasla u teoriju upravljanja društvenim organizacijama. Ključni pojam teorije hijerarhijskih sustava je princip koordinacije između različitih nivoa (elemenata) sustava, kako vertikalnih tako i horizontalnih. S te strane, problem koji razmatramo u ovom radu, kooperativni rad dva robota, predstavlja analogiju koordinacije dva elementa sustava na istom hijerarhijskom nivou.

Osnovna motivacija ovog rada je razvoj numeričkih algoritama koji bi omogućili operativnu primjenu optimalnog upravljanja na složene, nelinearne multivarijabilne sustave sa zadanim ograničenjima varijabli stanja i upravljanja. Pri tome je nužno povlačenje analogija s poznatim arhitekturama neuronskih mreža, ako postoji namjera primjene numeričkih algoritama optimalnog upravljanja na velike, kompleksne sustave (sustave visokih dimenzija vektora stanja), radi iskorištavanja mogućnosti paralelnog procesiranja. U radu se pod pojmom nelinearnost podrazumjeva jednoznačna nelinearnost, odnosno, ne tretira se višeznačna nelinearnost poput histereze. Također, svi algoritmi (osim u slučaju podpoglavlja 8.4) su off-line algoritmi, odnosno, ne primjenjuju se direktno u procesu upravljanja.

U drugom poglavlju razmatraju se neki elementarni rezultati iz teorije optimalnog upravljanja, s posebnim naglaskom na tretiranje raznih vrsta mogućih ograničenja. Cilj tog poglavlja je da se ilustriraju matematički problemi s kojima se suočavamo u slučaju direktne primjene nužnih uvjeta optimalnosti za nalaženje optimalnog rješenja. Ograničenja tipa nejednakosti po varijablama upravljanja, a posebno po varijablama stanja, čine nužne uvjete optimalnosti gotovo neupotrebljivim za primjenu osim za niždimenzionalne linearne sustave.

U trećem poglavlju razmatraju se neke standardne numeričke metode optimalnog upravljanja. Numeričko tretiranje problema optimalnog upravljanja je neophodno zbog matematičke kompleksnosti nužnih uvjeta optimalnosti. Numeričke metode optimalnog upravljanja mogu se podjeliti na metode koje numerički tretiraju nužne uvjete optimalnosti (metoda kvazilinearizacije, diskretno dinamičko programiranje) i direktne metode u koje možemo ubrojiti Ritzovu metodu i metodu diskretizacije kojom kontinuirani problem optimalnog upravljanja svodimo na problem nelinearnog programiranja.

U četvrtom poglavlju daje se izvod gradijentnog algoritma za diskretizirani problem optimalnog upravljanja. Izvod je baziran na principu BPTT (*backpropagation-through-time*) algoritma, odnosno na primjeni svojstva ulančanih derivacija i iteriranja unazad u vremenu. Zbog kompleksnosti izvoda algoritma za nelinearne multivarijabilne sustave, najprije smo izveli algoritam za jednodimenzionalni slučaj radi ilustracije osnovnog principa izvođenja algoritma. Također smo dali prikaz analogije između dobivenog algoritma za jednodimenzionalan slučaj i arhitekture dinamičkih neuronskih mreža, kao i unaprijednih neuronskih mreža s $N-1$ skrivenih slojeva (N je broj diskretiziranih vremenskih podintervala). Zatim slijedi izvod algoritma za općenite nelinearne multivarijabilne sustave bez ograničenja varijabli stanja i upravljanja. Nakon toga, u osnovnu shemu algoritma postepeno dodajemo modifikacije za slučaj ograničenja tipa rubnih uvjeta, te ograničenja tipa nejednakosti i jednakosti po varijablama stanja i upravljanja. Na kraju dajemo pregled konačnog oblika algoritma za problem optimalnog upravljanja u kojeg su uključeni svi, gore navedeni, tipovi ograničenja. Dobiveni algoritam smo testirali na nekim jednostavnim linearnim primjerima (s poznatim analitičkim rješenjem) radi mogućnosti direktne usporedbe numeričkih rezultata s analitičkim.

U petom poglavlju dajemo prikaz metode konjugiranog gradijenta, koja spada u metode drugog reda i ima puno bolja konvergencijska svojstva od gradijentnog algoritma s konstantnim koeficijentom konvergencije. Metoda konjugiranog gradijenta zahtijeva dodatnu jednodimenzionalnu minimizaciju funkcije cilja u svakom koraku iteriranja algoritma što povećava vrijeme izračunavanja jedne iteracije gradijentnog

algoritma. Međutim, ubrzanje konvergencije koje pri tome postižemo svakako opravdava korištenje metode konjugiranog gradijenta u slučajevima gdje možemo dobiti egzaktnu vrijednost gradijenta funkcije cilja.

U šestom poglavlju razmatramo problem vremenski optimalnog upravljanja. Prvo primjenjujemo analognu proceduru kao u četvrtom poglavlju uzimajući period diskretizacije promjenjivom varijablom i računamo derivaciju definirane funkcije cilja po periodu diskretizacije. Dobiveni algoritam je vrlo glomazan i nepraktičan za implementaciju. Nakon toga prikazali smo drugi algoritam koji je vrlo jednostavan i zahtijeva samo izračunavanje sume kaznenih funkcija.

U sedmom poglavlju razmatramo primjenu algoritama prikazanih u prethodnim poglavljima na problem optimalnog upravljanja robotom s dva stupnja slobode. Prvo se razmatra vremenski optimalno upravljanje robotom s ograničenjima varijabli upravljanja, a zatim isti problem s uvjetom zaobilaženja prepreke. Nakon toga razmatramo problem optimalnog upravljanja kooperativnim radom dva robota tokom prenošenja krutog tereta s jednog mjesta na drugo. Najprije smo razmatrali problem uz uvjet ograničenja upravljačkih varijabli i uvjet konstantne udaljenosti između prihvatnica manipulatora. Iz analize dobivenih rezultata proizlazi da bi, i pored zadovoljavanja svih postavljenih uvjeta, dolazilo do međusobnog sudara ruku manipulatora. Zatim smo, pored navedenih uvjeta, dodali uvjet međusobnog izbjegavanja sudara čime smo otklonili navedeni problem.

U osmom poglavlju izvodimo algoritam za određivanje koeficijenata pojačanja statičkog regulatora u povratnoj petlji. Izvod se, također, temelji na principu BPTT algoritma. Razmatra se problem regulacije za nelinearne multivarijabilne sustave s ograničenjima varijabli upravljanja.

2. Teorijske osnove optimalnog upravljanja

U ovom poglavlju prikazat ćemo neke ključne rezultate iz teorije optimalnog upravljanja s posebnim naglaskom na tretiranje raznih oblika ograničenja koji se mogu pojaviti u formulaciji problema.

Polazište za tretiranje problema optimalnog upravljanja je klasični varijacioni račun. Ograničenja koja nameće pristup preko varijacijskog računa mogu se izbjeći korištenjem Pontryaginovog principa maksimuma ili Bellmanovim konceptom dinamičkog programiranja. Analiza optimuma preko navedenih postupaka najčešće je ograničena na traženje nužnih uvjeta.

Izlaganje u ovom poglavlju prati reference [1]-[6], [9], [10].

2.1 Parametarska optimizacija s ograničenjima tipa jednakosti

Najjednostavnija klasa problema parametarske optimizacije je nalaženje vrijednosti m parametara $u_1, u_2, \ldots, u_m$ koje minimiziraju funkciju cilja (indeks performansi) koja je funkcija tih parametara,

$$F(u_1, u_2, \ldots, u_m).$$

Ako uvedemo tzv. vektor odlučivanja (*decision vector*)

$$\mathbf{u}^T = \begin{bmatrix} u_1 & u_2 & \cdots & u_m \end{bmatrix}^T,$$

možemo napisati funkciju cilja kao $F(\mathbf{u})$. Ako nema ograničenja na moguće vrijednosti od $\mathbf{u}$ i ako funkcija $F(\mathbf{u})$ ima prvu i drugu derivaciju za sve vrijednosti vektora $\mathbf{u}$, tada je nužan uvjet za minimum

$$\frac{\partial F(\mathbf{u})}{\partial \mathbf{u}} = \mathbf{0}, \tag{2.1}$$

odnosno, po komponentama

$$\frac{\partial F(\mathbf{u})}{\partial u_i} = 0, \quad i = 1, 2, \ldots, m. \tag{2.2}$$

Iz gornjeg uvjeta dobivamo sustav od m algebarskih jednadžbi, koje su općenito nelinearne, sa m nepoznatih parametara $u_1, u_2, \ldots, u_m$. Sve moguće vrijednosti parametara $u_1, u_2, \ldots, u_m$, koje zadovoljavaju gornji uvjet nazivamo stacionarnim točkama.

Općenitija klasa problema parametarske optimizacije je nalaženje vrijednosti m parametara $u_1, u_2, \ldots, u_m$ koji minimiziraju funkciju cilja koja je skalarna funkcija $n+m$ parametara

$$F(x_1, x_2, \ldots, x_n, u_1, u_2, \ldots, u_m),$$

gdje su n parametara stanja $x_1, x_2, \ldots, x_n$ određena parametrima odlučivanja kroz skup od n jednadžbi

$$f_i(x_1, x_2, \ldots, x_n, u_1, u_2, \ldots, u_m) = 0, \quad i = 1,2,\ldots,n.$$

Uvođenjem

$$\mathbf{x}^T = \begin{bmatrix} x_1 & x_2 & \cdots & x_n \end{bmatrix}^T, \qquad \mathbf{f}^T = \begin{bmatrix} f_1 & f_2 & \cdots & f_n \end{bmatrix}^T,$$

problem možemo formulirati na slijedeći način: treba naći vektor $\mathbf{u}$ koji minimizira

$$F(\mathbf{x}, \mathbf{u}), \tag{2.3}$$

gdje je vektor stanja određen vektorom odlučivanja pomoću jednadžbi

$$\mathbf{f}(\mathbf{x}, \mathbf{u}) = \mathbf{0}. \tag{2.4}$$

Rješenje (stacionarna točka) gornjeg problema je ona vrijednost vektora $\mathbf{u}$, u kojoj je $dF = 0$, za proizvoljni $d\mathbf{u}$, dok je $d\,\mathbf{f} = 0$. Imamo

$$dF = F_{\mathbf{x}} \cdot d\mathbf{x} + F_{\mathbf{u}} \cdot d\mathbf{u}, \tag{2.5}$$

$$d\,\mathbf{f} = \mathbf{f}_{\mathbf{x}} \cdot d\mathbf{x} + \mathbf{f}_{\mathbf{u}} \cdot d\mathbf{u}, \tag{2.6}$$

gdje su $F_{\mathbf{x}}$, F, vektori parcijalnih derivacija F po komponentama vektora **x**, odnosno **u**, respektivno, dok su $\mathbf{f}_{\mathbf{x}}$, $\mathbf{f}_{\mathbf{u}}$ Jakobijeve matrice parcijalnih derivacija komponenti vektora **f** po komponentama vektora **x**, odnosno **u**, respektivno. Iz zahtjeva $d\,\mathbf{f} = 0$, pod uvjetom da je $\mathbf{f}$ nesingularna matrica, možemo izraziti $d\mathbf{x}$ preko $d\mathbf{u}$

$$d\mathbf{x} = -\mathbf{f}_{\mathbf{x}}^{-1} \cdot \mathbf{f}_{\mathbf{u}} \cdot d\mathbf{u}. \tag{2.7}$$

Uvrstimo li gornji izraz u izraz za dF, dobivamo

$$dF = (F_{\mathbf{u}} - F_{\mathbf{x}} \cdot \mathbf{f}_{\mathbf{x}}^{-1} \cdot \mathbf{f}_{\mathbf{u}}) \cdot d\mathbf{u}. \tag{2.8}$$

Ako je $dF = 0$ za proizvoljni $d\mathbf{u}$, tada je neophodno da

$$F_{\mathbf{u}} - F_{\mathbf{x}} \cdot \mathbf{f}_{\mathbf{x}}^{-1} \cdot \mathbf{f}_{\mathbf{u}} = \mathbf{0}. \tag{2.9}$$

Gornjih m jednadžbi, zajedno s n jednadžbi (2.4) određuju m komponenti vektora **u** i n komponenti vektora **x**, u stacionarnoj točki.

Ako sada uvedemo

$$\lambda^T = -F_{\mathbf{x}} \cdot \mathbf{f}_{\mathbf{x}}^{-1}, \tag{2.10}$$

jednadžba (2.9) poprima oblik

$$F + \lambda^T \cdot \mathbf{f} = \mathbf{0}, \tag{2.11}$$

dok iz (2.10) slijedi

$$F_{\mathbf{x}} + \lambda^T \cdot \mathbf{f}_{\mathbf{x}} = \mathbf{0}. \tag{2.12}$$

Gornje dvije jednadžbe predstavljaju alternativni prikaz uvjeta (2.9), s tom razlikom što m jednadžbi (2.11), n jednadžbi (2.12), te n jednadžbi (2.4) određuju m komponenti vektora $\mathbf{u}$ i n komponenti vektora $\mathbf{x}$, te n komponenti vektora λ, u stacionarnoj točki. Ako sada uvedemo funkciju

$$H(\mathbf{x},\mathbf{u},\lambda) = F(\mathbf{x},\mathbf{u}) + \lambda^T \cdot \mathbf{f}(\mathbf{x},\mathbf{u}), \tag{2.13}$$

vidimo da nužne uvjete minimuma funkcije (2.3) uz ograničenja (2.4), predstavljene jednadžbama (2.11), (2.12) i (2.4), možemo dobiti iz uvjeta $dH = 0$, tretirajući vektore $\mathbf{x}$, $\mathbf{u}$, λ, kao međusobno nezavisne varijable funkcije (2.13). Imamo

$$dH = \frac{\partial H}{\partial \mathbf{x}} \cdot d\mathbf{x} + \frac{\partial H}{\partial \mathbf{u}} \cdot d\mathbf{u} + \frac{\partial H}{\partial \lambda} \cdot d\lambda, \tag{2.14}$$

odnosno, zbog uvjeta $dH = 0$ za proizvoljne $d\mathbf{x}$, $d\mathbf{u}$, $d\lambda$, dobivamo slijedeće uvjete

$$\frac{\partial H}{\partial \mathbf{x}} = F_{\mathbf{x}} + \lambda^T \cdot \mathbf{f}_{\mathbf{x}} = \mathbf{0}, \tag{2.15}$$

$$\frac{\partial H}{\partial \mathbf{u}} = F_{\mathbf{u}} + \lambda^T \cdot \mathbf{f}_{\mathbf{u}} = \mathbf{0}, \tag{2.16}$$

$$\frac{\partial H}{\partial \lambda} = \mathbf{f}(\mathbf{x},\mathbf{u}) = \mathbf{0}. \tag{2.17}$$

Komponente vektora λ, zovemo Lagrangeovi multiplikatori, dok funkciju $H(\mathbf{x},\mathbf{u},\lambda)$ zovemo Hamiltonian ili Hamiltonova funkcija.

Na ovaj način smo problem minimizacije funkcije (2.3) uz ograničenja (2.4), kojima su varijable $\mathbf{x}$ ovisne o varijablama $\mathbf{u}$, sveli na problem minimizacije, bez ograničenja, funkcije (2.13), tretirajući vektore $\mathbf{x}$, $\mathbf{u}$, λ, kao međusobno nezavisne varijable. Iz izraza (2.17) uvijek proizlaze zadane jednadžbe ograničenja, tako da se često u literaturi gleda varijacija Hamiltoniana samo u funkciji od $d\mathbf{x}$, $d\mathbf{u}$, tretirajući λ kao konstantu, te se na taj način dobivenim izrazima (2.15) i (2.16) doda jednadžba (2.4).

2.2 Optimizacija diskretnih dinamičkih sustava

Optimizacija diskretnih dinamičkih sustava spada također u probleme parametarske optimizacije. Razmotrimo diskretni dinamički sustav opisan nelinearnom diferencijskom jednadžbom

$$\mathbf{x}(i+1) = \mathbf{f}^i\big(\mathbf{x}(i),\mathbf{u}(i)\big), \qquad \mathbf{x}(0) = \mathbf{x}_0, \qquad i = 0,1,\ldots,N-1, \tag{2.18}$$

koja predstavlja skup od nN jednadžbi. Gornjim sustavom jednadžbi je niz n-dimenzionalnih vektora stanja $\mathbf{x}(i)$ određen nizom m-dimenzionalnih upravljačkih vektora $\mathbf{u}(i)$. Funkciju cilja definiramo u obliku

$$J = \phi\big(\mathbf{x}(N)\big) + \sum_{i=0}^{N-1} F\big(\mathbf{x}(i),\mathbf{u}(i)\big). \tag{2.19}$$

Problem se sastoji u pronalaženju niza upravljačkih vektora $\mathbf{u}(i)$, koji minimiziraju gornju funkciju cilja i pri tome zadovoljavaju sustav jednadžbi (2.18). Dodavanjem sustava jednadžbi (2.18) sa nizom Lagrangeovih multiplikatora u jednadžbu (2.19), dobivamo

$$J = \phi\big(\mathbf{x}(N)\big) + \sum_{i=0}^{N-1}\Big[F\big(\mathbf{x}(i),\mathbf{u}(i)\big) + \lambda^T(i+1)\Big(\mathbf{f}^i\big(\mathbf{x}(i),\mathbf{u}(i)\big) - \mathbf{x}(i+1)\Big)\Big]. \tag{2.20}$$

Definiramo skalarni niz funkcija (Hamiltonijan)

$$H^i = F\big(\mathbf{x}(i),\mathbf{u}(i)\big) + \lambda^T(i+1)\mathbf{f}^i\big(\mathbf{x}(i),\mathbf{u}(i)\big). \tag{2.21}$$

Promjenom indeksa u sumi izraza (2.20), dobivamo

$$J = \phi\big(\mathbf{x}(N)\big) - \lambda^T(N)\cdot\mathbf{x}(N) + H^0 + \sum_{i=1}^{N-1}\Big[H^i - \lambda^T(i)\cdot\mathbf{x}(i)\Big]. \tag{2.22}$$

Nužan uvjet minimuma funkcije cilja (2.22) je $dJ = 0$, pri proizvoljnoj promjeni $d\mathbf{u}(i)$, $d\mathbf{x}(i)$. Imamo

$$dJ = \left[\frac{\partial\phi}{\partial\mathbf{x}(N)} - \lambda^T(N)\right]\cdot d\mathbf{x}(N) + \sum_{i=1}^{N-1}\left\{\left[\frac{\partial H^i}{\partial\mathbf{x}(i)} - \lambda^T(i)\right]\cdot d\mathbf{x}(i) + \frac{\partial H^i}{\partial\mathbf{u}(i)}\cdot d\mathbf{u}(i)\right\} +$$
$$+ \frac{\partial H^0}{\partial\mathbf{x}(0)}\cdot d\mathbf{x}(0) + \frac{\partial H^0}{\partial\mathbf{u}(0)}\cdot d\mathbf{u}(0)$$

iz čega slijedi

$$\frac{\partial H^i}{\partial\mathbf{x}(i)} - \lambda^T(i) = \mathbf{0},$$

odnosno

$$\lambda^T(i) = \frac{\partial F^i}{\partial\mathbf{x}(i)} + \lambda^T(i+1)\cdot\frac{\partial\mathbf{f}^i}{\partial\mathbf{x}(i)}, \qquad i = 0,1,\ldots,N-1, \tag{2.23}$$

sa rubnim uvjetom

$$\lambda^T(N) = \frac{\partial\phi}{\partial\mathbf{x}(N)}. \tag{2.24}$$

Slijedeći uvjet koji mora biti zadovoljen je

$$\frac{\partial H^i}{\partial\mathbf{u}(i)} = \mathbf{0}, \qquad i = 0,1,\ldots,N-1. \tag{2.25}$$

Dakle, da bi niz upravljačkih vektora $\mathbf{u}(i)$, predstavljao stacionarnu točku funkcije cilja (2.19) uz zadana ograničenja (2.18), mora biti zadovoljen slijedeći sustav jednadžbi

$$\mathbf{x}(i+1) = \mathbf{f}^i\big(\mathbf{x}(i),\mathbf{u}(i)\big), \tag{2.26}$$

$$\lambda\ \ (i) = \frac{\partial F^i}{\partial\mathbf{x}(i)} + \lambda\ \ (i+1)\cdot\frac{\partial\mathbf{f}^i}{\partial\mathbf{x}(i)}, \tag{2.27}$$

$$\frac{\partial F^i}{\partial \mathbf{u}(i)} + \lambda^T(i+1) \cdot \frac{\partial \mathbf{f}^i}{\partial \mathbf{u}(i)} = \mathbf{0}, \tag{2.28}$$

za $i = 0,1,\ldots,N-1$, uz zadane početne i rubne uvjete

$$\mathbf{x}(0) = \mathbf{x}_0, \qquad \lambda^T(N) = \frac{\partial \phi}{\partial \mathbf{x}(N)}. \tag{2.29}$$

Diferencijske jednadžbe (2.26) i (2.27) su međusobno spregnute, budući da $\mathbf{u}(i)$ ovisi o $\lambda(i)$ preko (2.29). Jednadžbe (2.26)-(2.28) predstavljaju sustav od $(2n+m)N$ jednadžbi s isto toliko nepoznanica: $\mathbf{x}(1),\ldots,\mathbf{x}(N)$, $\mathbf{u}(0),\ldots,\mathbf{u}(N-1)$, $\lambda(0),\ldots,\lambda(N-1)$.

2.3 Optimizacija kontinuiranih dinamičkih sustava

Optimizacijski problemi za kontinuirane dinamičke sustave spadaju u varijacijski račun. Oni se također mogu razmatrati kao granični slučaj optimizacijskih problema za diskretne dinamičke sustave (period diskretizacije dovoljno mali u usporedbi s ukupnim vremenskim intervalom). Obrnuti slučaj je danas mnogo češći, kontinuirani sustavi aproksimiraju se diskretnim, zbog mogućnosti obrade na digitalnim računalima.

Razmotrimo slijedeći sustav nelinearnih diferencijalnih jednadžbi

$$\dot{\mathbf{x}}(t) = \mathbf{f}\big(\mathbf{x}(t), \mathbf{u}(t), t\big), \qquad \mathbf{x}(t_0) = \mathbf{x}_0, \qquad t_0 \le t \le t_f, \tag{2.30}$$

gdje je $\mathbf{x}(t)$ n-dimenzionalna vektorska funkcija (vektor stanja) koja je određena sa $\mathbf{u}(t)$, m-dimenzionalnom vektorskom funkcijom (vektor upravljanja). Razmotrimo funkciju cilja u obliku

$$J = \phi\big(\mathbf{x}(t_f), t_f\big) + \int_{t_0}^{t_f} F\big(\mathbf{x}(t), \mathbf{u}(t), t\big)\, dt, \tag{2.31}$$

što predstavlja kontinuirani oblik diskretne funkcije cilja (2.19). Poblem optimalnog upravljanja sa funkcijom cilja oblika (2.31) naziva se Bolzin problem. Problem se sastoji u pronalaženju vektorske funkcije $\mathbf{u}(t)$ koja minimizira funkciju cilja (2.31) pri čemu je povezanost između $\mathbf{x}(t)$ i $\mathbf{u}(t)$ dana sustavom diferencijalnih jednadžbi (2.30). Analogno kao u diskretnom slučaju, koristimo metodu Lagrangeovih multiplikatora da bi ubacili sustav diferencijalnih jednadžbi u funkciju cilja (2.31)

$$J = \phi\big(\mathbf{x}(t_f), t_f\big) + \int_{t_0}^{t_f} \Big[F\big(\mathbf{x}(t), \mathbf{u}(t), t\big) + \lambda^T(t) \cdot \big(\mathbf{f}\big(\mathbf{x}(t), \mathbf{u}(t), t\big) - \dot{\mathbf{x}}(t)\big) \Big] dt, \tag{2.32}$$

što je analogno izrazu (2.20). Definiramo Hamiltonian

$$H\big(\mathbf{x}(t), \mathbf{u}(t), \lambda(t), t\big) = F\big(\mathbf{x}(t), \mathbf{u}(t), t\big) + \lambda^T(t) \cdot \mathbf{f}\big(\mathbf{x}(t), \mathbf{u}(t), t\big). \tag{2.32}$$

Sada imamo

$$J = \phi\left(\mathbf{x}(t_f), t_f\right) + \int_{t_0}^{t_f} \left[H\left(\mathbf{x}(t), \mathbf{u}(t), \lambda(t), t\right) - \lambda^T(t) \cdot \dot{\mathbf{x}}(t) \right] dt \,. \tag{2.33}$$

Parcijalnom integracijom drugog člana podintegralnog izraza u (2.33), dobivamo

$$J = \phi\left(\mathbf{x}(t_f), t_f\right) - \lambda^T(t_f) \cdot \mathbf{x}(t_f) + \lambda^T(t_0) \cdot \mathbf{x}(t_0) + \int_{t_0}^{t_f} \left[H\left(\mathbf{x}(t), \mathbf{u}(t), \lambda(t), t\right) + \dot{\lambda}^T(t) \cdot \mathbf{x}(t) \right] dt \,.$$

Sada uzimamo prvu varijaciju funkcije cilja, δJ, u ovisnosti o varijaciji vektora upravljanja i vektora stanja, oko optimalnog vektora upravljanja i optimalnog vektora stanja. Imamo

$$\delta J = \left[\left(\frac{\partial \phi}{\partial \mathbf{x}} - \lambda^T \right) \cdot \delta \mathbf{x} \right]_{t=t_f} + \left[\lambda^T \cdot \delta \mathbf{x} \right]_{t=t_0} + \int_{t_0}^{t_f} \left[\left(\frac{\partial H}{\partial \mathbf{x}} + \dot{\lambda}^T \right) \cdot \delta \mathbf{x} + \frac{\partial H}{\partial \mathbf{u}} \cdot \delta \mathbf{u} \right] dt \,. \tag{2.34}$$

Nužan uvjet minimuma funkcije cilja je iščezavanje prve varijacije, $\delta J = 0$, za proizvoljne varijacije $\delta \mathbf{x}$, $\delta \mathbf{u}$. Dobivamo

$$\dot{\lambda}^T = -\frac{\partial H}{\partial \mathbf{x}} = -\frac{\partial F}{\partial \mathbf{x}} - \lambda^T \cdot \frac{\partial \mathbf{f}}{\partial \mathbf{x}} \,, \tag{2.35}$$

sa rubnim uvjetom

$$\lambda^T(t_f) = \frac{\partial \phi}{\partial \mathbf{x}(t_f)} \,. \tag{2.36}$$

Drugi uvjet koji dobivamo iz (2.34) je

$$\frac{\partial H}{\partial \mathbf{u}} = \mathbf{0} \,, \tag{2.37}$$

za $t_0 \le t \le t_f$. Jednadžbe (2.35), (2.36) i (2.37) poznate su u varijacijskom računu kao Euler-Lagrangeove jednadžbe.

Dakle, da bi našli upravljački vektor $\mathbf{u}(t)$ koji predstavlja stacionarnu vrijednost funkcije cilja (2.31), moraju biti zadovoljene slijedeće jednadžbe

$$\dot{\mathbf{x}} = \mathbf{f}\left(\mathbf{x}, \mathbf{u}, t\right) , \tag{2.38}$$

$$\dot{\lambda}^T = -\frac{\partial F}{\partial \mathbf{x}} - \lambda^T \cdot \frac{\partial \mathbf{f}}{\partial \mathbf{x}} \,, \tag{2.39}$$

$$\frac{\partial F}{\partial \mathbf{u}} + \lambda^T \cdot \frac{\partial \mathbf{f}}{\partial \mathbf{u}} = \mathbf{0} \,, \tag{2.40}$$

sa slijedećim početnim i rubnim uvjetima

$$\mathbf{x}(t_0) = \mathbf{x}_0 \,, \qquad \lambda\ (t_f) = \frac{\partial \phi}{\partial \mathbf{x}(t_f)} \,. \tag{2.41}$$

Vidimo da je gornji sustav jednadžbi analogan sustavu (2.26)-(2-29), s tom razlikom što ovdje imamo sustav od $2n$ diferencijalnih jednadžbi (2.38) i (2.39) s $2n$ rubnih uvjeta

(2.41), te sustav od m algebarskih jednadžbi (2.40) iz kojeg određujemo vektor $\mathbf{u}(t)$, nakon što nađemo $\mathbf{x}(t)$ i $\lambda(t)$.

Pogledajmo sada čemu je jednaka totalna vremenska derivacija Hamiltoniana, za optimalnu vrijednost vektora $\mathbf{u}(t)$ i $\mathbf{x}(t)$

$$\frac{dH}{dt} = \frac{\partial H}{\partial t} + \frac{\partial H}{\partial \mathbf{x}} \cdot \dot{\mathbf{x}} + \frac{\partial H}{\partial \mathbf{u}} \cdot \dot{\mathbf{u}} + \frac{\partial H}{\partial \lambda^T} \cdot \dot{\lambda}^T, \tag{2.42}$$

odnosno

$$\frac{dH}{dt} = \frac{\partial H}{\partial t} + \frac{\partial H}{\partial \mathbf{u}} \cdot \dot{\mathbf{u}} + \left(\frac{\partial H}{\partial \mathbf{x}} + \dot{\lambda}^T \right) \cdot \mathbf{f} \ . \tag{2.42}$$

Zbog (2.35) i (2.37) imamo

$$\frac{dH}{dt} = \frac{\partial H}{\partial t}. \tag{2.43}$$

Ako H (odnosno F i $\mathbf{f}$) nije eksplicitna funkcija vremena, imamo

$$\frac{dH}{dt} = 0, \tag{2.44}$$

što znači da je Hamiltonian konstantan za optimalnu trajektoriju.

Od interesa je pogledati rješenje Bolzinog problema u slučaju kada je zadan vektor stanja u konačnom vremenu t_f. Neka je zadano općenito ograničenje vektora stanja u t_f slijedećom jednadžbom

$$\psi\left(\mathbf{x}(t_f), t_f\right) = \mathbf{0}, \tag{2.45}$$

gdje je ψ q-dimenzionalna vektorska funkcija ($q \le n-1$ ako $F = 0$, ($q \le n$ ako $F \neq 0$). Ovdje ćemo također koristiti metodu Lagrangeovih multiplikatora, te dodajemo jednadžbu (2.45) pomnoženu s q-dimenzionalnim vektorom ν, u izraz (2.32)

$$J = \phi\left(\mathbf{x}(t_f), t_f\right) + \nu^T \cdot \psi\left(\mathbf{x}(t_f), t_f\right) + \int_{t_0}^{t_f} \left[F\left(\mathbf{x}(t), \mathbf{u}(t), t\right) + \lambda^T(t) \cdot \left(\mathbf{f}\left(\mathbf{x}(t), \mathbf{u}(t), t\right) - \dot{\mathbf{x}}(t) \right) \right] dt \, .$$

Ponavljajući analognu proceduru kao u predhodnom odjeljku, dobivamo sustav jednadžbi (2.38)-(2.40) s tom razlikom što rubni uvjet u (2.41) postaje

$$\lambda^T(t_f) = \frac{\partial \phi}{\partial \mathbf{x}(t_f)} + \nu^T \cdot \frac{\partial \psi}{\partial \mathbf{x}(t_f)}. \tag{2.46}$$

Skup parametara ν određujemo tako da zadovoljavaju q jednadžbi (2.45).

2.4 Slučaj sa nespecificiranim konačnim vremenom

U prethodnim podpoglavljima pretpostavljali smo da je konačno vrijeme t_f, specificirano, odnosno konstantno. Sada ćemo razmatrati slučaj kada je indeks performanse funkcija, ne samo vektora upravljanja i vektora stanja, nego i konačnog vremena t_f. Pretpostavljamo da je početno vrijeme i početno stanje određeno. Problem se, dakle, svodi na minimizaciju funkcije cilja

$$J = \phi\left(\mathbf{x}(t_f), t_f\right) + \int_{t_0}^{t_f} F\left(\mathbf{x}(t), \mathbf{u}(t), t\right) dt, \tag{2.47}$$

za sistem opisan sa

$$\dot{\mathbf{x}}(t) = \mathbf{f}\left(\mathbf{x}(t), \mathbf{u}(t), t\right), \qquad \mathbf{x}(t_0) = \mathbf{x}_0, \tag{2.48}$$

gdje je t_0 fiksno i gdje u nespecificiranom konačnom vremenu t_f treba zadovoljiti q jednadžbi za rubne uvjete

$$\psi\left(\mathbf{x}(t_f), t_f\right) = \mathbf{0}. \tag{2.49}$$

Metodom Lagrangeovih multiplikatora funkciji cilja (2.47) dodamo ograničenja (2.48) i (2.49)

$$J = \phi\left(\mathbf{x}(t_f), t_f\right) + \nu^T \cdot \psi\left(\mathbf{x}(t_f), t_f\right) + \int_{t_0}^{t_f} \left[F\left(\mathbf{x}(t), \mathbf{u}(t), t\right) + \lambda^T(t) \cdot \left(\mathbf{f}\left(\mathbf{x}(t), \mathbf{u}(t), t\right) - \dot{\mathbf{x}}(t)\right)\right] dt,$$

te definiramo Hamiltonian

$$H\left(\mathbf{x}(t), \mathbf{u}(t), \lambda(t), t\right) = F\left(\mathbf{x}(t), \mathbf{u}(t), t\right) + \lambda^T(t) \cdot \mathbf{f}\left(\mathbf{x}(t), \mathbf{u}(t), t\right). \tag{2.50}$$

Parcijalnom integracijom nove funkcije cilja, dobivamo

$$\begin{aligned} J = {} & \phi\left(\mathbf{x}(t_f), t_f\right) + \nu^T \cdot \psi\left(\mathbf{x}(t_f), t_f\right) - \lambda^T(t_f) \cdot \mathbf{x}(t_f) + \lambda^T(t_0) \cdot \mathbf{x}(t_0) + \\ & + \int_{t_0}^{t_f} \left[H\left(\mathbf{x}(t), \mathbf{u}(t), \lambda(t), t\right) + \dot{\lambda}^T(t) \cdot \mathbf{x}(t)\right] dt \end{aligned}. \tag{2.51}$$

Sada uvodimo prve varijacije oko optimalnih vrijednosti

$$\mathbf{x}(t) = \hat{\mathbf{x}}(t) + \delta\mathbf{x}(t), \qquad \mathbf{u}(t) = \hat{\mathbf{u}}(t) + \delta\mathbf{u}(t) \qquad t_f = \hat{t}_f + \delta t_f, \tag{2.52}$$

i formiramo razliku

$$\Delta J = J\left(\mathbf{x}(t), \mathbf{u}(t), t_f\right) - J\left(\hat{\mathbf{x}}(t), \hat{\mathbf{u}}(t), \hat{t}_f\right), \tag{2.53}$$

od koje ostavljamo samo linearne članove, odnosno prvu varijaciju δJ. Prvo ćemo pogledati čemu je jednaka varijacija vektora stanja u konačnom vremenu t_f

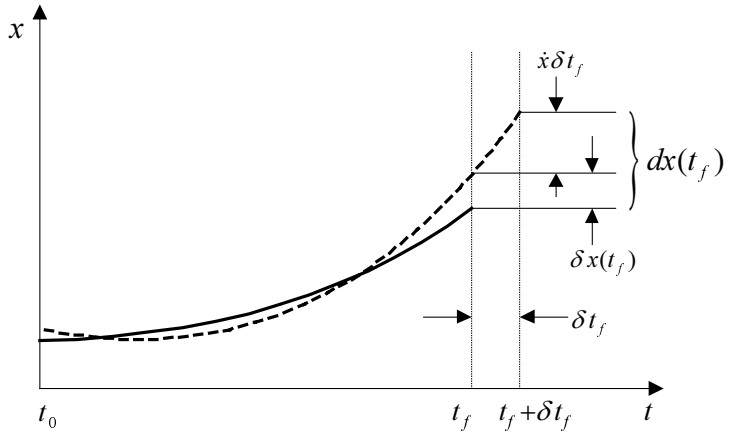
$$d\mathbf{x}(t_f) = \mathbf{x}(t_f + \delta t_f) + \delta\mathbf{x}(t_f + \delta t_f) - \mathbf{x}(t_f), \tag{2.54}$$

odnosno, nakon razvoja u Taylorov red, imamo

$$d\,\mathbf{x}(t_f) = \mathbf{x}(t_f) + \dot{\mathbf{x}}(t_f)\delta t_f + \delta\,\mathbf{x}(t_f) + \delta\,\dot{\mathbf{x}}(t_f)\delta t_f - \mathbf{x}(t_f),$$

te zanemarenjem kvadratnog člana $\delta\,\dot{\mathbf{x}}(t_f)\delta t_f$, dobivamo konačni oblik (slika 2.1)

$$d\,\mathbf{x}(t_f) = \delta\,\mathbf{x}(t_f) + \dot{\mathbf{x}}(t_f)\delta t_f. \tag{2.55}$$



Slika 2.1 *Veza između* $d\,\mathbf{x}(t_f)$, $\delta\,\mathbf{x}(t_f)$ i δt_f.

Varijacija $d\,\mathbf{x}(t_f)$, zove se *neizohrona* ili *asinhrona varijacija.* Sada ćemo pogledati čemu je jednako δJ

$$\delta J = \delta\Theta\Big(\mathbf{x}(t_f), t_f\Big) + \delta\,\omega\Big(\mathbf{x}(t_f), t_f\Big) + \delta\, I\Big(\mathbf{x}(t_f), \mathbf{u}(t_f), t_f\Big), \tag{2.56}$$

gdje je

$$\Theta\Big(\mathbf{x}(t_f), t_f\Big) = \phi\Big(\mathbf{x}(t_f), t_f\Big) + \nu^T \cdot \psi\Big(\mathbf{x}(t_f), t_f\Big),$$

$$\omega\Big(\mathbf{x}(t_f), t_f\Big) = -\lambda^T(t_f)\cdot\mathbf{x}(t_f),$$

$$I\Big(\mathbf{x}(t), \mathbf{u}(t), t\Big) = \int_{t_0}^{t_f}\Big[H\Big(\mathbf{x}(t), \mathbf{u}(t), \lambda(t),\, t\Big) + \dot{\lambda}^T(t)\cdot\mathbf{x}(t)\Big]dt.$$

Pogledajmo prvo

$$\begin{aligned}\delta\Theta\Big(\mathbf{x}(t_f), t_f\Big) &= \Theta\Big(\mathbf{x}(t_f + \delta t_f) + \delta\,\mathbf{x}(t_f + \delta t_f), t_f + \delta t_f\Big) - \Theta\Big(\mathbf{x}(t_f), t_f\Big) = \\ &= \Theta\Big(\mathbf{x}(t_f) + \dot{\mathbf{x}}(t_f)\delta t_f + \delta\,\mathbf{x}(t_f), t_f + \delta t_f\Big) - \Theta\Big(\mathbf{x}(t_f), t_f\Big) = \\ &= \Theta\Big(\mathbf{x}(t_f) + d\,\mathbf{x}(t_f), t_f + \delta t_f\Big) - \Theta\Big(\mathbf{x}(t_f), t_f\Big)\end{aligned}$$

odnosno

$$\delta\Theta\Big(\mathbf{x}(t_f), t_f\Big) = \frac{\partial\Theta}{\partial\,\mathbf{x}(t_f)}\,d\,\mathbf{x}(t_f) + \frac{\partial\Theta}{\partial t_f}\delta t_f. \tag{2.57}$$

Nadalje, imamo

$$\begin{aligned}\delta\omega\left(\mathbf{x}(t_f),t_f\right) &= \omega\left(\mathbf{x}(t_f+\delta t_f)+\delta\mathbf{x}(t_f+\delta t_f),t_f+\delta t_f\right)-\omega\left(\mathbf{x}(t_f),t_f\right)= \\ &= -\lambda^T(t_f+\delta t_f)\cdot\left(\mathbf{x}(t_f+\delta t_f)+\delta\mathbf{x}(t_f+\delta t_f)\right)+\lambda^T(t_f)\mathbf{x}(t_f)= \\ &= -\left(\lambda^T(t_f)+\dot{\lambda}^T(t_f)\delta t_f\right)\left(\mathbf{x}(t_f)+d\,\mathbf{x}(t_f)\right)+\lambda^T(t_f)\mathbf{x}(t_f)\end{aligned}$$

Nakon zanemarenja članova drugog reda imamo

$$\delta\omega\left(\mathbf{x}(t_f),t_f\right) = -\dot{\lambda}^T(t_f)\cdot\mathbf{x}(t_f)\delta t_f-\lambda^T(t_f)\cdot d\,\mathbf{x}(t_f). \tag{2.58}$$

Na kraju, imamo

$$\begin{aligned}\delta I\left(\mathbf{x}(t),\mathbf{u}(t),t\right) = &\int_{t_0}^{t_f+\delta t_f}\left[H\left(\mathbf{x}(t)+\delta\mathbf{x}(t),\mathbf{u}(t)+\delta\mathbf{u}(t),\lambda(t),\,t\right)+\dot{\lambda}^T(t)\cdot\left(\mathbf{x}(t)+\delta\mathbf{x}(t)\right)\right]dt- \\ &-\int_{t_0}^{t_f}\left[H\left(\mathbf{x}(t),\mathbf{u}(t),\lambda(t),\,t\right)+\dot{\lambda}^T(t)\cdot\mathbf{x}(t)\right]dt\end{aligned}$$

odnosno

$$\begin{aligned}\delta I\left(\mathbf{x}(t),\mathbf{u}(t),t\right) = &H\left(\mathbf{x}(t_f),\mathbf{u}(t_f),\lambda(t_f),\,t_f\right)\delta t_f+\dot{\lambda}^T(t_f)\cdot\mathbf{x}(t_f)+ \\ &+\int_{t_0}^{t_f}\left[\left(\frac{\partial H}{\partial\mathbf{x}(t)}+\lambda^T(t)\mathbf{x}(t)\right)\cdot\delta\mathbf{x}(t)+\frac{\partial H}{\partial\mathbf{u}(t)}\right]dt\end{aligned}. \tag{2.59}$$

Sada stavimo (2.57)-(2.59) u (2.56) i dobivamo

$$\begin{aligned}\delta J = &\left(H\left(\mathbf{x}(t_f),\mathbf{u}(t_f),\lambda(t_f),\,t_f\right)+\frac{\partial\Theta}{\partial t_f}\right)\delta t_f+\left(\frac{\partial\Theta}{\partial\mathbf{x}(t_f)}-\lambda^T(t_f)\right)\cdot d\,\mathbf{x}(t_f)+ \\ &+\int_{t_0}^{t_f}\left[\left(\frac{\partial H}{\partial\mathbf{x}(t)}+\dot{\lambda}^T(t)\right)\cdot\delta\mathbf{x}(t)+\frac{\partial H}{\partial\mathbf{u}(t)}\cdot\delta\mathbf{u}(t)\right]dt\end{aligned} \tag{2.60}$$

Da bi dobili nužne uvjete za minimum funkcije cilja, mora biti $\delta J = 0$, pa iz (2.60) proizlaze jednadžbe koje određuju optimalni vektor upravljanja

$$\frac{\partial H}{\partial\lambda^T} = \dot{\mathbf{x}} = \mathbf{f}\left(\mathbf{x},\mathbf{u},t\right), \tag{2.61}$$

$$\frac{\partial H}{\partial\mathbf{x}} = -\dot{\lambda}^T = \frac{\partial F}{\partial\mathbf{x}}+\lambda^T\cdot\frac{\partial\mathbf{f}}{\partial\mathbf{x}}, \tag{2.62}$$

$$\frac{\partial H}{\partial\mathbf{u}} = \mathbf{0} = \frac{\partial F}{\partial\mathbf{u}}+\lambda^T\cdot\frac{\partial\mathbf{f}}{\partial\mathbf{u}}, \tag{2.63}$$

sa početnim i rubnim uvjetima

$$\mathbf{x}(t_0) = \mathbf{x}_0, \qquad \lambda^T(t_f) = \frac{\partial\phi}{\partial\mathbf{x}(t_f)}+\nu^T\cdot\frac{\partial\psi}{\partial\mathbf{x}(t_f)}, \qquad \psi\left(\mathbf{x}(t_f),t_f\right) = \mathbf{0}, \tag{2.64}$$

i

$$H\left(\mathbf{x}(t_f),\mathbf{u}(t_f),\lambda(t_f),t_f\right)+\frac{\partial\phi}{\partial t_f}+\nu\cdot\frac{\partial\psi}{\partial t_f}=0\,. \tag{2.65}$$

Vidimo da su jednadžbe (2.61)-(2.64) identične jednadžbama za Bolzin problem sa složenim rubnim uvjetima (2.49), s tom razlikom da ovdje imamo dodatni uvjet iz kojeg određujemo nespecificirano konačno vrijeme t_f.

Posebna klasa problema optimalnog upravljanja s nespecificiranim konačnim vremenom je problem najmanjeg vremena (*minimum time problem*) za koji vrijedi

$$\phi\left(\mathbf{x}(t_f),t_f\right)=t_f\,,\qquad F\left(\mathbf{x}(t),\mathbf{u}(t),t\right)=0\,, \tag{2.66}$$

sa rubnim uvjetom

$$\psi\left(\mathbf{x}(t_f),t_f\right)=\mathbf{x}(t_f)=\mathbf{0}\,. \tag{2.67}$$

Hamiltonian (2.50) postaje

$$H\left(\mathbf{x}(t),\mathbf{u}(t),\lambda(t),t\right)=\lambda^T(t)\cdot\mathbf{f}\left(\mathbf{x}(t),\mathbf{u}(t),t\right). \tag{2.68}$$

U tom slučaju jednadžbe (2.61)-(2.65) poprimaju oblik

$$\dot{\mathbf{x}}=\mathbf{f}\left(\mathbf{x},\mathbf{u},t\right), \tag{2.69}$$

$$\dot{\lambda}^T=-\lambda^T\cdot\frac{\partial\,\mathbf{f}}{\partial\mathbf{x}}, \tag{2.70}$$

$$\lambda^T\cdot\frac{\partial\,\mathbf{f}}{\partial\mathbf{u}}=\mathbf{0}, \tag{2.71}$$

sa početnim i rubnim uvjetima

$$\mathbf{x}(t_0)=\mathbf{x}_0\,,\qquad \lambda(t_f)=\nu\,,\qquad \mathbf{x}(t_f)=\mathbf{0}\,, \tag{2.72}$$

i

$$H\left(\mathbf{x}(t_f),\mathbf{u}(t_f),\lambda(t_f),t_f\right)=-1\,. \tag{2.73}$$

Ako Hamiltonian nije eksplicitna funkcija vremena, tada vrijedi jednakost (2.44), što znači da je Hamiltonian konstantan tokom cijelog vremenskog intervala, pa vrijedi

$$H\left(\mathbf{x}(t),\mathbf{u}(t),\lambda(t),t\right)=H\left(\mathbf{x}(t),\mathbf{u}(t),\lambda(t)\right)=-1\,, \tag{2.74}$$

za $t_0\le t\le t_f$.

Međutim, pored uvjeta (2.69)-(2.73), jedinstveno rješenje ovog problema postoji onda i samo onda ako vektor upravljanja ima ograničenja.

Iste uvjete bi dobili da smo umjesto (2.66) stavili

$$\phi\left(\mathbf{x}(t_f),t_f\right)=0\,,\qquad F\left(\mathbf{x}(t),\mathbf{u}(t),t\right)=1\,, \tag{2.75}$$

s tim da bi Hamiltonian, u tom slučaju, bio

$$H\left(\mathbf{x}(t),\mathbf{u}(t),\lambda(t),t\right)=1+\lambda^{T}(t)\cdot\mathbf{f}\left(\mathbf{x}(t),\mathbf{u}(t),t\right), \tag{2.76}$$

uvjet (2.73) postao bi

$$H\left(\mathbf{x}(t_f),\mathbf{u}(t_f),\lambda(t_f),t_f\right)=0, \tag{2.77}$$

dok bi uvjet (2.74) postao

$$H\left(\mathbf{x}(t),\mathbf{u}(t),\lambda(t),t\right)=H\left(\mathbf{x}(t),\mathbf{u}(t),\lambda(t)\right)=0, \tag{2.78}$$

za $t_0 \le t \le t$.

2.5 Optimalno upravljanje s ograničenjima vektora upravljanja

Do sada smo razmatrali probleme optimalnog upravljanja bez ograničenja na vektor upravljanja. S druge strane, rješenja (optimalni vektor upravljanja i stanja) su bila neprekidna u cijelom intervalu $t_0 \le t \le t$. Ako je vektor upravljanja ograničen skupom nejednadžbi

$$\mathbf{g}\left(\mathbf{u}(t),t\right)\ge\mathbf{0}, \tag{2.79}$$

gdje je $\mathbf{g}$ *r*-komponentna vektorska funkcija, tada je jasno da ne možemo više koristiti nužan uvjet optimalnosti rješenja u obliku

$$\frac{\partial H}{\partial\mathbf{u}}=\mathbf{0}, \tag{2.80}$$

zbog toga što rješenje dobiveno izrazom (2.80) može kršiti ograničenja (2.79).

Također, moguće je da čak i u slučaju kada nisu zadana ograničenja vektora upravljanja, Hamiltonian ovisi linearno o vektoru upravljanja i zbog toga parcijalne derivacije (2.80) ne sadrže komponente vektora upravljanja što onemogućava nalaženje rješenja. Takav slučaj naziva se *singularno optimalno upravljanje*.

Nadalje, postoji mogućnost da jednadžba (2.80) ima višestruke korijene. Postavlja se pitanje, koje od tih rješenja izabrati kao optimalno rješenje.

Da bi rješili gore navedene probleme, moramo imati općenitiji nužni uvjet optimalnosti od (2.80). Takav općeniti nužan uvjet optimalnosti daje Pontryaginov princip maksimuma (odnosno minimuma, ako se radi o minimizaciji), koji glasi: U slučaju svih singularnih i neregularnih situacija uključujući i ograničenja vektora upravljanja, izbor komponenti vektora upravljanja mora biti takav da maksimizira (odnosno minimizira) vrijednost Hamiltoniana (2.50). Također, kao i do sada, vrijede jednadžbe

$$\dot{\mathbf{x}}=\frac{\partial H}{\partial\lambda^{T}},\qquad\qquad \dot{\lambda}^{T}=-\frac{\partial H}{\partial\mathbf{x}}, \tag{2.81}$$

sa odgovarajućim rubnim uvjetima.

Pri tome se dopušta mogućnost da komponente vektora upravljanja ne moraju biti neprekidne funkcije vremena, već mogu biti prekidne funkcije u konačnom broju vremenskih trenutaka unutar intervala $t_0 \le t \le t$ (po dijelovima neprekidne funkcije).

Ako vektor upravljanja nije ograničen, tada iz Pontryaginovog principa direktno proizlazi uvjet (2.80). Strogi dokaz principa maksimuma može se naći u monografiji [1].

Rješavanje problema primjenom principa maksimuma može u nekim slučajevima da se bitno pojednostavi prebacivanjem skupa nejednadžbi (2.79) u skup jednadžbi uvođenjem dodatnih varijabli

$$\dot{\mathbf{z}}^2 = \mathbf{g}\big(\mathbf{u}(t), t\big), \tag{2.82}$$

ili

$$\mathbf{y}^2 = \mathbf{g}\big(\mathbf{u}(t), t\big). \tag{2.83}$$

Dodatne varijable pojavljuju se u kvadratnom obliku jer je tada desna strana gornjih jednadžbi uvijek veća ili jednaka nuli, čime je zadovoljena nejednadžba (2.79).

Sada kada smo ograničenja sveli na oblik jednakosti, možemo primjeniti metodu Lagrangeovih multiplikatora ubacujući jednakosti (2.83) u funkciju cilja

$$\begin{aligned} J = {} & \phi\big(\mathbf{x}(t_f), t_f\big) + \nu^T \cdot \psi\big(\mathbf{x}(t_f), t_f\big) + \\ & + \int_{t_0}^{t_f} \Big[H\big(\mathbf{x}(t), \mathbf{u}(t), \lambda^T(t), t\big) - \lambda^T(t) \cdot \dot{\mathbf{x}}(t) + \gamma^T(t) \cdot \big(\mathbf{g}\big(\mathbf{u}(t), t\big) - \mathbf{y}^2(t)\big) \Big] dt \end{aligned} \tag{2.84}$$

Ponavljajući proceduru traženja prve varijacije funkcije cilja, dobivamo slijedeće uvjete koji moraju biti zadovoljeni da bi prva varijacija iščezavala, $\delta J = 0$,

$$\frac{\partial H}{\partial \lambda^T} = \dot{\mathbf{x}} = \mathbf{f}\big(\mathbf{x}, \mathbf{u}, t\big), \qquad \frac{\partial H}{\partial \gamma^T} = \mathbf{y}^2 = \mathbf{g}\big(\mathbf{u}, t\big), \tag{2.85}$$

$$\frac{\partial H}{\partial \mathbf{x}} = -\dot{\lambda}^T = \frac{\partial F}{\partial \mathbf{x}} + \lambda^T \cdot \frac{\partial \mathbf{f}}{\partial \mathbf{x}}, \tag{2.86}$$

$$\gamma_i y_i = 0, \qquad i = 1,...,r, \tag{2.87}$$

$$\frac{\partial F}{\partial \mathbf{u}} + \lambda^T \cdot \frac{\partial \mathbf{f}}{\partial \mathbf{u}} + \gamma^T \cdot \frac{\partial \mathbf{g}}{\partial \mathbf{u}} = \mathbf{0}, \tag{2.88}$$

sa početnim i rubnim uvjetima

$$\mathbf{x}(t_0) = \mathbf{x}_0, \qquad \lambda^T(t_f) = \frac{\partial \phi}{\partial \mathbf{x}(t_f)} + \nu^T \cdot \frac{\partial \psi}{\partial \mathbf{x}(t_f)}, \qquad \psi\big(\mathbf{x}(t_f), t_f\big) = \mathbf{0}, \tag{2.89}$$

i

$$H\big(\mathbf{x}(t_f), \mathbf{u}(t_f), \lambda(t_f), t_f\big) + \frac{\partial \phi}{\partial t_f} + \nu \cdot \frac{\partial \psi}{\partial t_f} = 0. \tag{2.90}$$

Jednadžbama (2.85)-(2.90) definirano je $2n+2r+m+q+1$ jednadžbi za određivanje veličina $\mathbf{x}(t)$, $\lambda(t)$, $\mathbf{y}(t)$, $\gamma(t)$, $\mathbf{u}(t)$, ν, t_f.

2.6 Optimalno upravljanje s ograničenjima vektora stanja

Sada ćemo proširiti razmatranja iz prethodnog podpoglavlja uključujući ograničenja na vektor stanja, koja možemo prikazati sustavom nejednadžbi

$$\mathbf{h}\big(\mathbf{x}(t), t\big) \geq \mathbf{0}, \tag{2.91}$$

gdje je $\mathbf{h}$ s-dimenzionalna vektorska funkcija. Gornje nejednadžbe definiraju prostor dozvoljenih stanja $G \subset R^n$. Gornji izraz u slučaju jednakosti predstavlja jednadžbu za graničnu plohu prostora G, dok je u slučaju stroge nejednakosti definiran otvoreni podprostor prostora G. Pontryaginov princip maksimuma može se primjeniti za određivanje onog dijela optimalne trajektorije koji pripada otvorenom podprostoru prostora G, dok dio trajektorije koji se nalazi na graničnoj plohi prostora G mora zadovoljavati poseban skup uvjeta, uključujući i uvjete skoka u karakterističnim točkama trajektorije koji moraju biti zadovoljeni zbog diskontinuiranosti trajektorije u faznom prostoru. Detaljan prikaz ovih uvjeta može se naći u [9]. Ovdje ćemo se ograničiti na tretiranje gornjeg problema svođenjem sustava nejednadžbi (2.91) na sustav jednadžbi uvođenjem dodatnih varijabli.

Postoji nekoliko načina na koje možemo gornja ograničenja tipa nejednadžbi prebaciti u ograničenja tipa jednadžbi. Definirajmo novu varijablu x_{n+1} sa

$$\dot{x}_{n+1} = f_{n+1} = \sum_{i=1}^{s} h_i^{\,2}(\mathbf{x},t) H(h_i), \tag{2.92}$$

gdje je $H\big(h_i(\mathbf{x},t)\big)$ modificirana Heavisideova step funkcija definirana kao

$$H\big(h_i(\mathbf{x},t)\big) = \begin{cases} 0 \;; & h_i(\mathbf{x},t) \geq 0 \\ K_i \;; & h_i(\mathbf{x},t) < 0 \end{cases} \tag{2.93}$$

gdje je $K_i > 0$, $i = 1,...,s$, i početni uvjet je

$$\dot{x}_{n+1}(t_0) = 0. \tag{2.94}$$

Uz ovakvu definiciju dodatne varijable x_{n+1}, vidimo da je $\dot{x}_{n+1}(t_f)$ direktna mjera kršenja ograničenja, tako da je zahtjev koji postavljamo na tu varijablu

$$\dot{x}_{n+1}(t_f) = 0, \tag{2.95}$$

što nam garantira da su sva ograničenja zadovoljena.

Modifikacija gore navedenog pristupa je da uvedemo s pomoćnih varijabli i isto toliko ograničenja tipa jednakosti

$$\dot{x}_{n+i} = h_i^{\,2}(\mathbf{x},t) H(h_i), \qquad \dot{x}_{n+i}(t_0) = 0, \qquad i = 1,...,s, \tag{2.96}$$

koja dodamo funkciji cilja

$$\hat{J} = J + \sum_{i=1}^{s} x_{n+i}(t_f)\,. \tag{2.97}$$

Pošto je minimalna vrijednost sume vrijednosti pomoćnih varijabli u t_f jednaka nuli, vidimo da minimizacijom gornje funkcije cilja ujedno zadovoljavamo i ograničenja (2.91).

Pošto smo problem sveli na ograničenja tipa jednakosti, možemo primjeniti metodu Lagrangeovih multiplikatora. Podintegralna funkcija ili Lagrangian, sada poprima oblik, za slučaj (2.92),

$$L = H - \lambda^T \cdot \dot{\mathbf{x}} + \gamma^T \cdot \left(\mathbf{g} - \mathbf{y}^2\right) + \lambda_{n+1}\left(f_{n+1} - \dot{x}_{n+1}\right). \tag{2.98}$$

Euler-Lagrangeove jednadžbe, u tom slučaju, poprimaju isti oblik kao (2.85)-(2.90) s tom razlikom da se (2.86) modificira s

$$\dot{\lambda}^T = -\frac{\partial F}{\partial \mathbf{x}} - \lambda^T \cdot \frac{\partial \mathbf{f}}{\partial \mathbf{x}} - \lambda_{n+1}\frac{\partial f_{n+1}}{\partial \mathbf{x}}, \tag{2.99}$$

i imamo još

$$\dot{\lambda}_{n+1}(t) = 0\,, \tag{2.100}$$

tako da se sustav jednadžbi (2.85)-(2.90) uz (2.99) proširuje s još dvije jednadžbe, (2.92) i (2.100), s dodatnim rubnim uvjetima (2.94) i (2.95).

2.7 Hamilton-Jacobi-Bellmanova jednadžba

Ovdje ćemo prikazati još jedan oblik općenitog kriterija optimalnosti baziranog na Bellmanovom principu optimalnosti, odnosno, konceptu dinamičkog programiranja.

Razmotrimo kontinuirani nelinearni dinamički sustav opisan s (2.30) i kriterij optimalnosti (2.31). Vektor upravljanja $\mathbf{u}(t)$ ograničen je na podprostor *m*-dimenzionalnog Euklidskog prostora, $\mathbf{u}(t) \in E$, gdje je $E \subset R^m$. Pretpostavimo da je poznata optimalna trajektorija vektora stanja $\hat{\mathbf{x}}(t)$ od početnog stanja $\mathbf{x}(t_0)$ do stanja u krajnjoj točki $\mathbf{x}(t_f)$.

Po Bellmanovom konceptu optimiranja stanja promatranog sistema (2.30), svaki segment (odsječak) optimalne trajektorije $\hat{\mathbf{x}}(t)$, mora sam po sebi biti optimalna trajektorija. Ako je moguće pronaći barem jedan segment trajektorije stanja koji nije optimalna trajektorija, onda se on može zamjeniti optimalnom trajektorijom, što znači da promatrana trajektorija u cjelini ipak nije optimalna trajektorija. Taj Bellmanov princip najopćenitiji je nužan uvjet optimalnosti.

Minimum funkcionala (2.31) označit ćemo s $J_m\left(\mathbf{x}(t), t\right)$. Budući da po Bellmanovom principu svaki segment optimalne trajektorije između bilo koje točke na optimalnoj trajektoriji $\hat{\mathbf{x}}(t)$ i krajnje točke $\hat{\mathbf{x}}(t\ )$ mora biti optimalan, vrijedi slijedeća relacija

$$J_m\left(\hat{\mathbf{x}}(t),t\right)=\min_{\mathbf{u}\in E}\left\{\phi\left(\hat{\mathbf{x}}(t_f),t_f\right)+\int_t^{t_f}F\left(\hat{\mathbf{x}}(s),\mathbf{u}(s),t\right)ds\right\}. \tag{2.101}$$

Iz gornjeg izraza jasno je da je rubni uvjet za J_m

$$J_m\left(\hat{\mathbf{x}}(t_f),t_f\right)=\phi\left(\hat{\mathbf{x}}(t_f),t_f\right). \tag{2.102}$$

Za vremenski trenutak $t_1=t+\Delta t$ gdje je Δt dovoljno mali vremenski interval, imamo

$$J_m\left(\hat{\mathbf{x}}(t_1),t_1\right)=\min_{\mathbf{u}\in E}\left\{\phi\left(\hat{\mathbf{x}}(t_f),t_f\right)+\int_{t_1}^{t_f}F\left(\hat{\mathbf{x}}(s),\mathbf{u}(s),t\right)ds\right\}. \tag{2.103}$$

Koristeći aproksimaciju

$$J_m\left(\hat{\mathbf{x}}(t),t\right)=\min_{\mathbf{u}\in E}\left\{F\left(\hat{\mathbf{x}}(t),\mathbf{u}(t),t\right)\Delta t+J_m\left(\hat{\mathbf{x}}(t_1),t_1\right)\right\}, \tag{2.104}$$

možemo drugi član na desnoj strani razviti u Taylorov red, zadržavajući samo linearne članove

$$J_m\left(\hat{\mathbf{x}}(t_1),t_1\right)=J_m\left(\hat{\mathbf{x}}(t),t\right)+\mathbf{f}^T\left(\hat{\mathbf{x}}(t),\,\mathbf{u}(t),\,t\right)\cdot\frac{\partial J_m\left(\hat{\mathbf{x}}(t),t\right)}{\partial\hat{\mathbf{x}}(t)}+\frac{\partial J_m\left(\hat{\mathbf{x}}(t),t\right)}{\partial t}\Delta t\,. \tag{2.105}$$

Nakon supstitucije gornjeg izraza u (2.104) i dijeljenja sa Δt, imamo

$$-\frac{\partial J_m\left(\hat{\mathbf{x}}(t),t\right)}{\partial t}=\min_{\mathbf{u}\in E}\left\{F\left(\hat{\mathbf{x}}(t),\,\mathbf{u}(t),\,t\right)+\mathbf{f}^T\left(\hat{\mathbf{x}}(t),\,\mathbf{u}(t),\,t\right)\cdot\frac{\partial J_m\left(\hat{\mathbf{x}}(t),t\right)}{\partial\hat{\mathbf{x}}(t)}\right\}. \tag{2.106}$$

Za optimalni vektor upravljanja $\hat{\mathbf{u}}(t)$ gornja jednadžba postaje

$$-\frac{\partial J_m\left(\hat{\mathbf{x}}(t),t\right)}{\partial t}=F\left(\hat{\mathbf{x}}(t),\,\hat{\mathbf{u}}(t),\,t\right)+\mathbf{f}^T\left(\hat{\mathbf{x}}(t),\,\hat{\mathbf{u}}(t),\,t\right)\cdot\frac{\partial J_m\left(\hat{\mathbf{x}}(t),t\right)}{\partial\hat{\mathbf{x}}(t)}. \tag{2.107}$$

S obzirom da vrijedi

$$\frac{d\,J_m\left(\hat{\mathbf{x}}(t),t\right)}{d\,t}=\frac{\partial J_m\left(\hat{\mathbf{x}}(t),t\right)}{\partial t}+\mathbf{f}^T\left(\hat{\mathbf{x}}(t),\,\hat{\mathbf{u}}(t),\,t\right)\cdot\frac{\partial J_m\left(\hat{\mathbf{x}}(t),t\right)}{\partial\hat{\mathbf{x}}(t)},$$

jednadžbu (2.106) možemo prikazati u slijedećem obliku

$$\min_{\mathbf{u}\in E}\left\{F\left(\hat{\mathbf{x}}(t),\,\mathbf{u}(t),\,t\right)+\frac{d\,J_m\left(\hat{\mathbf{x}}(t),t\right)}{d\,t}\right\}=0\,. \tag{2.108}$$

Da bi uspostavili vezu između jednadžbe (2.107) i Hamiltonovih kanonskih jednadžbi, pogledat ćemo čemu je jednaka mala promjena indeksa performanse $J_m\left(\mathbf{x}(t),\,t\right)$ u odnosu na malu promjenu početnih uvjeta i malu promjenu početnog vremena. Taj problem se svodi na određivanje prve varijacije izraza (2.101) s nespecificiranim početnim vremenom, što je ekvivalentno problemu s nespecificiranim konačnim vremenom uz negativan predznak ispred integrala (podpoglavlje 2.4). Kao rezultat dobivamo izraz (2.60) s negativnim predznakom. Pošto funkcija Θ nije definirana u početnom vremenu, odnosno $\Theta=0$, izraz (2.60) postaje

$$\delta J_m = -H\left(\mathbf{x}(t),\mathbf{u}(t),\lambda(t),t\right)\delta t + \lambda^T(t)\cdot d\,\mathbf{x}(t) - \int_t^{t_f}\left[\left(\frac{\partial H}{\partial \mathbf{x}} + \dot{\lambda}^T\right)\cdot\delta\mathbf{x} + \frac{\partial H}{\partial \mathbf{u}}\cdot\delta\mathbf{u}\right]d\tau\,.$$

Pošto je izraz pod integralom zbog Bellmanovog principa optimalnosti jednak nuli na optimalnoj trajektoriji unutar vremenskog intervala $t \le t \le t_f$, imamo

$$d\,J_m = -H\left(\mathbf{x}(t),\mathbf{u}(t),\lambda(t),t\right)dt + \lambda^T(t)\cdot d\,\mathbf{x}\ , \tag{2.109}$$

odnosno

$$H = -\frac{\partial J_m}{\partial t}, \qquad \lambda^T = \frac{\partial J_m}{\partial \mathbf{x}}\ . \tag{2.110}$$

Na osnovi gornjih izraza, jednadžbu (2.106) možemo napisati kao

$$-\frac{\partial J_m}{\partial t} = \hat{H}\left(\hat{\mathbf{x}},\frac{\partial J_m}{\partial \mathbf{x}},t\right) \tag{2.111}$$

gdje je

$$\hat{H}\left(\hat{\mathbf{x}},\frac{\partial J_m}{\partial \mathbf{x}},t\right) = \min_{u\in E} H\left(\hat{\mathbf{x}},\frac{\partial J_m}{\partial \mathbf{x}},\mathbf{u},t\right). \tag{2.112}$$

Jednadžba (2.111), odnosno (2.106), naziva se Hamilton-Jacobi-Bellmanova jednadžba. Radi se o nelinearnoj parcijalnoj diferencijalnoj jednadžbi prvog reda s rubnim uvjetom (2.102). Jednadžba (2.112) predstavlja, u stvari, drugi prikaz Pontryaginovog principa maksimuma (odnosno minumuma u našem slučaju), optimalni vektor upravljanja $\hat{\mathbf{u}}(t)$ minimizira Hamiltonovu funkciju. Drugim riječima, lijeva strana izraza (2.112) predstavlja desnu stranu izraza (2.107), dok desna strana izraza (2.112) predstavlja desnu stranu izraza (2.106). Ovim je, na neki način, pokazana ekvivalentnost između Pontryaginovog principa maksimuma i Bellmanovog koncepta dinamičkog programiranja.

Ako nema ograničenja na vektor stanja i upravljanja, tada $\hat{\mathbf{u}}(t)$ mora biti takav da zadovoljava nužan uvjet optimalnosti (2.80).

Nadalje, ako pomoću uvjeta optimalnosti (2.80) možemo izraziti $\mathbf{u}(t)$ kao funkciju od $\mathbf{x}(t),\lambda(t),t$, tada u Hamiltonovoj funkciji možemo eliminirati vektor upravljanja i dobiti jednadžbu (2.111) koja ovisi samo o $\mathbf{x}(t)$, t, odnosno samo o $\mathbf{x}(t)$ ukoliko Hamiltonian ne ovisi eksplicitno o vremenu. U tom slučaju, pod uvjetom da uspijemo riješiti jednadžbu (2.111), možemo dobiti vektor upravljanja $\mathbf{u}(t)$ u funkciji vektora stanja $\mathbf{x}(t)$, drugim riječima, dobili bi rješenje problema optimalnog upravljanja u obliku povratne veze po varijablama stanja. Time bi ujedno bio riješen i problem optimalne regulacije.

Nelinearnu parcijalnu diferencijalnu jednadžbu (2.107) s rubnim uvjetom (2.102) općenito je vrlo teško rješiti numerički. Najčešće metode za njeno rješavanje su metoda karakteristika i razvoj u red potencija. Razvojem u red potencija funkcije $J_m\left(\mathbf{x}(t),t\right)$, dobivamo

$$J_m\left(\mathbf{x}(t), t\right) = \sum_{j=1}^{n} p_j x_j + \frac{1}{2}\sum_{j=1}^{n}\sum_{k=1}^{n} p_{jk} x_j x_k + \frac{1}{6}\sum_{j=1}^{n}\sum_{k=1}^{n}\sum_{l=1}^{n} p_{jkl} x_j x_k x_l + \ldots \,, \tag{2.113}$$

što znači da se problem svodi na određivanje nepoznatih koeficijenata p_j, p_j , p_{jkl}, itd. Ako gornji izraz uvrstimo u jednadžbu (2.107) te koeficijente uz sve potencije vektora stanja izjednačimo s nulom, dobivamo sustav od

$$L = \sum_{j=1}^{N} \frac{(n-1+j)!}{(n-1)!\, j!}, \tag{2.114}$$

Riccatijevih diferencijalnih jednadžbi, gdje je *N* stupanj najviše potencije u razvoju. Npr. za $n = 4$, $N = 4$, trebamo rješiti sustav od 69 diferencijalnih jednadžbi da bi dobili rješenje u obliku povratne veze. Dobra strana ovog pristupa je da trebamo rješiti problem s početnim uvjetima, dok su loše strane to što nema garancije da je sistem stabilan, i drugo, broj Riccatijevih diferencijalnih jednadžbi rapidno se povećava s porastom reda sustava *n* i reda polinoma *N*.

3. Numeričke metode optimalnog upravljanja

U prethodnom poglavlju prikazali smo nužne uvjete za problem optimalnog upravljanja. Ti uvjeti svode se općenito na sustav nelinearnih diferencijalnih i algebarskih jednadžbi s rubnim uvjetima. Konkretno, za slučaj bez ograničenja vektora stanja i upravljanja problem se sastoji u pronalaženju

a) n varijabli stanja $\mathbf{x}(t)$,

b) n Lagrangeovih multiplikatora $\lambda(t)$,

c) m upravljačkih varijabli $\mathbf{u}(t)$,

koji simultano zadovoljavaju

I) sustav od n diferencijalnih jednadžbi (zadani početni rubni uvjeti) koji sadrži vektore $\mathbf{x}(t)$ i $\mathbf{u}(t)$ (jednadžba (2.38)).

II) sustav od n diferencijalnih jednadžbi (zadani konačni rubni uvjeti) koji sadrži vektore $\mathbf{x}(t)$, $\lambda(t)$ i $\mathbf{u}(t)$ (jednadžba (2.39)).

III) sustav od m algebarskih jednadžbi (uvjeti optimalnosti) koji sadrži vektore $\mathbf{x}(t)$, $\lambda(t)$ i $\mathbf{u}(t)$ (jednadžba (2.40)).

IV) početne i konačne rubne uvjete za vektore $\mathbf{x}(t)$ i $\lambda(t)$ (jednadžba (2.41)).

Rješavanje takvog problema je vrlo netrivijalno čak i za linearne dinamičke sustave, dok se uvođenjem ograničenja problem dodatno usložnjava. Zbog toga je nužno razviti odgovarajuće numeričke metode za rješavanje tog problema.

Sve numeričke metode koje tretiraju taj problem mogu se staviti u okvir principa "sukcesivne linearizacije". Krene se od nekog početnog rješenja koje zadovoljava jedan, dva ili tri od četiri, gore navedena, kriterija (tzv. nominalno rješenje). Tada se nominalno rješenje iterativnim postupkom modificira sve dok, s odgovarajućom točnošću, ne zadovolji preostale uvjete.

Ovdje ćemo ukratko razmotriti neke klasične numeričke metode optimalnog upravljanja, [3], [4], [9], [11], [12].

3.1 Metoda kvazilinearizacije

Ovdje ćemo dati prikaz iterativne metode za rješavanje sustava diferencijalnih jednadžbi s rubnim uvjetima (nužni uvjeti optimalnosti I-IV). Metodom kvazilinearizacije transformiramo sustav nelinearnih diferencijalnih jednadžbi s rubnim uvjetima u sustav nestacionarnih linearnih diferencijalnih jednadžbi s rubnim uvjetima, koji je lakši za rješavanje. Za nesingularan problem vektor upravljanja $\mathbf{u}(t)$ možemo prikazati kao funkciju od $\mathbf{x}(t)$ i $\lambda(t)$, na osnovi uvjeta optimalnosti

$$\frac{\partial H}{\partial \mathbf{u}} = \mathbf{0} \quad \Rightarrow \quad \mathbf{u} = \mathbf{u}\left(\mathbf{x}, \lambda, t\right). \tag{3.1}$$

U tom slučaju, u dinamičkim jednadžbama sustava (2.38) i u jednadžbama za Lagrangeove multiplikatore (2.39), možemo eliminirati vektor upravljanja i dobiti sustav međusobno spregnutih diferencijalnih jednadžbi za $\mathbf{x}(t)$ i $\lambda(t)$

$$\dot{\mathbf{x}} = \mathbf{f}\left(\mathbf{x}, \lambda, t\right), \tag{3.2}$$

$$\dot{\lambda} = \mathbf{g}\left(\mathbf{x}, \lambda, t\right), \tag{3.3}$$

sa zadanim rubnim uvjetima (2.41).

Gornji sustav diferencijalnih jednadžbi prikazat ćemo jednom jednadžbom

$$\dot{\mathbf{z}} = \mathbf{h}\left(\mathbf{z}, t\right), \tag{3.4}$$

gdje su $\mathbf{z} = \begin{bmatrix} \mathbf{x} & \lambda \end{bmatrix}^T$, $\mathbf{h} = \begin{bmatrix} \mathbf{f} & \mathbf{g} \end{bmatrix}^T$.

Linearizacijom jednadžbe (3.4) oko nominalne trajektorije imamo

$$\dot{\mathbf{z}}^{k+1}(t) = \mathbf{h}\left(\mathbf{z}^k(t), t\right) + \frac{\partial \mathbf{h}}{\partial \mathbf{z}^k}\left(\mathbf{z}^{k+1}(t) - \mathbf{z}^k(t)\right), \tag{3.5}$$

gdje je $\mathbf{z}^k(t)$ nominalna trajektorija u *k*-tom koraku iteriranja ($k = 0,1,2,...$).

Jednadžba (3.5) predstavlja, u stvari, sustav neautonomnih linearnih diferencijalnih jednadžbi

$$\dot{\mathbf{z}}^{k+1}(t) = \mathbf{A}(t) \cdot \mathbf{z}^{k+1}(t) + \mathbf{b}(t), \tag{3.6}$$

gdje je

$$\mathbf{A}(t) = \frac{\partial \mathbf{h}\left(\mathbf{z}^k(t), t\right)}{\partial \mathbf{z}^k}, \tag{3.7}$$

$$\mathbf{b}(t) = \mathbf{h}\left(\mathbf{z}^k(t), t\right) - \mathbf{A}(t) \cdot \mathbf{z}^k(t). \tag{3.8}$$

Da bi rješili linearni sustav (3.6) s rubnim uvjetima možemo primjeniti Riccatijevu transformaciju, kojom problem s rubnim uvjetima prebacujemo u problem s početnim uvjetima. Druga mogućnost je korištenje principa superpozicije koji vrijedi za linearne sustave. Postupak iteriranja se nastavlja sve dok razlika između dva rješenja

$$\varepsilon(t) = \left| \mathbf{z}^{k+1}(t) - \mathbf{z}^k(t) \right|, \tag{3.9}$$

ne bude dovoljno mala.

3.2 Diskretno dinamičko programiranje

Diskretno dinamičko programiranje bazira se na vremenskoj diskretizaciji Hamilton-Jacobi-Bellmanove jednadžbe (2.108). Aproksimirajući vremensku derivaciju

$$\frac{d\,J\left(\mathbf{x}(t),t\right)}{d\,t} \approx \frac{J\left(\mathbf{x}(t+\Delta t),t+\Delta t\right)-J\left(\mathbf{x}(t),t\right)}{\Delta t} \approx \frac{J\left(\mathbf{x}(t)+\dot{\mathbf{x}}(t)\Delta t,t+\Delta t\right)-J\left(\mathbf{x}(t),t\right)}{\Delta t}$$

dobivamo

$$J\left(\mathbf{x}(t),t\right)=\min_{\mathbf{u}\in E}\left\{F\left(\mathbf{x}(t),\,\mathbf{u}(t),\,t\right)\Delta t+J\left(\mathbf{x}(t)+\mathbf{f}\left(\mathbf{x}(t),\mathbf{u}(t)\right)\Delta t,\,t+\Delta t\right)\right\}. \tag{3.10}$$

Uvedemo li vremensku diskretizaciju

$$t_k=k\Delta t\,,\qquad \mathbf{x}_k=\mathbf{x}(t_k)\,,\qquad \text{itd.},\qquad \text{za}\qquad k=0,1,...,N$$

gornju jednadžbu možemo prikazati na slijedeći način

$$J_k\left(\mathbf{x}_k\right)=\min_{\mathbf{u}_k\in E}\left\{F_k\left(\mathbf{x}_k\,,\,\mathbf{u}_k\,,\right)\Delta t+J_{k+1}\left(\mathbf{x}_k+\mathbf{f}\left(\mathbf{x}_k\,,\mathbf{u}_k\right)\Delta t\right)\right\}, \tag{3.11}$$

s rubnim uvjetom

$$J_N\left(\mathbf{x}_N\right)=\phi_N\left(\mathbf{x}_N\right). \tag{3.12}$$

Jednadžba (3.11) naziva se *funkcijska jednadžba dinamičkog programiranja*. Rješavanje gornje jednadžbe provodi se unazad u vremenu počevši od $k=N$. Rezultat minimizacije u svakom vremenskom trenutku $t_k=k\Delta t$ je optimalni vektor $\hat{\mathbf{u}}_k$ kao funkcija stanja $\mathbf{x}_k$. Međutim, takav pristup je nepraktičan za veliki broj vremenskih podintervala *N*, a također i za nelinearne dinamičke sustave s ograničenjima vektora stanja i upravljanja. Zbog toga su razvijene različite metode za rješavanje jednadžbe (3.11) koje se mogu naći u [11], [12].

Većina tih metoda bazira se na diskretizaciji vektora stanja $\mathbf{x}_k$, gdje se dozvoljeni interval vektora stanja (obično definiran ograničenjima) podijeli na *M* diskretnih vrijednosti. Na taj način, ograničenja vektora stanja i upravljanja, koja kod svih ostalih numeričkih metoda optimalnog upravljanja predstavljaju veliki problem, ovdje predstavljaju olakšanje. Međutim, taj pristup postavlja velike zahtijeve na potrebnu memoriju za pohranjivanje međurezultata izračunavanja, kao i na samu brzinu računanja. S obzirom da u svakom vremenskom trenutku *k* trebamo memorirati vektor $\mathbf{f}\left(\mathbf{x}_k,\mathbf{u}_k\right)$ kao i $J_{k+1}\left(\mathbf{x}_k+\mathbf{f}\left(\mathbf{x}_k,\mathbf{u}_k\right)\Delta t\right), J_k\left(\mathbf{x}_k\right)$ potrebno je

$$\Gamma=(n+2)\,M^n\,, \tag{3.13}$$

memorijskih mjesta. Vidimo da povećavanjem reda sustava taj broj drastično raste, što predstavlja najveću manu diskretnog dinamičkog programiranja. Npr. za sustav četvrtog reda uz $M=100$, potrebno nam je $\Gamma=6\cdot10^8$ memorijskih mjesta, odnosno, oko $600\,Mb$ RAM memorije.

Iako je diskretno dinamičko programiranje vrlo efikasno za tretiranje problema s ograničenjima vektora stanja i upravljanja, zbog navedenog problema, primjenjuje se uglavnom za sustave niskih dimenzija (do $n = 3$).

3.3 Ritzova metoda

Razmotrimo problem nalaženja minimuma funkcionala $J\left[\mathbf{x}(t), \mathbf{u}(t)\right]$. Neka je $\left(\mathbf{x}^k(t), \mathbf{u}^k(t)\right)$, $k = 0,1,\ldots$, beskonačan konvergentni niz funkcija, takav da funkcija cilja postaje sve manja kako se *k* povećava (tzv. minimizirajući niz). Odnosno,

$$J\left[\mathbf{x}^{k+1}(t), \mathbf{u}^{k+1}(t)\right] < J\left[\mathbf{x}^k(t), \mathbf{u}^k(t)\right], \tag{3.14}$$

$$J\left[\lim_{k\to\infty}\left(\mathbf{x}^k(t), \mathbf{u}^k(t)\right)\right] = \lim_{k\to\infty} J\left[\mathbf{x}^k(t), \mathbf{u}^k(t)\right]. \tag{3.15}$$

Razmatramo problem minimizacije funkcije cilja

$$J\left[\mathbf{x}(t), \mathbf{u}(t)\right] = \int_{t_0}^{t_f} F\left(\mathbf{x}(t), \mathbf{u}(t), t\right) dt\,, \tag{3.16}$$

uz zadani sustav nelinearnih diferencijalnih jednadžbi s rubnim uvjetima

$$\dot{\mathbf{x}}(t) = \mathbf{f}\left(\mathbf{x}(t), \mathbf{u}(t), t\right), \qquad \mathbf{x}(t_0) = \mathbf{x}_0\,,\ \mathbf{x}(t_f) = \mathbf{x}_f\,. \tag{3.17}$$

Uvođenjem Lagrangeovih multiplikatora, možemo reformulirati gornji problem kao minimizaciju slijedećeg funkcionala

$$J\left[\mathbf{x}(t), \mathbf{u}(t), \lambda(t)\right] = \int_{t_0}^{t_f} \Phi\left(\mathbf{x}(t), \dot{\mathbf{x}}(t), \mathbf{u}(t), \lambda(t), t\right) dt\,, \tag{3.18}$$

gdje je

$$\Phi = F + \lambda^T \cdot (\mathbf{f} - \dot{\mathbf{x}})\,. \tag{3.19}$$

Uzmimo sada niz funkcija $\mathbf{w}^0(t)$, $w^1(t)$, $w^2(t)$, …, $w^k(t)$, gdje $\mathbf{w}\ (t)$ zadovoljava sve rubne uvjete, dok su $w^1(t)$, $w^2(t)$, …, $w^k(t)$ tzv. osnovne funkcije (*basis functions*) koje se uglavnom izabiru tako da budu međusobno ortogonalne i da iščezavaju u početnoj i konačnoj točki. Formiramo linearnu kombinaciju

$$\mathbf{W}^k(t) = \mathbf{w}^0 + \sum_{j=1}^{k} \mathbf{c}_j w^j(t)\,, \tag{3.19}$$

gdje je $\mathbf{W}^k(t)$ proširena vektorska funkcija $\left[\mathbf{x}^k(t) \quad \mathbf{u}^k(t) \quad \lambda^k(t)\right]^T$, dok su $\mathbf{c}_j$ slobodni parametri koje treba odrediti iz uvjeta minimalne vrijednosti funkcije cilja. Za komponente vektora $\mathbf{w}^0(t)$ obično se uzimaju linearne funkcije oblika $\mathbf{w}^0(t) = \mathbf{a}t + \mathbf{b}$ gdje se komponente vektora **a** i **b** određuju tako da zadovoljavaju rubne uvjete. Ukoliko rubni uvjeti nisu specificirani za neke komponente vektora $\mathbf{W}\ (t)$, tada se

odgovarajuće komponente vektora **a** i **b** ostavljaju slobodnima i određuju iz uvjeta minimalne vrijednosti funkcije cilja.

Uvrstimo li izraz (3.19) u jednadžbu (3.18), te izračunamo integrale odgovarajućih kombinacija funkcija $\mathbf{w}^0(t)$, $w^1(t)$, …, $w\ (t)$, dobivamo funkcional kao funkciju slobodnih parametara

$$J = J\left(\mathbf{c}_1, ..., \mathbf{c}_k, \tilde{\mathbf{a}}, \tilde{\mathbf{b}}\right), \tag{3.20}$$

gdje su $\tilde{\mathbf{a}}$ i $\tilde{\mathbf{b}}$ vektori odgovarajućih slobodnih komponenti vektora **a** i **b**.

Minimalnu vrijednost funkcionala (3.20) možemo odrediti iz uvjeta

$$\frac{\partial J}{\partial \mathbf{c}_j} = \mathbf{0}, \qquad \frac{\partial J}{\partial \tilde{\mathbf{a}}} = \mathbf{0}, \qquad \frac{\partial J}{\partial \tilde{\mathbf{b}}} = \mathbf{0}, \tag{3.21}$$

koji predstavljaju, općenito, skup nelinearnih algebarskih jednadžbi. Druga mogućnost za određivanje minimuma gornjeg funkcionala je korištenje neke od standardnih numeričkih metoda parametarske optimizacije, kao npr. gradijentnog algoritma.

Glavni problemi kod Ritzove metode su izbor odgovarajućeg skupa osnovnih funkcija i određivanje dovoljnog broja članova k u razvoju (3.19). Također, za nelinearne dinamičke sustave kao i za nekvadratične funkcije cilja, problem je izračunati sve potrebne integrale kombinacija osnovnih funkcija.

4. Optimalno upravljanje nelinearnim sustavima primjenom BPTT algoritma

Backpropagation-through-time (BPTT) algoritam, [13]-[16], je vremensko proširenje *backpropagation* algoritma (BP), za slučaj kada je greška koja se minimizira dana duž određenog vremenskog intervala (kao u slučaju optimalnog upravljanja). U srži BP algoritma leži jednostavno i egzaktno računanje derivacija funkcije cilja u odnosu na parametre (težine) sustava i podešavanje parametara na osnovi tih derivacija u samo jednom prolazu kroz sistem (gradijentni algoritam). BPTT algoritam proširuje ovu metodu primjenom na dinamičke sustave. Zbog činjenice da dinamički sustavi sadrže povratnu vezu, direktno računanje derivacija može biti vrlo komplicirano. Rješenje tog problema leži u lančanom pravilu za obične derivacije [13], [17], [18], što ima za posljedicu prostiranje pogreške (podešavanje parametara) unazad u vremenu, na osnovu čega je algoritam i dobio ime.

BPTT algoritam je vrlo efikasna metoda s primjenom na analizu osjetljivosti, prepoznavanje uzoraka, identifikaciju sustava, optimalno upravljanje, detekciju grešaka, itd. Može biti primjenjen na neuronske mreže, fuzzy sustave, ekonometrijske modele, itd.

U ovom poglavlju izvodimo numerički algoritam baziran na gradijentnoj metodi, za računanje upravljačkih varijabli koje minimiziriju zadani kriterij optimalnosti. Pri tom ćemo koristiti osnovnu ideju BPTT algoritma (lančano pravilo za obične derivacije i propagacija unazad u vremenu) bez pozivanja na formalizam dinamičkih neuronskih mreža.

4.1 Izvod BPTT algoritma za nelinearne sustave prvog reda

Na početku, radi boljeg uvida u osnovnu ideju BPTT algoritma, prikazati ćemo njegovu primjenu na nelinearne dinamičke sustave prvog reda. Dinamiku sustava prvog reda možemo prikazati općenitom nelinearnom diferencijalnom jednadžbom prvog reda

$$\dot{x}(t) = \phi\big(x(t),\, u(t)\big), \quad x(t_0) = x_0, \tag{4.1}$$

gdje je $x(t)$ stanje sustava, dok je upravljačka funkcija $u(t)$ jednoznačno određena kriterijem optimalnosti

$$J = \min_{u(t)} \int_{t_0}^{t_f} F\big(x(t),\, u(t)\big)dt\,. \tag{4.2}$$

Problem optimalnog upravljanja sastoji se u nalaženju funkcije $u(t)$ koja minimizira kriterij optimalnosti (4.2).

Da bi pristupili numeričkom tretiranju prethodnog problema, napravit ćemo njegovu vremensku diskretizaciju

$$x_{i+1} = x_i + T\phi(x_i, u_i), \quad x_0 = x_0, \quad i = 0,1,\ldots,N-1; \tag{4.3}$$

$$J(u_0, u_1, \ldots, u_{N-1}) = \min_u T\sum_{i=0}^{N-1} F(x_i, u_i), \tag{4.4}$$

gdje je $x_i = x(iT)$, $u_i = u(iT)$, N je broj podintervala, a T je period diskretizacije

$$T = \frac{t_f - t_0}{N}. \tag{4.5}$$

Na ovaj način problem smo sveli, sa određivanja kontinuirane funkcije $u(t)$ koja minimizira (4.2), na određivanje skupa varijabli $(u_0, u_1, \ldots, u_{N-1})$ koji minimizira funkciju (4.4) uz ograničenja tipa jednakosti (4.3).

Taj problem se može riješiti različitim metodama nelinearnog programiranja, npr. metodom Lagrangeovih multiplikatora, metodom kaznenih funkcija, itd. Međutim, ideja svih tih metoda je da se varijable $(u_0, u_1, \ldots, u_{N-1})$, $(x_0, x_1, \ldots, x_N)$ tretiraju kao nezavisne varijable, što se postiže stavljanjem ograničenja tipa jednakosti (4.3) u funkciju cilja. U algoritmu koji ćemo sada prikazati, samo varijable $(u_0, u_1, \ldots, u_{N-1})$ tretiramo kao nezavisne (kao što i jesu) dok ćemo ograničenja (4.3) uzimati u obzir tijekom računanja gradijenta funkcije (4.4).

Varijable $(u_0, u_1, \ldots, u_{N-1})$ koje minimiziraju funkciju (4.4), odredit ćemo iterativno primjenom gradijentnog algoritma

$$u_j^{(k+1)} = u_j^{(k)} - \eta \frac{\partial J(u_0^{(k)}, u_1^{(k)}, \ldots, u_{N-1}^{(k)})}{\partial u_j^{(k)}}, \tag{4.6}$$

gdje je $j = 0,1,\ldots,N-1$; $k = 0,1,\ldots,M$, dok je η koeficijent konvergencije, indeks k predstavlja k-tu iteraciju gradijentnog algoritma, M je broj iteracija algoritma.

Gradijent funkcije cilja (kriterija optimalnosti), na osnovi izraza (4.4) i (4.3), za k-tu iteraciju je

$$\frac{\partial J}{\partial u_j} = T\frac{\partial F(x_j, u_j)}{\partial u_j} + T\sum_{i=j+1}^{N-1} \frac{\partial F(x_i, u_i)}{\partial u_j}, \tag{4.7}$$

gdje je $j = 0,1,\ldots,N-1$, i taj izraz predstavlja suštinu BPTT algoritma. Iz izraza (4.4) vidimo da funkcija cilja J predstavlja zbroj funkcija $F(x_i, u_i)$ za $i = 0,1,\ldots,N-1$, tako da od parcijalne derivacije J po u_i preostanu sumandi $F(x_i, u_i)$ za koje je $i \geq j$. Prvi član na desnoj strani izraza (4.7) posljedica je međusobne nezavisnosti varijabli $(u_0, u_1, \ldots, u_{\ -1})$, dok je drugi član na desnoj strani posljedica međusobne ovisnosti varijabli stanja i upravljačkih varijabli prema izrazu (4.3) koji možemo općenito napisati kao

$$x_{i+1} = f(x_i, u_i), \qquad i = 0,1,\ldots,N-1, \tag{4.8}$$

gdje je

$$f(x_i, u_i) = x_i + T\phi(x_i, u_i). \tag{4.9}$$

Parcijalne derivacije pod sumom u izrazu (4.7) predstavljaju, zbog ovisnosti (4.8), uvjetne derivacije, pa ako tu sumu označimo s

$$S(j) = \sum_{i=j+1}^{N-1} \frac{\partial F(x_i, u_i)}{\partial u_j}, \qquad j = 0,1,\ldots,N-2, \tag{4.10}$$

imamo

$$S(j) = \sum_{i=j+1}^{N-1} \frac{\partial F(x_i, u_i)}{\partial u_j} = \sum_{i=j+1}^{N-1} \frac{\partial F(x_i, u_i)}{\partial x_i} \frac{\partial x_i}{\partial u_j}. \tag{4.11}$$

Ako bi gornju sumu, $S(j)$, računali od $j = 0$ do $j = N-2$ imali bi ukupno $(N-1)(N-2)/2$ operacija zbrajanja i množenja. Međutim ako gornju sumu počnemo računati od $j = N-2$ do $j = 0$ dobit ćemo rekurzivne relacije za sumu koje će bitno smanjiti broj računskih operacija za njeno izračunavanje.

Za $j = N-2$ imamo

$$S(N-2) = \frac{\partial F(x_{N-1}, u_{N-1})}{\partial x_{N-1}} \frac{\partial x_{N-1}}{\partial u_{N-2}} = \frac{\partial F(x_{N-1}, u_{N-1})}{\partial x_{N-1}} \frac{\partial f(x_{N-2}, u_{N-2})}{\partial u_{N-2}}.$$

Označimo

$$P(N-2) = \frac{\partial F(x_{N-1}, u_{N-1})}{\partial x_{N-1}}.$$

Za $j = N-3$ imamo

$$\begin{aligned} S(N-3) &= \frac{\partial F(x_{N-2}, u_{N-2})}{\partial x_{N-2}} \frac{\partial x_{N-2}}{\partial u_{N-3}} + \frac{\partial F(x_{N-1}, u_{N-1})}{\partial x_{N-1}} \frac{\partial x_{N-1}}{\partial u_{N-3}} = \\ &= \frac{\partial F(x_{N-2}, u_{N-2})}{\partial x_{N-2}} \frac{\partial x_{N-2}}{\partial u_{N-3}} + \frac{\partial F(x_{N-1}, u_{N-1})}{\partial x_{N-1}} \frac{\partial x_{N-1}}{\partial x_{N-2}} \frac{\partial x_{N-2}}{\partial u_{N-3}} = \\ &= \left[\frac{\partial F(x_{N-2}, u_{N-2})}{\partial x_{N-2}} + \frac{\partial x_{N-1}}{\partial x_{N-2}} \frac{\partial F(x_{N-1}, u_{N-1})}{\partial x_{N-1}} \right] \frac{\partial x_{N-2}}{\partial u_{N-3}} \\ &= \left[\frac{\partial F(x_{N-2}, u_{N-2})}{\partial x_{N-2}} + \frac{\partial f(x_{N-2}, u_{N-2})}{\partial x_{N-2}} P(N-2) \right] \frac{\partial f(x_{N-3}, u_{N-3})}{\partial u_{N-3}} \end{aligned}$$

Izraz u uglatoj zagradi označit ćemo sa

$$P(N-3) = \left[\frac{\partial F(x_{N-2}, u_{N-2})}{\partial x_{N-2}} + \frac{\partial f(x_{N-2}, u_{N-2})}{\partial x_{N-2}} P(N-2) \right].$$

Nadalje, za $j = N-4$ imamo

$$S(N-4) = \frac{\partial F(x_{N-3},u_{N-3})}{\partial x_{N-3}}\frac{\partial x_{N-3}}{\partial u_{N-4}} + \frac{\partial F(x_{N-2},u_{N-2})}{\partial x_{N-2}}\frac{\partial x_{N-2}}{\partial u_{N-4}} + \frac{\partial F(x_{N-1},u_{N-1})}{\partial x_{N-1}}\frac{\partial x_{N-1}}{\partial u_{N-4}} =$$

$$= \frac{\partial F(x_{N-3},u_{N-3})}{\partial x_{N-3}}\frac{\partial x_{N-3}}{\partial u_{N-4}} + \frac{\partial F(x_{N-2},u_{N-2})}{\partial x_{N-2}}\frac{\partial x_{N-2}}{\partial x_{N-3}}\frac{\partial x_{N-3}}{\partial u_{N-4}} +$$

$$+ \frac{\partial F(x_{N-1},u_{N-1})}{\partial x_{N-1}}\frac{\partial x_{N-1}}{\partial x_{N-2}}\frac{\partial x_{N-2}}{\partial x_{N-3}}\frac{\partial x_{N-3}}{\partial u_{N-4}} =$$

$$= \left[\frac{\partial F(x_{N-3},u_{N-3})}{\partial x_{N-3}} + \frac{\partial x_{N-2}}{\partial x_{N-3}}\left[\frac{\partial F(x_{N-2},u_{N-2})}{\partial x_{N-2}} + \frac{\partial x_{N-1}}{\partial x_{N-2}}\frac{\partial F(x_{N-1},u_{N-1})}{\partial x_{N-1}}\right]\right]\frac{\partial x_{N-3}}{\partial u_{N-4}}$$

$$= \left[\frac{\partial F(x_{N-3},u_{N-3})}{\partial x_{N-3}} + \frac{\partial f(x_{N-3},u_{N-3})}{\partial x_{N-3}}P(N-3)\right]\frac{\partial f(x_{N-4},u_{N-4})}{\partial u_{N-4}}$$

Izraz u zagradi označit ćemo

$$P(N-4) = \left[\frac{\partial F(x_{N-3},u_{N-3})}{\partial x_{N-3}} + \frac{\partial f(x_{N-3},u_{N-3})}{\partial x_{N-3}}P(N-3)\right].$$

Gornje izraze možemo sada prikazati na slijedeći način

$$P(N-2) = \frac{\partial F(x_{N-},u_{N-})}{\partial x_{N-}},$$

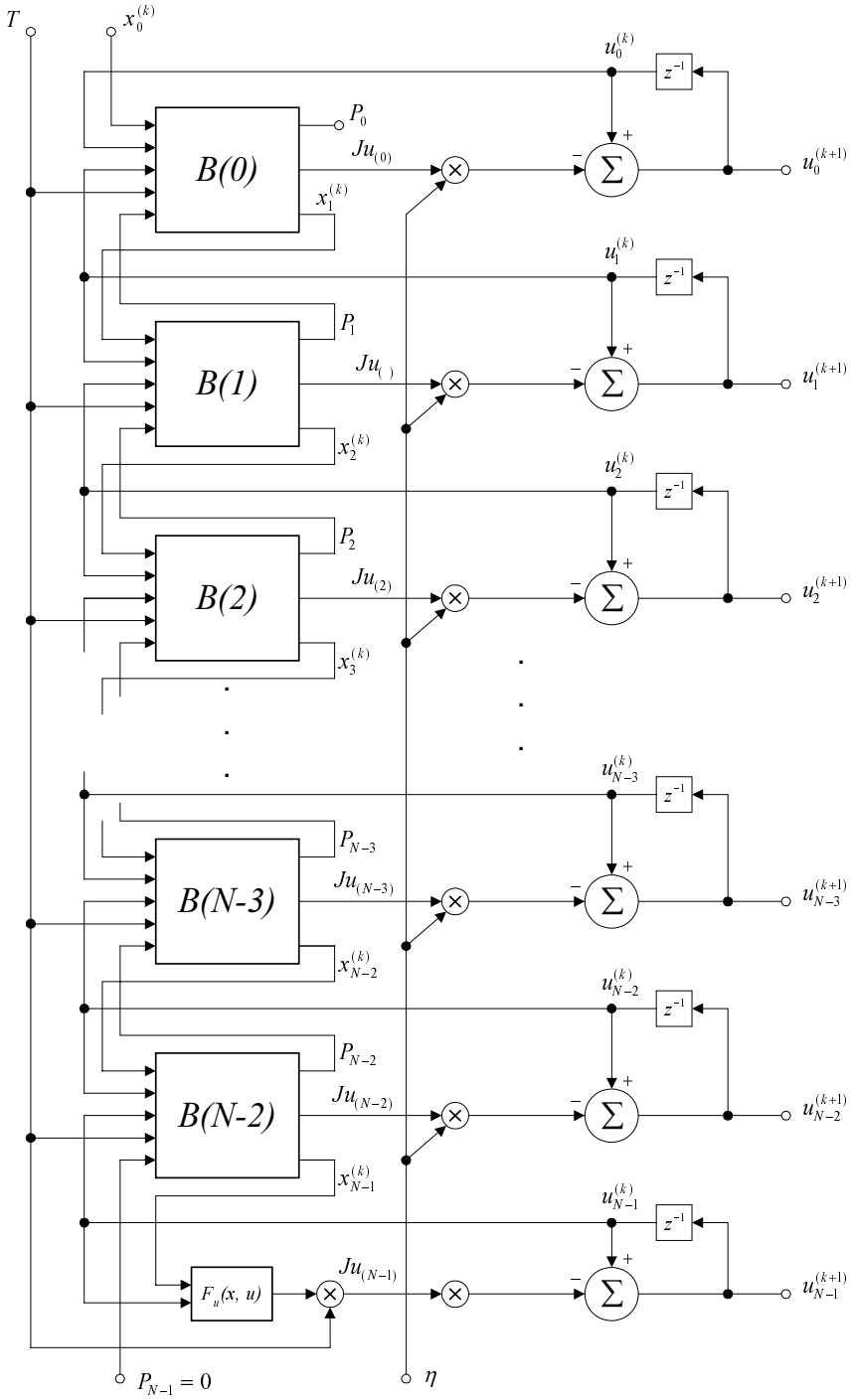
$$P(N-3) = \frac{\partial F(x_{N-2},u_{N-2})}{\partial x_{N-2}} + \frac{\partial f(x_{N-2},u_{N-2})}{\partial x_{N-2}}P(N-2),$$

$$P(N-4) = \frac{\partial F(x_{N-3},u_{N-3})}{\partial x_{N-3}} + \frac{\partial f(x_{N-3},u_{N-3})}{\partial x_{N-3}}P(N-3),$$

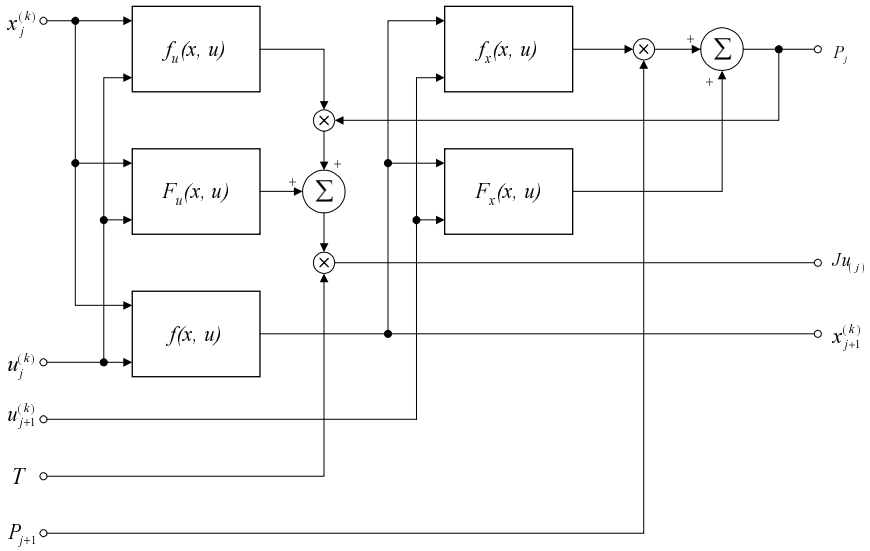
$$S(N-2) = P(N-2)\frac{\partial f(x_{N-2},u_{N-2})}{\partial u_{N-2}},$$

$$S(N-3) = P(N-3)\frac{\partial f(x_{N-3},u_{N-3})}{\partial u_{N-3}},$$

$$S(N-4) = P(N-4)\frac{\partial f(x_{N-4},u_{N-4})}{\partial u_{N-4}},$$



Slika 4.1 *Shematski prikaz gradijentnog algoritma.*



Slika 4.2 *Shematski prikaz bloka B(j) za računanje vrijednosti gradijenta na osnovi BPTT algoritma.*

iz čega se mogu napisati slijedeće rekurzivne relacije

$$P(j)=\frac{\partial F(x_{j+1},u_{j+1})}{\partial x_{j+1}}+\frac{\partial f(x_{j+1},u_{j+1})}{\partial x_{j+1}}P(j+1) \quad ; \qquad P(N-1)=0\,, \tag{4.12}$$

$$S(j)=P(j)\frac{\partial f(x_j,u_j)}{\partial u_j}\,, \tag{4.13}$$

$$\frac{\partial J}{\partial u_j}=T\left(\frac{\partial F(x_j,u_j)}{\partial u_j}+S(j)\right) \quad ; \qquad \frac{\partial J}{\partial u_{N-1}}=T\frac{\partial F(x_{N-1},u_{N-1})}{\partial u_{N-1}}\,, \tag{4.14}$$

za $j=N-2,\ N-3,\ \ldots,0$.

Na slici 4.1 i 4.2 dan je shematski prikaz gradijentnog algoritma (4.6), (4.12)-(4.14). Ako blokove *B(j)* koji računaju vrijednost gradijenta shvatimo kao perceptrone, vidimo da su izlazi perceptrona spojeni na ulaze susjednih perceptrona, odnosno, ovakav prikaz navedenog algoritma analogan je koncepciji rekurentnih (dinamičkih) neuronskih mreža. Na slici 4.3 dana je alternativna reprezentacija algoritma (4.6), (4.12)-(4.14).

Algoritam (4.12) do (4.14) izveden je za problem optimalnog upravljanja sustava (4.1) sa zadanim početnim uvjetima (rubni uvjet nije specificiran). Taj problem proširiti ćemo za slučaj sa zadanim rubnim uvjetima, odnosno

$$\dot{x}(t)=\phi\big(x(t),\,u(t)\big)\,, \qquad x(t_0)=x_0\,, \quad x(t_f)=x_f\,, \tag{4.15}$$

ili u diskretnom obliku

$$x_{i+1} = f(x_i, u_i), \qquad x_0 = x_0, \qquad x_N = x_f, \tag{4.16}$$

gdje je $i = 0,1,\ldots,N-1$.

Da bi mogli koristiti algoritam (4.12) do (4.14) za problem s rubnim uvjetima, rubni uvjet ćemo razmatrati kao ograničenje i tretirat ćemo ga metodom kaznenih funkcija. Ako definiramo rubni uvjet u obliku

$$x_N - x_f = 0, \tag{4.17}$$

i formiramo kaznenu funkciju

$$G_B(u_0, u_1, \ldots, u_{N-1}) = K_B\left(x_N - x_f\right)^2, \tag{4.18}$$

gdje x_N ovisi o $(u_0, u_1, \ldots, u_{N-1})$ implicitno preko (4.16), a K_B je konstantni pozitivni koeficijent kaznene funkcije, vidimo da funkcija (4.18) ima minimalnu vrijednost jednaku nuli, kada je ispunjen uvjet (4.17), dok je za slučaj kada nije ispunjen taj uvjet vrijednost funkcije veća od nule. Kada kaznenu funkciju dodamo funkciji cilja (4.4),

$$J(u_0, u_1, \ldots, u_{N-1}) = \min_u T\sum_{i=0}^{N-1} F(x_i, u_i) + K_B(x_N - x_f)^2 \tag{4.19}$$

gradijentni algoritam će nam garantirati da ćemo minimizacijom funkcije (4.19), odnosno (4.18), težiti zadovoljenju jednakosti (4.17). Zbog toga, moramo izvesti gradijent kaznene funkcije (4.18)

$$\frac{\partial G_B}{\partial u_j} = 2K_B\left(x_N - x_f\right)\frac{\partial x_N}{\partial u_j}, \qquad j = 0,1,\ldots,N-1. \tag{4.20}$$

Zbog istih razloga kao kod izvoda algoritma (4.12) - (4.14), krenuti ćemo od $j = N-1$

$$\frac{\partial G_B}{\partial u_{N-1}} = 2K_B\left(x_N - x_f\right)\frac{\partial x_N}{\partial u_{N-1}},$$

$$\frac{\partial G_B}{\partial u_{N-2}} = 2K_B\left(x_N - x_f\right)\frac{\partial x_N}{\partial u_{N-2}} = 2K_B\left(x_N - x_f\right)\frac{\partial x_N}{\partial x_{N-1}}\frac{\partial x_{N-1}}{\partial u_{N-2}},$$

$$\frac{\partial G_B}{\partial u_{N-3}} = 2K_B\left(x_N - x_f\right)\frac{\partial x_N}{\partial u_{N-3}} = 2K_B\left(x_N - x_f\right)\frac{\partial x_N}{\partial x_{N-1}}\frac{\partial x_{N-1}}{\partial x_{N-2}}\frac{\partial x_{N-2}}{\partial u_{N-3}},$$

$$\frac{\partial G_B}{\partial u_{N-4}} = 2K_B\left(x_N - x_f\right)\frac{\partial x_N}{\partial u_{N-4}} = 2K_B\left(x_N - x_f\right)\frac{\partial x_N}{\partial x_{N-1}}\frac{\partial x_{N-1}}{\partial x_{N-2}}\frac{\partial x_{N-2}}{\partial x_{N-3}}\frac{\partial x_{N-3}}{\partial u_{N-4}}.$$

Ako stavimo

$$D(N-2) = \frac{\partial x_N}{\partial x_{N-1}} = \frac{\partial f(x_{N-1}, u_{N-1})}{\partial x_{N-1}},$$

$$D(N-3) = \frac{\partial x_N}{\partial x_{N-1}}\frac{\partial x_{N-1}}{\partial x_{N-2}} = D(N-2)\frac{\partial f(x_{N-2}, u_{N-2})}{\partial x_{N-2}},$$

$$D(N-4) = \frac{\partial x_N}{\partial x_{N-1}}\frac{\partial x_{N-1}}{\partial x_{N-2}}\frac{\partial x_{N-2}}{\partial x_{N-}} = D(N-3)\frac{\partial f(x_{N-}, u_{N-})}{\partial x_{N-}},$$

možemo napisati slijedeće rekurzivne relacije

$$D(j) = D(j+1)\frac{\partial f(x_{j+1}, u_{j+1})}{\partial x_{j+1}} \; ; \qquad D(N-1) = 1, \tag{4.21}$$

$$\frac{\partial G_B}{\partial u_j} = 2K_B\left(x_N - x_f\right)D(j)\frac{\partial f(x_j, u_j)}{\partial u_j}, \tag{4.22}$$

za $j = N-1, N-2, \ldots, 0$. Ovim smo rješili problem rubnih uvjeta. Slijedeći problem su ograničenja tipa nejednakosti

$$g_j\left(x(t),\, u(t)\right) \geq 0\,, \qquad j = 1, \ldots, p\,, \tag{4.23}$$

ili u diskretnom obliku

$$g_j\left(x_i,\, u_i\right) \geq 0\,, \qquad i = 0,1,\ldots,N-1\,, \quad j = 1,\ldots,p\,, \tag{4.24}$$

koji ćemo također rješiti, formiranjem kaznene funkcije oblika

$$G_V(u_0, u_1, \ldots, u_{N-1}) = K_V^* \sum_{j=1}^{p}\sum_{i=0}^{N-1} g_j^2(x_i, u_i) H^-\left(g_j(x_i, u_i)\right), \tag{4.25}$$

gdje je

$$H^-(g_j) = \begin{cases} 0; & g_j \geq 0 \\ 1; & g_j < 0 \end{cases}, \tag{4.26}$$

a K_V^* predstavlja koeficijent kaznene funkcije koji nam govori koliko "kažnjavamo" kršenje ograničenja (4.24). Drugim riječima, iz oblika funkcija (4.25) i (4.26) vidi se da funkcija (4.25) ima minimalnu vrijednost jednaku nuli u slučaju da su zadovoljena sva ograničenja (4.24), inače će funkcija imati vrijednost veću od nule, proporcionalno koeficijentu K_V^*. Ako sada pridružimo kaznenu funkciju (4.25) funkciji cilja (4.4) imamo

$$J(u_0, u_1, \ldots, u_{N-1}) = T\sum_{i=0}^{N-1}\left(F(x_i, u_i) + K_V \sum_{j=1}^{p} g_j^2(x_i, u_i) H^-\left(g_j(x_i, u_i)\right)\right) \tag{4.27}$$

gdje je $TK_V = K_V^*$. Vidimo da smo ovim problem sveli na oblik (4.4), gdje je nova 'podintegralna' funkcija

$$F'(x_i, u_i) = F(x_i, u_i) + K_V \sum_{j=1}^{p} g_j^2(x_i, u_i) H^-\left(g_j(x_i, u_i)\right), \tag{4.28}$$

a njene parcijalne derivacije

$$\frac{\partial F'(x_i, u_i)}{\partial u_i} = \frac{\partial F(x_i, u_i)}{\partial u_i} + 2K_V \sum_{j=1}^{p} g_j(x_i, u_i)\frac{\partial g_j(x_i, u_i)}{\partial u_i} H^-\left(g_j(x_i, u_i)\right), \tag{4.29}$$

$$\frac{\partial F'(x_i,u_i)}{\partial x_i}=\frac{\partial F(x_i,u_i)}{\partial x_i}+2K_V\sum_{j=1}^{p}g_j(x_i,u_i)\frac{\partial g_j(x_i,u_i)}{\partial x_i}H^{-}\Big(g_j(x_i,u_i)\Big), \quad (4.30)$$

tako da je dodatak drugog člana na desnoj strani gornjih izraza jedina modifikacija algoritma (4.12) - (4.14) u slučaju ograničenja (4.24).

Na kraju, sumirat ćemo gornje rezultate za općeniti problem optimalnog upravljanja s rubnim uvjetima i ograničenjima

$$J(u_0,u_1,\ldots,u_{N-1})=\min_u T\sum_{i=0}^{N-1}F(x_i,u_i), \quad (4.31)$$

$$x_{i+1}=f(x_i,u_i), \quad x_0=x_0, \quad x_N=x_f, \quad (4.32)$$

$$g_j\big(x_i,\,u_i\big)\geq 0, \quad (4.33)$$

za $i=0,1,\ldots,N-1$, $j=1,\ldots,p$.

Ukupna funkcija cilja s kaznenim funkcijama je

$$J(u_0,u_1,\ldots,u_{N-1})=\min_u T\sum_{i=0}^{N-1}\left(F(x_i,u_i)+K_V\sum_{j=1}^{p}g_j^2(x_i,u_i)H^{-}\Big(g_j(x_i,u_i)\Big)\right)+$$
$$+K_B\Big(x_N-x_f\Big)^2 \quad (4.33a)$$

Gradijent gornje funkcije cilja u *k*-tom koraku gradijentnog algoritma (4.6) dobiva se na osnovi slijedećeg algoritma

1. ***Inicijalizacija gradijentnog algoritma****. Za k=0 pridjelimo upravljačkim varijablama* $(u_0^{(0)},\,u_1^{(0)},\,\ldots,\,u_{N-1}^{(0)})$ *proizvoljne vrijednosti (mogu biti i izvan dozvoljenog područja u fazi računarske simulacije).*

2. ***Računanje varijabli stanja*** $(x_0^{(k)},\,x_1^{(k)},\,\ldots,\,x_N^{(k)})$ *za* $k=0,\,1,\,\ldots,\,M$ *na osnovu*

$$x_{i+1}^{(k)}=f(x_i^{(k)},u_i^{(k)}), \quad x_0^{(k)}=x_0, \quad i=0,1,\ldots,N-1;$$

3. ***Računanje gradijenta***

$$\frac{\partial J(u_0^{(k)},\,u_1^{(k)},\,\ldots,\,u_{N-1}^{(k)})}{\partial u_j^{(k)}}$$

za $k=0,\,1,\,\ldots,\,M$ *na osnovi BPTT algoritma.*

3.1 ***Inicijalizacija BPTT algoritma*** za $j = N-1$.

$$J_u(j) = T\left(F_u(j) + 2K_V \sum_{i=1}^{p} g^i(j) g_u^i(j) H^-\left(g^i(j)\right)\right) + 2K_B\left(x \ - x_f\right) f_u(j) \ ;$$

$$P(j) = 0\,;$$

$$D(j) = 1\,;$$

3.2 ***Iteriranje*** za $j = N-2,\ N-3,\ \ldots, 0$.

$$F_x'(j+1) = F_x(j+1) + 2K_V \sum_{i=1}^{p} g^i(j+1) g_x^i(j+1) H^-\left(g^i(j+1)\right),$$

$$F_u'(j+1) = F_u(j+1) + 2K_V \sum_{i=1}^{p} g^i(j+1) g_u^i(j+1) H^-\left(g^i(j+1)\right),$$

$$P(j) = F_x'(j+1) + f_x(j+1) P(j+1)\ ,$$

$$D(j) = D(j+1) f_x(j+1),$$

$$J_u(j) = T\left(F_u'(j) + P(j) f_u(j)\right) + 2K_B\left(x_N - x_f\right) D(j) f_u(j),$$

4. ***Računanje nove iteracije upravljačkih varijabli*** $(u_0^{(k+1)},\ u_1^{(k+1)},\ \ldots,\ u_{N-1}^{(k+1)})$ *na osnovu gradijentnog algoritma*

$$u_j^{(k+1)} = u_j^{(k)} - \eta \frac{\partial J(u_0^{(k)},\ u_1^{(k)},\ \ldots,\ u_{N-1}^{(k)})}{\partial u_j^{(k)}},$$

za $k = 0, 1, \ldots, M$. *Pomičemo indeks za jedan* $k \leftarrow k+1$ *i vraćamo se na 2. korak.*

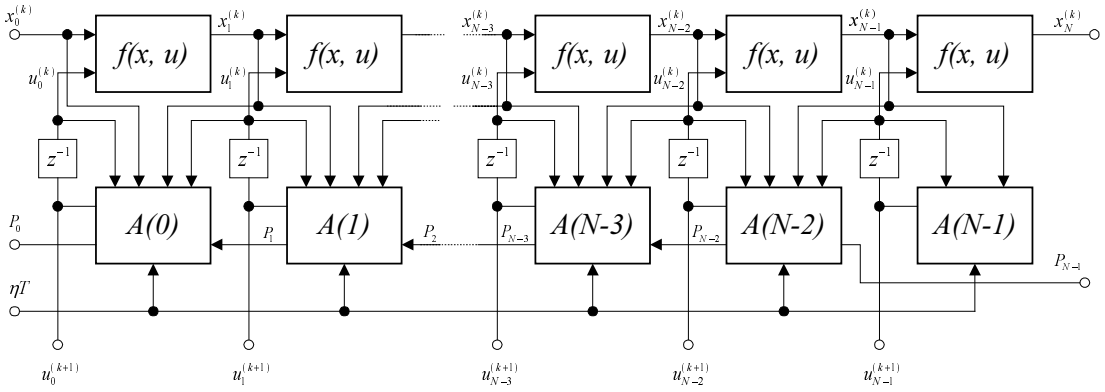
Koristili smo skraćene oznake slijedećih veličina za *k*-ti korak iteracije gradijentnog algoritma

$$F_x(i) \equiv \frac{\partial F(x_i, u_i)}{\partial x_i},\ F_u(i) \equiv \frac{\partial F(x_i, u_i)}{\partial u_i},\ f\ (i) \equiv \frac{\partial f(x_i, u_i)}{\partial x_i},\ f_u(i) \equiv \frac{\partial f(x_i, u_i)}{\partial u_i},$$

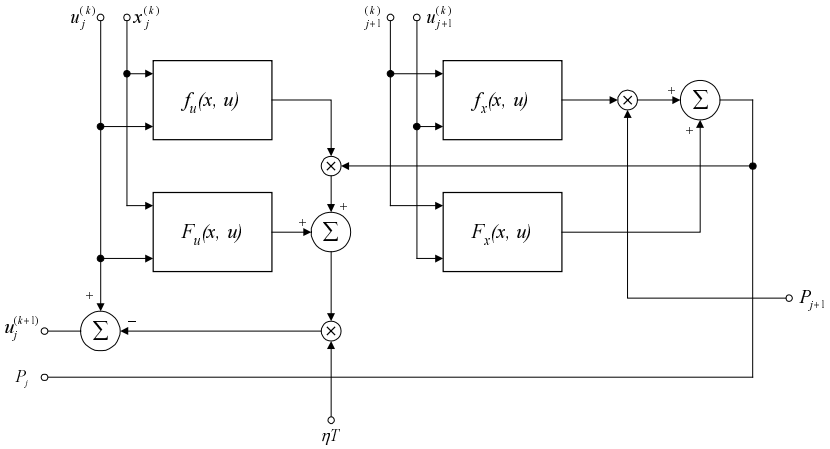
$$g_x^j(i) \equiv \frac{\partial g_j(x_i, u_i)}{\partial x_i},\ g^j(i) \equiv g_j(x_i, u_i),\ J_u(i) \equiv \frac{\partial J}{\partial u_i}.$$

Na slici 4.3 shematski je prikazan gore naveden algoritam (za slučaj bez zadanog rubnog uvjeta i ograničenja) na drugi način nego na slici 4.1. Imamo unaprijednu strukturu koja odgovara dinamici sustava (4.8) za koji tražimo optimalnu upravljačku funkciju (odgovara 2. koraku u navedenom algoritmu), a ispod nje imamo strukturu koja predstavlja 3. i 4. korak u navedenom algoritmu i ima propagaciju vrijednosti

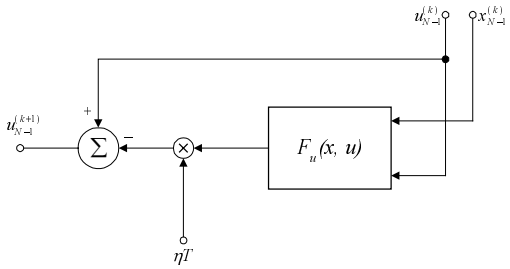
upravljačkih varijabli unazad u vremenu. Ova struktura je potpuno analogna sa strukturom unaprijednih neuronskih mreža s *N-1* skrivenih slojeva i s povratnim prostiranjem greške (*error backpropagation*) kao mehanizmom učenja (podešavanja parametara). Kad to usporedimo s prikazom na slici 4.1, možemo reći da navedeni algoritam možemo shvatiti kao dinamičku neuronsku mrežu s jednim slojem, ili kao statičku neuronsku mrežu s *N-1* skrivenih slojeva.



Slika 4.3 *Shematski prikaz BPTT algoritma kao prostiranje greške unazad u vremenu.*



Slika 4.4 *Shematski prikaz bloka A(j).*



Slika 4.5 *Shematski prikaz bloka A(N-1).*

Primjer 4.1 Razmotrimo problem optimalnog upravljanja za sustav prvog reda

$$\dot{x} = -x + u\,,$$

sa zadanim početnim i rubnim uvjetima

$$x(0) = 2\,, \qquad x(2) = 0\,,$$

i ograničenjima

$$\left| u \right| \le 1\,,$$

te s kriterijem optimalnosti

$$J = \min_u \int_0^2 \left| u \right| dt\,.$$

Analitičko rješenje ovog problema je

$$u(t) = \begin{cases} 0; & 0 \le t \le 1.685 \\ -1; & 1.685 \le t \le 2 \end{cases}, \qquad x(t) = \begin{cases} 2\mathrm{e}^{-t}\ ; & 0 \le t \le 1.685 \\ \mathrm{e}^{2-t} - 1; & 1.685 \le t \le 2 \end{cases}.$$

Ako sada primjenimo prethodno izvedeni algoritam na rješavanje ovog problema, imamo

$$F(x,u) = \left| u \right| = u \cdot \mathrm{sgn}(u)\,,$$

$$F_x(x,u) = 0\,, \qquad F_u(x,u) = \mathrm{sgn}(u)\,,$$

$$f(x,u) = x + T(-x + u)\,,$$

$$f_x(x,u) = 1 - T\,, \qquad f_u(x,u) = T\,,$$

$$g_1(x,u) = 1 - u\,, \qquad g_2(x,u) = 1 + u\,.$$

Uvrstimo li te vrijednosti u algoritam imamo

$$F_x'(j) = 0\,,$$

$$F_u'(j) = \mathrm{sgn}(u_j) + 2K_V\left((1+u_j)H^-(1+u_j) - (1-u_j)H^-(1-u_j)\right),$$

$$P(j) = (1-T)P(j+1)\ ,$$

$$D(j) = (1-T)D(j+1)\,,$$

$$J_u(j) = T\left(F_u'(j) + TP(j)\right) + 2K_B\left(x_N - x_f\right)D(j)\,,$$

za $j = N-2,\ N-3,\ \ldots, 0$.

Imajući u vidu da je za $j = N-1$

$$P(j) = 0\,; \quad D(j) = 1\,,$$

gornje jednadžbe svode se na

$$F_u'(j) = \text{sign}(u_j) + 2K_V\left((1+u_j)H^-(1+u_j) - (1-u_j)H^-(1-u_j)\right),$$

$$J_u(j) = TF_u'(j) + 2K_B\left(x_N - x_f\right)\cdot(1-T)^{N-j+1},$$

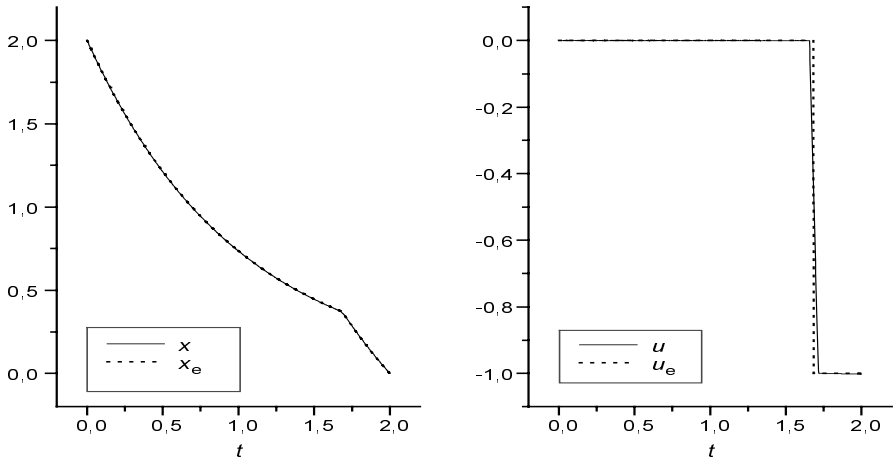
za $j = N-1,\ N-2,\ \ldots, 0$.

Na slici 4.6 prikazana su rješenja $x(t)$, $u(t)$, dobivena BPTT algoritmom, u usporedbi s analitičkim rješenjima $x_e(t)$, $u_e(t)$, dok su na slici 4.7 prikazana odstupanja analitičkih rješenja od numeričkih, $e_x = x_e(t) - x(t)$, $e_u = u_e(t) - u(t)$.

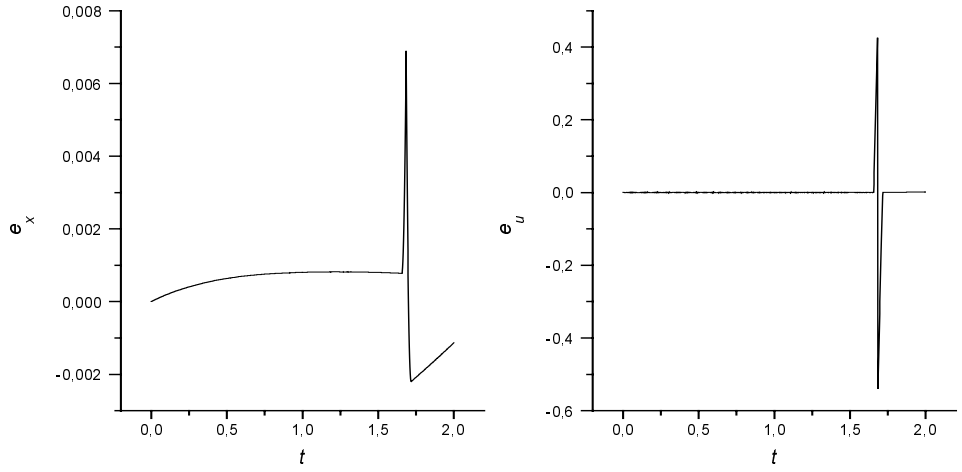
Vrijednosti numeričkih parametara su

$N = 1000$, $M = 20000$,

$\eta = 0.2$, $K_B = 600$, $K_V = 100$.



Slika 4.6 *Usporedba analitičkih rješenja (x_e, u_e) s numeričkim (x, u).*



Slika 4.7 *Odstupanja analitičkih rješenja (x_e, u_e) od numeričkih (x, u).*

Definirajmo slijedeće funkcije

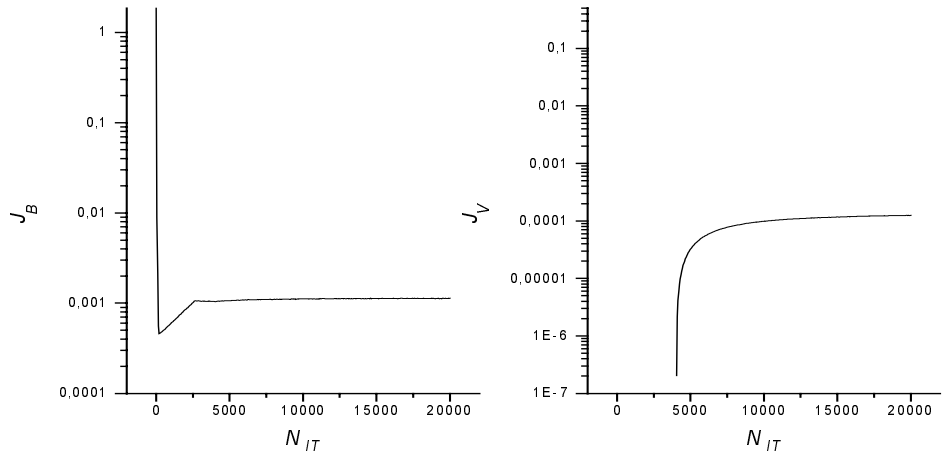
$$J_B = \left| x_N - x_f \right|,$$

$$J_V = \frac{1}{N} \sum_{j=0}^{N-1} \left(\left| 1 + u_j \right| \cdot H^-(1 + u_j) + \left| 1 - u_j \right| \cdot H^-(1 - u_j) \right),$$

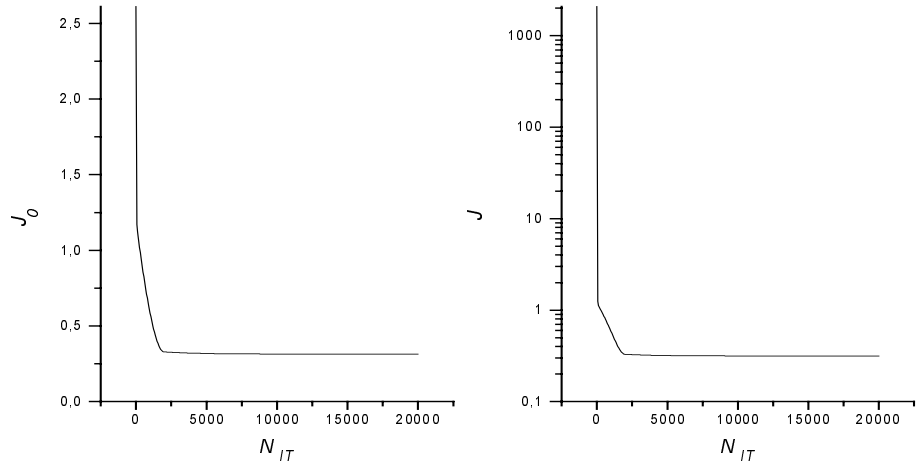
$$J_0 = T \sum_{j=0}^{N-1} \left| u_j \right|,$$

$$J = T \sum_{j=0}^{N-1} \left[\left| u_j \right| + K \left((1 + u_j)^2 H^-(1 + u_j) + (1 - u_j)^2 H^-(1 - u_j) \right) \right] + K_B (x_N - x_f)^2,$$

gdje J_B predstavlja mjeru odstupanja od rubnog uvjeta, J_V mjeri odstupanje od zadanih ograničenja, J_0 je zadana funkcija cilja, J je ukupna funkcija cilja s ograničenjima. Vrijednosti ovih funkcija, u ovisnosti o broju iteracija gradijentnog algoritma, prikazane su na slici 4.8 i 4.9.



Slika 4.8 *Vrijednosti* J_B, J_V *u ovisnosti o broju iteracija* N_{IT}.



Slika 4.9 *Vrijednosti* J_0, J *u ovisnosti o broju iteracija* N_{IT}.

4.2 Izvod BPTT algoritma za nelinearne multivarijabilne sustave

Izveli smo BPTT algoritam za nelinearne sustave prvog reda, radi jasnije ilustracije osnovne ideje algoritma, dok ćemo ovdje izvesti, po istom principu, algoritam za multivarijabilne nelinearne sustave, oblika

$$\dot{x}_k = \phi_k(x_1, x_2, \ldots, x_n, u_1, u_2, \ldots, u_m), \quad x_k(t_0) = x_0^k, \quad k = 1,2,\ldots,n, \tag{4.34}$$

ili u vektorskom obliku

$$\dot{\mathbf{x}}(t) = \phi(\mathbf{x}(t),\ \mathbf{u}(t)), \qquad \mathbf{x}(t\ ) = \mathbf{x}\ , \tag{4.35}$$

gdje je $\mathbf{x}(t)$ vektor stanja, a $\mathbf{u}(t)$ vektor upravljanja,

$$\mathbf{x} = \begin{bmatrix} x_1 & x_2 & \cdots & x_n \end{bmatrix}^T, \qquad \mathbf{u} = \begin{bmatrix} u_1 & u_2 & \cdots & u_m \end{bmatrix}^T, \tag{4.36}$$

Analogno (4.2) imamo kriterij optimalnosti

$$J = \min_{\mathbf{u}(t)} \int_{t_0}^{t_f} F\big(\mathbf{x}(t),\ \mathbf{u}(t)\big)dt\ . \tag{4.37}$$

Problem optimalnog upravljanja sastoji se u nalaženju vektorske funkcije upravljanja, $\mathbf{u}(t)$, koja minimizira kriterij optimalnosti (4.37).

Vremenskom diskretizacijom sustava diferencijalnih jednadžbi (4.34) dobivamo sustav diferencijskih jednadžbi

$$x_{i+1}^k = f^k(x_i^1, x_i^2, \ldots, x_i^n, u_i^1, u_i^2, \ldots, u_i^m), \quad k = 1,2,\ldots,n \qquad i = 0,1,\ldots,N-1 \tag{4.38}$$

odnosno

$$\mathbf{x}_{i+1} = \mathbf{f}(\mathbf{x}_i, \mathbf{u}_i), \tag{4.39}$$

gdje je

$$\mathbf{f}(\mathbf{x}_i, \mathbf{u}_i) = \mathbf{x}_i + T\phi(\mathbf{x}_i, \mathbf{u}_i), \quad \mathbf{x}_i = \begin{bmatrix} x_i^1 & x_i^2 & \cdots & x_i^n \end{bmatrix}^T, \quad \mathbf{u}_i = \begin{bmatrix} u_i^1 & u_i^2 & \cdots & u_i^m \end{bmatrix}^T,$$

dok je funkcija cilja

$$\begin{aligned} &J(u_0^1, u_0^2, \ldots, u_0^m, u_1^1, u_1^2, \ldots, u_1^m, \ldots, u_{N-1}^1, u_{N-1}^2, \ldots, u_{N-1}^m) = \\ &= \min_{\mathbf{u}} T\sum_{i=0}^{N-1} F(x_i^1, x_i^2, \ldots, x_i^n, u_i^1, u_i^2, \ldots, u_i^m) \end{aligned}, \tag{4.40}$$

odnosno

$$J(\mathbf{u}_0, \mathbf{u}_1, \ldots, \mathbf{u}_{N-1}) = \min_{\mathbf{u}} T\sum_{i=0}^{N-1} F(\mathbf{x}_i, \mathbf{u}_i)\ . \tag{4.41}$$

Gradijentni algoritam je u ovom slučaju

$$\mathbf{u}_j{}^{k+1} = \mathbf{u}_j{}^k - \eta \frac{\partial J(\mathbf{u}_1^k, \mathbf{u}_1^k, \ldots, \mathbf{u}_{N-1}^k)}{\partial \mathbf{u}_j{}^k}, \tag{4.42}$$

gdje je $j = 0,1,\ldots,N-1;\quad k = 0,1,\ldots,M$, dok je η koeficijent konvergencije, indeks *k* predstavlja *k*-tu iteraciju gradijentnog algoritma, *M* je broj iteracija algoritma.

Gradijent funkcije cilja je

$$\frac{\partial J}{\partial u_j^k} = T\frac{\partial F(\mathbf{x}_j,\mathbf{u}_j)}{\partial u_j^k} + T\sum_{i=j+1}^{N-1}\frac{\partial F(\mathbf{x}_i,\mathbf{u}_i)}{\partial u_j^k}, \tag{4.43}$$

$k = 1,2,\ldots,m;\quad j = 0,1,\ldots,N-1$, što je analogno izrazu (4.7) s tom razlikom što sada imamo *n* varijabli stanja, x_i^l, $(l = 1,2,\ldots,n)$, koje implicitno ovise o upravljačkoj varijabli u_j^k, $k = 1,2,\ldots,m$, te stoga moramo računati s totalnim diferencijalom funkcije $F(\mathbf{x}_j,\mathbf{u}_j)$ po varijablama stanja $\mathbf{x}_i$, tako da imamo

$$\frac{\partial F(\mathbf{x}_i,\mathbf{u}_i)}{\partial u_j^k} = \frac{\partial F(\mathbf{x}_i,\mathbf{u}_i)}{\partial x_i^1}\frac{\partial x_i^1}{\partial u_j^k} + \frac{\partial F(\mathbf{x}_i,\mathbf{u}_i)}{\partial x_i^2}\frac{\partial x_i^2}{\partial u_j^k} + \ldots + \frac{\partial F(\mathbf{x}_i,\mathbf{u}_i)}{\partial x_i^n}\frac{\partial x_i^n}{\partial u_j^k},$$

ili

$$\frac{\partial F(\mathbf{x}_i,\mathbf{u}_i)}{\partial u_j^k} = \sum_{r=1}^{n}\frac{\partial F(\mathbf{x}_i,\mathbf{u}_i)}{\partial x_i^r}\frac{\partial x_i^r}{\partial u_j^k}.$$

Slijedeće što treba izračunati su parcijalne derivacije $\frac{\partial x_i^p}{\partial u_j^k}$.

Imamo

$$\frac{\partial x_{j+1}^p}{\partial u_j^k} = \frac{\partial f^p(\mathbf{x}_j,\mathbf{u}_j)}{\partial u_j^k} \qquad p = 1,2,\ldots,n$$

$$\frac{\partial x_{j+2}^p}{\partial u_j^k} = \frac{\partial f^p(\mathbf{x}_{j+1},\mathbf{u}_{j+1})}{\partial x_{j+1}^1}\frac{\partial x_{j+1}^1}{\partial u_j^k} + \frac{\partial f^p(\mathbf{x}_{j+1},\mathbf{u}_{j+1})}{\partial x_{j+1}^2}\frac{\partial x_{j+1}^2}{\partial u_j^k} + \ldots + \frac{\partial f^p(\mathbf{x}_{j+1},\mathbf{u}_{j+1})}{\partial x_{j+1}^n}\frac{\partial x_{j+1}^n}{\partial u_j^k}$$

$$\vdots$$

$$\frac{\partial x_i^p}{\partial u_j^k} = \frac{\partial f^p(\mathbf{x}_{i-1},\mathbf{u}_{i-1})}{\partial x_{i-1}^1}\frac{\partial x_{i-1}^1}{\partial u_j^k} + \frac{\partial f^p(\mathbf{x}_{i-1},\mathbf{u}_{i-1})}{\partial x_{i-1}^2}\frac{\partial x_{i-1}^2}{\partial u_j^k} + \ldots + \frac{\partial f^p(\mathbf{x}_{i-1},\mathbf{u}_{i-1})}{\partial x_{i-1}^n}\frac{\partial x_{i-1}^n}{\partial u_j^k}$$

ili

$$\frac{\partial x_i^p}{\partial u_j^k} = \sum_{r=1}^{n}\frac{\partial f^p(\mathbf{x}_{i-1},\mathbf{u}_{i-1})}{\partial x_{i-1}^r}\frac{\partial x_{i-1}^r}{\partial u_j^k}$$

za $p = 1,2,\ldots,n;\quad k = 1,2,\ldots,m;\quad j = 0,1,\ldots,N-1;\quad i = j+1,\ldots,N-1$.

Sada ćemo izvesti rekurzivne relacije za izračunavanje sume u izrazu (4.43)

$$S_k(j) = \sum_{i=j+1}^{N-1}\frac{\partial F(\mathbf{x}_i,\mathbf{u}_i)}{\partial u_j^k}$$

Krenut ćemo unazad u vremenu. Imamo za $j = N-2;\ i = N-1$

$$S_k(N-2)=\frac{\partial F(\mathbf{x}_{N-1},\mathbf{u}_{N-1})}{\partial u_{N-2}^k}=\sum_{r=1}^{n}\frac{\partial F(\mathbf{x}_{N-1},\mathbf{u}_{N-1})}{\partial x_{N-1}^r}\frac{\partial x_{N-1}^r}{\partial u_{N-2}^k}$$

gdje je

$$\frac{\partial x_{N-1}^r}{\partial u_{N-2}^k}=\frac{\partial f^r(\mathbf{x}_{N-2},\mathbf{u}_{N-2})}{\partial u_{N-2}^k}$$

odnosno

$$S_k(N-2)=\sum_{r=1}^{n}\frac{\partial F(\mathbf{x}_{N-1},\mathbf{u}_{N-1})}{\partial x_{N-1}^r}\frac{\partial f^r(\mathbf{x}_{N-2},\mathbf{u}_{N-2})}{\partial u_{N-2}^k}$$

Nadalje za $j=N-3;\ \ i=N-1,\ N-2$

$$S_k(N-3)=\frac{\partial F(\mathbf{x}_{N-2},\mathbf{u}_{N-2})}{\partial u_{N-3}^k}+\frac{\partial F(\mathbf{x}_{N-1},\mathbf{u}_{N-1})}{\partial u_{N-3}^k}=$$
$$=\sum_{r=1}^{n}\frac{\partial F(\mathbf{x}_{N-2},\mathbf{u}_{N-2})}{\partial x_{N-2}^r}\frac{\partial x_{N-2}^r}{\partial u_{N-3}^k}+\sum_{r=1}^{n}\frac{\partial F(\mathbf{x}_{N-1},\mathbf{u}_{N-1})}{\partial x_{N-1}^r}\frac{\partial x_{N-1}^r}{\partial u_{N-3}^k}$$

Pošto je

$$\frac{\partial x_{N-1}^r}{\partial u_{N-3}^k}=\sum_{l=1}^{n}\frac{\partial f^r(\mathbf{x}_{N-2},\mathbf{u}_{N-2})}{\partial x_{N-2}^l}\frac{\partial x_{N-2}^l}{\partial u_{N-3}^k}=\sum_{l=1}^{n}\frac{\partial f^r(\mathbf{x}_{N-2},\mathbf{u}_{N-2})}{\partial x_{N-2}^l}\frac{\partial f^l(\mathbf{x}_{N-3},\mathbf{u}_{N-3})}{\partial u_{N-3}^k}$$

i

$$\frac{\partial x_{N-2}^r}{\partial u_{N-3}^k}=\frac{\partial f^r(\mathbf{x}_{N-3},\mathbf{u}_{N-3})}{\partial u_{N-3}^k}$$

imamo

$$S_k(N-3)=\sum_{r=1}^{n}\frac{\partial F(\mathbf{x}_{N-2},\mathbf{u}_{N-2})}{\partial x_{N-2}^r}\frac{\partial f^r(\mathbf{x}_{N-3},\mathbf{u}_{N-3})}{\partial u_{N-3}^k}+$$
$$+\sum_{r=1}^{n}\sum_{l=1}^{n}\frac{\partial F(\mathbf{x}_{N-1},\mathbf{u}_{N-1})}{\partial x_{N-1}^r}\frac{\partial f^r(\mathbf{x}_{N-2},\mathbf{u}_{N-2})}{\partial x_{N-2}^l}\frac{\partial f^l(\mathbf{x}_{N-3},\mathbf{u}_{N-3})}{\partial u_{N-3}^k}$$

odnosno

$$S_k(N-3)=\sum_{r=1}^{n}\frac{\partial F(\mathbf{x}_{N-2},\mathbf{u}_{N-2})}{\partial x_{N-2}^r}\frac{\partial f^r(\mathbf{x}_{N-3},\mathbf{u}_{N-3})}{\partial u_{N-3}^k}+$$
$$+\sum_{l=1}^{n}\frac{\partial f^l(\mathbf{x}_{N-3},\mathbf{u}_{N-3})}{\partial u_{N-3}^k}\sum_{r=1}^{n}\frac{\partial F(\mathbf{x}_{N-1},\mathbf{u}_{N-1})}{\partial x_{N-1}^r}\frac{\partial f^r(\mathbf{x}_{N-2},\mathbf{u}_{N-2})}{\partial x_{N-2}^l}$$

i poslije zamjene indeksa $r\to l,\ \ l\to r$

$$S_k(N-3)=\sum_{r=1}^{n}\frac{\partial F(\mathbf{x}_{N-2},\mathbf{u}_{N-2})}{\partial x_{N-2}^r}\frac{\partial f^r(\mathbf{x}_{N-3},\mathbf{u}_{N-3})}{\partial u_{N-3}^k}+$$
$$+\sum_{r=1}^{n}\frac{\partial f^r(\mathbf{x}_{N-3},\mathbf{u}_{N-3})}{\partial u_{N-3}^k}\sum_{l=1}^{n}\frac{\partial F(\mathbf{x}_{N-1},\mathbf{u}_{N-1})}{\partial x_{N-1}^l}\frac{\partial f^l(\mathbf{x}_{N-2},\mathbf{u}_{N-2})}{\partial x_{N-2}^r}$$

dobivamo

$$S_k(N-3)=\sum_{r=1}^{n}\frac{\partial f^r(\mathbf{x}_{N-3},\mathbf{u}_{N-3})}{\partial u_{N-3}^k}\left[\frac{\partial F(\mathbf{x}_{N-2},\mathbf{u}_{N-2})}{\partial x_{N-2}^r}+\sum_{l=1}^{n}\frac{\partial F(\mathbf{x}_{N-1},\mathbf{u}_{N-1})}{\partial x_{N-1}^l}\frac{\partial f^l(\mathbf{x}_{N-2},\mathbf{u}_{N-2})}{\partial x_{N-2}^r}\right]$$

Idemo dalje ponoviti analognu proceduru za još jedan korak
$j=N-4;\ \ i=N-1,\ N-2,\ N-3$

$$S_k(N-4)=\frac{\partial F(\mathbf{x}_{N-3},\mathbf{u}_{N-3})}{\partial u_{N-4}^k}+\frac{\partial F(\mathbf{x}_{N-2},\mathbf{u}_{N-2})}{\partial u_{N-4}^k}+\frac{\partial F(\mathbf{x}_{N-1},\mathbf{u}_{N-1})}{\partial u_{N-4}^k}=$$

$$=\sum_{r=1}^{n}\frac{\partial F(\mathbf{x}_{N-3},\mathbf{u}_{N-3})}{\partial x_{N-3}^r}\frac{\partial x_{N-3}^r}{\partial u_{N-4}^k}+\sum_{r=1}^{n}\frac{\partial F(\mathbf{x}_{N-2},\mathbf{u}_{N-2})}{\partial x_{N-2}^r}\frac{\partial x_{N-2}^r}{\partial u_{N-4}^k}+\sum_{r=1}^{n}\frac{\partial F(\mathbf{x}_{N-1},\mathbf{u}_{N-1})}{\partial x_{N-1}^r}\frac{\partial x_{N-1}^r}{\partial u_{N-4}^k}$$

Imamo nadalje

$$S_k(N-4)=\sum_{r=1}^{n}\frac{\partial F(\mathbf{x}_{N-3},\mathbf{u}_{N-3})}{\partial x_{N-3}^r}\frac{\partial f^r(\mathbf{x}_{N-4},\mathbf{u}_{N-4})}{\partial u_{N-4}^k}+$$

$$+\sum_{r=1}^{n}\sum_{l=1}^{n}\frac{\partial F(\mathbf{x}_{N-2},\mathbf{u}_{N-2})}{\partial x_{N-2}^r}\frac{\partial f^r(\mathbf{x}_{N-3},\mathbf{u}_{N-3})}{\partial x_{N-3}^l}\frac{\partial f^l(\mathbf{x}_{N-4},\mathbf{u}_{N-4})}{\partial u_{N-4}^k}+$$

$$+\sum_{r=1}^{n}\sum_{l=1}^{n}\sum_{q=1}^{n}\frac{\partial F(\mathbf{x}_{N-1},\mathbf{u}_{N-1})}{\partial x_{N-1}^r}\frac{\partial f^r(\mathbf{x}_{N-2},\mathbf{u}_{N-2})}{\partial x_{N-2}^l}\frac{\partial f^l(\mathbf{x}_{N-3},\mathbf{u}_{N-3})}{\partial x_{N-3}^q}\frac{\partial f^q(\mathbf{x}_{N-4},\mathbf{u}_{N-4})}{\partial u_{N-4}^k}$$

Poslije zamjene indeksa $r\to l,\ \ l\to r$ u drugoj sumi i $q\to r,\ r\to q$ u trećoj imamo

$$S_k(N-4)=\sum_{r=1}^{n}\frac{\partial f^r(\mathbf{x}_{N-4},\mathbf{u}_{N-4})}{\partial u_{N-4}^k}\left[\frac{\partial F(\mathbf{x}_{N-3},\mathbf{u}_{N-3})}{\partial x_{N-3}^r}+\right.$$

$$\left.+\sum_{l=1}^{n}\frac{\partial f^l(\mathbf{x}_{N-3},\mathbf{u}_{N-3})}{\partial x_{N-3}^r}\left[\frac{\partial F(\mathbf{x}_{N-2},\mathbf{u}_{N-2})}{\partial x_{N-2}^r}+\sum_{q=1}^{n}\frac{\partial F(\mathbf{x}_{N-1},\mathbf{u}_{N-1})}{\partial x_{N-1}^q}\frac{\partial f^q(\mathbf{x}_{N-2},\mathbf{u}_{N-2})}{\partial x_{N-2}^l}\right]\right]$$

Ako sada uvedemo oznake

$$P^r(N-2)=\frac{\partial F(\mathbf{x}_{N-1},\mathbf{u}_{N-1})}{\partial x_{N-1}^r},$$

$$P^r(N-3)=\frac{\partial F(\mathbf{x}_{N-2},\mathbf{u}_{N-2})}{\partial x_{N-2}^r}+\sum_{l=1}^{n}\frac{\partial f^l(\mathbf{x}_{N-2},\mathbf{u}_{N-2})}{\partial x_{N-2}^r}P^l(N-2)$$

$$P^r(N-4)=\frac{\partial F(\mathbf{x}_{N-3},\mathbf{u}_{N-3})}{\partial x_{N-3}^r}+\sum_{l=1}^{n}\frac{\partial f^l(\mathbf{x}_{N-3},\mathbf{u}_{N-3})}{\partial x_{N-3}^r}P^l(N-3)$$

imamo

$$S_k(N-2)=\sum_{r=1}^{n}\frac{\partial f^r(\mathbf{x}_{N-2},\mathbf{u}_{N-2})}{\partial u_{N-2}^k}P^r(N-2)$$

$$S_k(N-3)=\sum_{r=1}^{n}\frac{\partial f^r(\mathbf{x}_{N-3},\mathbf{u}_{N-3})}{\partial u_{N-3}^k}P^r(N-3)$$

$$S_k(N-4) = \sum_{r=1}^{n} \frac{\partial f^r(\mathbf{x}_{N-4}, \mathbf{u}_{N-4})}{\partial u_{N-4}^k} P^r(N-4)$$

Sada možemo postupak generalizirati u obliku algoritma

$$P^r(j) = \frac{\partial F(\mathbf{x}_{j+1}, \mathbf{u}_{j+1})}{\partial x_{j+1}^r} + \sum_{l=1}^{n} \frac{\partial f^l(\mathbf{x}_{j+1}, \mathbf{u}_{j+1})}{\partial x_{j+1}^r} P^l(j+1); \quad P^r(N-1) = 0, \tag{4.44}$$

$$S_k(j) = \sum_{r=1}^{n} \frac{\partial f^r(\mathbf{x}_j, \mathbf{u}_j)}{\partial u_j^k} P^r(j), \tag{4.45}$$

gdje je $r = 1,2,\ldots,n; \quad j = N-2, N-3,\ldots,0$.

Na kraju možemo navesti kompletan algoritam optimalnog upravljanja

$$u_j^k(i+1) = u_j^k(i) - \eta(i) \frac{\partial J}{\partial u_j^k(i)}, \qquad i = 0,1,\ldots,M, \tag{4.46}$$

Gradijent za *i*-tu iteraciju je

$$\frac{\partial J}{\partial u_j^k} = T\left(\frac{\partial F(\mathbf{x}_j, \mathbf{u}_j)}{\partial u_j^k} + S_k(j) \right),$$

$$S_k(j) = \sum_{r=1}^{n} \frac{\partial f^r(\mathbf{x}_j, \mathbf{u}_j)}{\partial u_j^k} P^r(j), \tag{4.47}$$

$$P^r(j) = \frac{\partial F(\mathbf{x}_{j\ 1}, \mathbf{u}_{j\ 1})}{\partial x_{j\ 1}^r} + \sum_{=1}^{n} \frac{\partial f\ (\mathbf{x}_{j\ 1}, \mathbf{u}_{j\ 1})}{\partial x_{j\ 1}^r} P\ (j+1)\ ; \quad P^r(N-1) = 0,$$

za $r = 1,2,\ldots,n; \quad k = 1,2,\ldots,m; \quad j = N-2, N-3,\ldots,0$.

U matričnoj interpretaciji, uvodeći

$$\mathbf{U}(j) = \begin{bmatrix} \frac{\partial f^1}{\partial u_j^1} & \frac{\partial f^2}{\partial u_j^1} & \cdots & \frac{\partial f^n}{\partial u_j^1} \\ \frac{\partial f^1}{\partial u_j^2} & \frac{\partial f^2}{\partial u_j^2} & \cdots & \frac{\partial f^n}{\partial u_j^2} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial f^1}{\partial u_j^m} & \frac{\partial f^2}{\partial u_j^m} & \cdots & \frac{\partial f^n}{\partial u_j^m} \end{bmatrix}, \qquad \mathbf{X}(j) = \begin{bmatrix} \frac{\partial f^1}{\partial x_j^1} & \frac{\partial f^2}{\partial x_j^1} & \cdots & \frac{\partial f^n}{\partial x_j^1} \\ \frac{\partial f^1}{\partial x_j^2} & \frac{\partial f^2}{\partial x_j^2} & \cdots & \frac{\partial f^n}{\partial x_j^2} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial f^1}{\partial x_j^n} & \frac{\partial f^2}{\partial x_j^n} & \cdots & \frac{\partial f^n}{\partial x_j^n} \end{bmatrix}, \tag{4.47a}$$

$$\mathbf{S}(j)=\begin{bmatrix} S_1(j) \\ S_2(j) \\ \vdots \\ S_m(j) \end{bmatrix},\quad \mathbf{P}(j)=\begin{bmatrix} P^1(j) \\ P^2(j) \\ \vdots \\ P^n(j) \end{bmatrix},\quad \mathbf{F}_x(j)=\begin{bmatrix} \dfrac{\partial F}{\partial x_j^1} \\ \dfrac{\partial F}{\partial x_j^2} \\ \vdots \\ \dfrac{\partial F}{\partial x_j^n} \end{bmatrix},\quad \mathbf{F}_u(j)=\begin{bmatrix} \dfrac{\partial F}{\partial u_j^1} \\ \dfrac{\partial F}{\partial u_j^2} \\ \vdots \\ \dfrac{\partial F}{\partial u_j^m} \end{bmatrix},\quad \mathbf{J}_u(j)=\begin{bmatrix} \dfrac{\partial J}{\partial u_j^1} \\ \dfrac{\partial J}{\partial u_j^2} \\ \vdots \\ \dfrac{\partial J}{\partial u_j^m} \end{bmatrix},$$

algoritam (4.47) poprima oblik

$$\begin{aligned} &\mathbf{J}_u(j)=T\big(\mathbf{F}_u(j)+\mathbf{S}(j)\big), \qquad \mathbf{J}_u(N-1)=T\mathbf{F}_u(N-1), \\ &\mathbf{S}(j)=\mathbf{U}(j)\cdot\mathbf{P}(j), \\ &\mathbf{P}(j)=\mathbf{F}_x(j+1)+\mathbf{X}(j+1)\cdot\mathbf{P}(j+1), \qquad \mathbf{P}(N-1)=\mathbf{0}, \end{aligned} \tag{4.48}$$

za $j=N-2,N-3,\ldots,0$.

4.3 Metoda kaznenih funkcija za ograničenja tipa rubnih uvjeta

Gore navedeni algoritam izveden je za problem optimalnog upravljanja za dinamiku nelinearnih sustava sa zadanim početnim uvjetima (4.35). Sada ćemo razmotriti problem sa zadanim rubnim uvjetima

$$\dot{\mathbf{x}}(t)=\phi\big(\mathbf{x}(t),\ \mathbf{u}(t)\big), \qquad \mathbf{x}(t_0)=\mathbf{x}_0, \qquad \mathbf{x}(t_f)=\mathbf{x}_f, \tag{4.49}$$

Da bi uračunali rubni uvjet, dodat ćemo funkciji cilja (4.41) kaznenu funkciju u obliku

$$G_B(\mathbf{u}_0,\mathbf{u}_1,\ldots,\mathbf{u}_{N-1})=K_B\sum_{k=1}^{n}\left(x_N^k-x_f^k\right)^2, \tag{4.50}$$

gdje je K_B koeficijent kaznene funkcije, a x_f^k su zadane vrijednosti varijabli stanja na kraju vremenskog intervala.

Parcijalna derivacija kaznene funkcije po upravljačkoj varijabli je

$$\frac{\partial G_B}{\partial u_j^p}=\sum_{k=1}^{n}\frac{\partial G_B}{\partial x_N^k}\frac{\partial x_N^k}{\partial u_j^p}. \tag{4.51}$$

Sada nam preostaje odrediti parcijalne derivacije varijabli stanja po upravljačkim varijablama, počevši od kraja vremenskog intervala

$$\frac{\partial x_N^k}{\partial u_{N-1}^p}=\frac{\partial f^k(\mathbf{x}_{N-1},\mathbf{u}_{N-1})}{\partial u_{N-1}^p}$$

$$\frac{\partial x_N^k}{\partial u_{N-2}^p} = \sum_{r=1}^{n} \frac{\partial f^k(\mathbf{x}_{N-1}, \mathbf{u}_{N-1})}{\partial x_{N-1}^r} \frac{\partial x_{N-1}^r}{\partial u_{N-2}^p} = \sum_{r=1}^{n} \frac{\partial f^k(\mathbf{x}_{N-1}, \mathbf{u}_{N-1})}{\partial x_{N-1}^r} \frac{\partial f^r(\mathbf{x}_{N-2}, \mathbf{u}_{N-2})}{\partial u_{N-2}^p} \tag{4.51a}$$

$$\frac{\partial x_N^k}{\partial u_{N-3}^p} = \sum_{r=1}^{n} \frac{\partial f^k(\mathbf{x}_{N-1}, \mathbf{u}_{N-1})}{\partial x_{N-1}^r} \frac{\partial x_{N-1}^r}{\partial u_{N-3}^p} = \sum_{r=1}^{n} \frac{\partial f^k(\mathbf{x}_{N-1}, \mathbf{u}_{N-1})}{\partial x_{N-1}^r} \frac{\partial f^r(\mathbf{x}_{N-2}, \mathbf{u}_{N-2})}{\partial u_{N-3}^p} =$$

$$= \sum_{r=1}^{n} \frac{\partial f^k(\mathbf{x}_{N-1}, \mathbf{u}_{N-1})}{\partial x_{N-1}^r} \sum_{q=1}^{n} \frac{\partial f^r(\mathbf{x}_{N-2}, \mathbf{u}_{N-2})}{\partial x_{N-2}^q} \frac{\partial x_{N-2}^q}{\partial u_{N-3}^p} =$$

$$= \sum_{r=1}^{n} \frac{\partial f^k(\mathbf{x}_{N-1}, \mathbf{u}_{N-1})}{\partial x_{N-1}^r} \sum_{q=1}^{n} \frac{\partial f^r(\mathbf{x}_{N-2}, \mathbf{u}_{N-2})}{\partial x_{N-2}^q} \frac{\partial f^q(\mathbf{x}_{N-3}, \mathbf{u}_{N-3})}{\partial u_{N-3}^p}$$

Ako uvedemo matričnu notaciju, koristeći oznake matrica iz (4.47a), i uvodeći slijedeće matrice

$$\mathbf{X}_u(j) = \begin{bmatrix} \frac{\partial x_N^1}{\partial u_j^1} & \frac{\partial x_N^1}{\partial u_j^2} & \cdots & \frac{\partial x_N^1}{\partial u_j^m} \\ \frac{\partial x_N^2}{\partial u_j^1} & \frac{\partial x_N^2}{\partial u_j^2} & \cdots & \frac{\partial x_N^2}{\partial u_j^m} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial x_N^n}{\partial u_j^1} & \frac{\partial x_N^n}{\partial u_j^2} & \cdots & \frac{\partial x_N^n}{\partial u_j^m} \end{bmatrix}, \quad \mathbf{G}_u(j) = \begin{bmatrix} \frac{\partial G_B}{\partial u_j^1} \\ \frac{\partial G_B}{\partial u_j^2} \\ \vdots \\ \frac{\partial G_B}{\partial u_j^m} \end{bmatrix}, \quad \mathbf{G}_x(N) = \begin{bmatrix} \frac{\partial G_B}{\partial x_N^1} \\ \frac{\partial G_B}{\partial x_N^2} \\ \vdots \\ \frac{\partial G_B}{\partial x_N^n} \end{bmatrix}, \tag{4.51b}$$

možemo napisati

$$\begin{aligned} &\mathbf{X}_u(N-1) = \mathbf{U}^T(N-1) \\ &\mathbf{X}_u(N-2) = \mathbf{X}^T(N-1) \cdot \mathbf{U}^T(N-2) \\ &\mathbf{X}_u(N-3) = \mathbf{X}^T(N-1) \cdot \mathbf{X}^T(N-2) \cdot \mathbf{U}^T(N-3) \\ &\mathbf{X}_u(N-4) = \mathbf{X}^T(N-1) \cdot \mathbf{X}^T(N-2) \cdot \mathbf{X}^T(N-3) \cdot \mathbf{U}^T(N-4) \\ &\vdots \\ &\mathbf{X}_u(j) = \mathbf{X}^T(N-1) \cdot \mathbf{X}^T(N-2) \cdot \ldots \cdot \mathbf{X}^T(j+1) \cdot \mathbf{U}^T(j) \end{aligned} \tag{4.51c}$$

gdje su $\mathbf{U}^T$ i $\mathbf{X}^T$ transponirane matrice.

Gornji sustav, zajedno sa (4.51) možemo preglednije zapisati u obliku

$$\mathbf{G}_u(j) = \mathbf{X}_u^T(j)\cdot\mathbf{G}_x(N)\ , \qquad \mathbf{G}_u(N-1) = \mathbf{U}(N-1)\cdot\mathbf{G}_x(N)\ ,$$
$$\mathbf{X}_u(j) = \mathbf{D}(j)\cdot\mathbf{U}^T(j), \tag{4.52}$$
$$\mathbf{D}(j) = \mathbf{D}(j+1)\cdot\mathbf{X}^T(j+1), \quad \mathbf{D}(N-1) = \mathbf{I},$$

za $j = N-2, N-3, \ldots, 0$.

4.4 Metoda kaznenih funkcija za ograničenja tipa nejednakosti

Za ograničenja tipa nejednakosti

$$\mathbf{g}\big(\mathbf{x}(t),\, \mathbf{u}(t)\big) \geq 0\,, \tag{4.53}$$

ili za diskretizirani slučaj i napisano po komponentama

$$g\ (\mathbf{x}_i,\, \mathbf{u}_i) \geq 0\,, \tag{4.54}$$

za $i = 0,1,\ldots,N-1; \quad j = 1,2,\ldots,p$,

također možemo koristiti metodu kaznenih funkcija u obliku

$$G_V(\mathbf{u}_0, \mathbf{u}_1, \ldots, \mathbf{u}_{N-1}) = K_V^* \sum_{i=0}^{N-1}\sum_{j=1}^{p} g_j^2(\mathbf{x}_i,\, \mathbf{u}_i) H^-\Big(g_j(\mathbf{x}_i,\, \mathbf{u}_i)\Big), \tag{4.55}$$

gdje je funkcija $H^-(x)$ definirana prema (4.26), a konstanta K_V^* predstavlja veličinu "kazne" za prekoračenje ograničenja.

Sa ovom kaznenom funkcijom ukupna funkcija cilja ima slijedeći oblik

$$J(\mathbf{u}_0, \mathbf{u}_1, \ldots, \mathbf{u}_{N-1}) = \min_{\mathbf{u}} T\sum_{i=0}^{N-1}\left[F(\mathbf{x}_i,\, \mathbf{u}_i) + K_V \sum_{j=1}^{p} g_j^2(\mathbf{x}_i,\, \mathbf{u}_i) H^-\Big(g_j(\mathbf{x}_i,\, \mathbf{u}_i)\Big)\right], \tag{4.56}$$

gdje je $K_V^* = TK_V$. Sada 'podintegralna funkcija' ima slijedeći oblik

$$F'(\mathbf{x}_i,\, \mathbf{u}_i) = F(\mathbf{x}_i,\, \mathbf{u}_i) + K_V \sum_{j=1}^{p} g_j^2(\mathbf{x}_i,\, \mathbf{u}_i) H^-\Big(g_j(\mathbf{x}_i,\, \mathbf{u}_i)\Big), \tag{4.57}$$

a njene parcijalne derivacije su

$$\frac{\partial F'(\mathbf{x}_i,\, \mathbf{u}_i)}{\partial u_i^k} = \frac{\partial F(\mathbf{x}_i,\, \mathbf{u}_i)}{\partial u_i^k} + 2K_V \sum_{j=1}^{p} g_j(\mathbf{x}_i,\, \mathbf{u}_i) \frac{\partial g_j(\mathbf{x}_i,\, \mathbf{u}_i)}{\partial u_i^k} H^-\Big(g_j(\mathbf{x}_i,\, \mathbf{u}_i)\Big), \tag{4.57a}$$

$$\frac{\partial F'(\mathbf{x}_i,\, \mathbf{u}_i)}{\partial x_i^k} = \frac{\partial F(\mathbf{x}_i,\, \mathbf{u}_i)}{\partial x_i^k} + 2K_V \sum_{j=1}^{p} g_j(\mathbf{x}_i,\, \mathbf{u}_i) \frac{\partial g_j(\mathbf{x}_i,\, \mathbf{u}_i)}{\partial x_i^k} H^-\Big(g_j(\mathbf{x}_i,\, \mathbf{u}_i)\Big). \tag{4.57b}$$

U matričnoj notaciji, ako stavimo

$$\mathbf{V}_u(i)=\begin{bmatrix} \frac{\partial g_1}{\partial u_i^1} & \frac{\partial g_2}{\partial u_i^1} & \cdots & \frac{\partial g_p}{\partial u_i^1} \\ \frac{\partial g_1}{\partial u_i^2} & \frac{\partial g_2}{\partial u_i^2} & \cdots & \frac{\partial g_p}{\partial u_i^2} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial g_1}{\partial u_i^m} & \frac{\partial g_2}{\partial u_i^m} & \cdots & \frac{\partial g_p}{\partial u_i^m} \end{bmatrix}, \qquad \mathbf{V}_x(i)=\begin{bmatrix} \frac{\partial g_1}{\partial x_i^1} & \frac{\partial g_2}{\partial x_i^1} & \cdots & \frac{\partial g_p}{\partial x_i^1} \\ \frac{\partial g_1}{\partial x_i^2} & \frac{\partial g_2}{\partial x_i^2} & \cdots & \frac{\partial g_p}{\partial x_i^2} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial g_1}{\partial x_i^n} & \frac{\partial g_2}{\partial x_i^n} & \cdots & \frac{\partial g_p}{\partial x_i^n} \end{bmatrix},$$

$$\mathbf{g}(i)=\begin{bmatrix} g_1(\mathbf{x}_i,\mathbf{u}_i)H^{-}\big((g_1(\mathbf{x}_i,\mathbf{u}_i)\big) \\ g_2(\mathbf{x}_i,\mathbf{u}_i)H^{-}\big((g_2(\mathbf{x}_i,\mathbf{u}_i)\big) \\ \vdots \\ g_p(\mathbf{x}_i,\mathbf{u}_i)H^{-}\big((g_p(\mathbf{x}_i,\mathbf{u}_i)\big) \end{bmatrix}, \tag{4.57c}$$

možemo gornje izraze napisati u obliku

$$\begin{aligned} \mathbf{F}_u'(i)&=\mathbf{F}_u(i)+2K_V\mathbf{V}_u(i)\cdot\mathbf{g}(i) \\ \mathbf{F}_x'(i)&=\mathbf{F}_x(i)+2K_V\mathbf{V}_x(i)\cdot\mathbf{g}(i) \end{aligned}, \tag{4.57d}$$

Iz ovoga slijedi da jedina modifikacija koju trebamo učiniti u algoritmu je da vektorima $\mathbf{F}_u(i)$ i $\mathbf{F}_x(i)$ dodamo drugi član na desnoj strani prethodnih izraza, koji odgovara derivaciji kaznene funkcije.

4.5 Metoda kaznenih funkcija za ograničenja tipa jednakosti

Na kraju, razmotrit ćemo slučaj ograničenja tipa jednakosti

$$\mathbf{h}\big(\mathbf{x}(t),\,\mathbf{u}(t)\big)=0\,, \tag{4.58}$$

ili za diskretizirani slučaj i napisano po komponentama

$$h_j(\mathbf{x}_i,\,\mathbf{u}_i)=0\,, \tag{4.59}$$

za $i=0,1,\ldots,N-1;\quad j=1,2,\ldots,r$,

Kaznena funkcija je u obliku

$$G_E(\mathbf{u}_0,\mathbf{u}_1,\ldots,\mathbf{u}_{N-1})=K_E^*\sum_{i=0}^{N-1}\sum_{j=1}^{r}h_j^2(\mathbf{x}_i,\,\mathbf{u}_i)\,, \tag{4.60}$$

gdje je K_E^* koeficijent proporcionalnosti uz kaznenu funkciju. Vidimo da gornja kaznena funkcija ima minimalnu vrijednost jednaku nuli, u slučaju kada su zadovoljena ograničenja tipa jednakosti.

Sa ovom kaznenom funkcijom ukupna funkcija cilja ima slijedeći oblik

$$J(\mathbf{u}_0, \mathbf{u}_1, ..., \mathbf{u}_{N-1}) = \min_{\mathbf{u}} T \sum_{i=0}^{N-1} \left[F(\mathbf{x}_i,\ \mathbf{u}_i) + K_E \sum_{j=1}^{r} h_j^2(\mathbf{x}_i,\ \mathbf{u}_i) \right], \tag{4.61}$$

gdje je $K_E^* = TK_E$. Sada 'podintegralna funkcija' ima slijedeći oblik

$$F'(\mathbf{x}\ ,\ \mathbf{u}\ ) = F(\mathbf{x}\ ,\ \mathbf{u}\ ) + K_E \sum_{=} h\ (\mathbf{x}\ ,\ \mathbf{u}\ ), \tag{4.62}$$

a njene parcijalne derivacije su

$$\frac{\partial F'(\mathbf{x}_i, \mathbf{u}_i)}{\partial u_i^k} = \frac{\partial F(\mathbf{x}_i, \mathbf{u}_i)}{\partial u_i^k} + 2K_E \sum_{j=1}^{r} h_j(\mathbf{x}_i, \mathbf{u}_i) \frac{\partial h_j(\mathbf{x}_i, \mathbf{u}_i)}{\partial u_i^k}, \tag{4.62a}$$

$$\frac{\partial F'(\mathbf{x}_i, \mathbf{u}_i)}{\partial x_i^k} = \frac{\partial F(\mathbf{x}_i, \mathbf{u}_i)}{\partial x_i^k} + 2K_E \sum_{j=1}^{r} h_j(\mathbf{x}_i, \mathbf{u}_i) \frac{\partial h_j(\mathbf{x}_i, \mathbf{u}_i)}{\partial x_i^k}. \tag{4.62b}$$

U matričnoj notaciji, stavljajući

$$\mathbf{E}_u(i) = \begin{bmatrix} \frac{\partial h_1}{\partial u_i^1} & \frac{\partial h_2}{\partial u_i^1} & \cdots & \frac{\partial h_r}{\partial u_i^1} \\ \frac{\partial h_1}{\partial u_i^2} & \frac{\partial h_2}{\partial u_i^2} & \cdots & \frac{\partial h_r}{\partial u_i^2} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial h_1}{\partial u_i^m} & \frac{\partial h_2}{\partial u_i^m} & \cdots & \frac{\partial h_r}{\partial u_i^m} \end{bmatrix}, \quad \mathbf{E}_x(i) = \begin{bmatrix} \frac{\partial h_1}{\partial x_i^1} & \frac{\partial h_2}{\partial x_i^1} & \cdots & \frac{\partial h_r}{\partial x_i^1} \\ \frac{\partial h_1}{\partial x_i^2} & \frac{\partial h_2}{\partial x_i^2} & \cdots & \frac{\partial h_r}{\partial x_i^2} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial h_1}{\partial x_i^n} & \frac{\partial h_2}{\partial x_i^n} & \cdots & \frac{\partial h_r}{\partial x_i^n} \end{bmatrix}, \tag{4.62c}$$

$$\mathbf{h}(i) = \begin{bmatrix} h_1(\mathbf{x}_i, \mathbf{u}_i) \\ h_2(\mathbf{x}_i, \mathbf{u}_i) \\ \vdots \\ h_r(\mathbf{x}_i, \mathbf{u}_i) \end{bmatrix},$$

možemo gornje izraze napisati u obliku

$$\begin{aligned} \mathbf{F}_u'(i) &= \mathbf{F}_u(i) + 2K_V \mathbf{E}_u(i) \cdot \mathbf{h}(i) \\ \mathbf{F}_x'(i) &= \mathbf{F}_x(i) + 2K_V \mathbf{E}_x(i) \cdot \mathbf{h}(i) \end{aligned}, \tag{4.62d}$$

4.6 Konačni oblik BPTT algoritma za multivarijabilne sustave

Na kraju ćemo dati prikaz BPTT algoritma za općeniti problem optimalnog upravljanja s ograničenjima za multivarijabilne nelinerne dinamičke sustave, gdje ćemo sumirati rezultate dobivene u prethodnim podpoglavljima. Općenita formulacija problema je

$$\begin{aligned}
&\dot{\mathbf{x}}(t) = \phi\big(\mathbf{x}(t),\, \mathbf{u}(t)\big), \qquad \mathbf{x}(t_0) = \mathbf{x}_0, \qquad \mathbf{x}(t_f) = \mathbf{x}_f, \\
&\mathbf{g}\big(\mathbf{x}(t),\, \mathbf{u}(t)\big) \geq 0, \\
&\mathbf{h}\big(\mathbf{x}(t),\, \mathbf{u}(t)\big) = 0, \\
&J = \min_{\mathbf{u}(t)} \int_{t_0}^{t_f} F\big(\mathbf{x}(t),\, \mathbf{u}(t)\big) dt\,.
\end{aligned} \tag{4.63}$$

Nakon vremenske diskretizacije imamo

$$\begin{aligned}
&\mathbf{x}_{i+1} = \mathbf{f}(\mathbf{x}_i, \mathbf{u}_i), \qquad \mathbf{x}_{} = \mathbf{x}_{}, \qquad \mathbf{x}_N = \mathbf{x}_f, \\
&g_j(\mathbf{x}_i,\, \mathbf{u}_i) \geq 0, \qquad j = 1,2,\ldots,p, \\
&h_k(\mathbf{x}_i,\, \mathbf{u}_i) = 0, \qquad k = 1,2,\ldots,r, \\
&J(\mathbf{u}_0, \mathbf{u}_1, \ldots, \mathbf{u}_{N-1}) = \min_{\mathbf{u}} T \sum_{i=0}^{N-1} F(\mathbf{x}_i, \mathbf{u}_i)
\end{aligned} \tag{4.64}$$

za $i = 0,1,\ldots,N-1$, gdje je

$$\mathbf{f}(\mathbf{x}_i, \mathbf{u}_i) = \mathbf{x}_i + T\phi(\mathbf{x}_i, \mathbf{u}_i), \qquad \mathbf{x}_i = \begin{bmatrix} x_i^1 & x_i^2 & \cdots & x_i^n \end{bmatrix}^T, \qquad \mathbf{u}_i = \begin{bmatrix} u_i^1 & u_i^2 & \cdots & u_i^m \end{bmatrix}^T.$$

Problem se sastoji u pronalaženju upravljačkih varijabli $(\mathbf{u}_0, \mathbf{u}_1, \ldots, \mathbf{u}_{N-1})$, koje minimiziraju navedeni kriterij optimalnosti. Algoritam možemo prikazati u slijedećem obliku

1. ***Inicijalizacija gradijentnog algoritma***. *Za k=0 pridjelimo upravljačkim varijablama* $(\mathbf{u}_0^{(0)},\ \mathbf{u}_1^{(0)},\ \ldots,\ \mathbf{u}_{N-1}^{(0)})$ *proizvoljne vrijednosti (mogu biti i izvan dozvoljenog područja za vrijeme računarske simulacije).*

2. ***Računanje varijabli stanja*** $(\mathbf{x}_0^{(k)},\ \mathbf{x}_1^{(k)},\ \ldots,\ \mathbf{x}_N^{(k)})$ *za* $k = 0,\ 1,\ \ldots,\ M$ *na osnovu*

$$\mathbf{x}_{i+1}^{(k)} = \mathbf{f}(\mathbf{x}_i^{(k)}, \mathbf{u}_i^{(k)}), \quad \mathbf{x}_0^{(k)} = \mathbf{x}_0, \qquad i = 0,1,\ldots,N-1;$$

3. ***Računanje gradijenta***

$$\frac{\partial J(\mathbf{u}_0^{(k)},\ \mathbf{u}_1^{(k)},\ \ldots,\ \mathbf{u}_{N-1}^{(k)})}{\partial \mathbf{u}_j^{(k)}}$$

za $k = 0,\ 1,\ \ldots,\ M$ *na osnovi BPTT algoritma.*

3.1 ***Inicijalizacija BPTT algoritma*** za $j = N-1$.

$$\mathbf{J}_u(j) = T\left(\mathbf{F}_u(j) + 2K_V \mathbf{V}_u(j)\cdot\mathbf{g}(j) + 2K_E\mathbf{E}_u(j)\cdot\mathbf{h}(j)\right) + \mathbf{U}(N-1)\cdot\mathbf{G}_x(N),$$
$$\mathbf{P}(N-1) = \mathbf{0},$$
$$\mathbf{D}(N-1) = \mathbf{I},$$

3.2 ***Iteriranje*** za $j = N-2,\ N-3,\ \ldots, 0$.

$$\mathbf{F}_u'(j+1) = \mathbf{F}_u(j+1) + 2K_V\mathbf{V}_u(j+1)\cdot\mathbf{g}(j+1) + 2K_E\mathbf{E}_u(j+1)\cdot\mathbf{h}(j+1),$$
$$\mathbf{F}_x'(j+1) = \mathbf{F}_x(j+1) + 2K_V\mathbf{V}_x(j+1)\cdot\mathbf{g}(j+1) + 2K_E\mathbf{E}_x(j+1)\cdot\mathbf{h}(j+1),$$
$$\mathbf{P}(j) = \mathbf{F}_x'(j+1) + \mathbf{X}(j+1)\cdot\mathbf{P}(j+1),$$
$$\mathbf{D}(j) = \mathbf{D}(j+1)\mathbf{X}^T(j+1),$$
$$\mathbf{J}_u(j) = T\left(\mathbf{F}_u'(j) + \mathbf{U}(j)\cdot\mathbf{P}(j)\right) + \mathbf{U}(j)\cdot\mathbf{D}^T(j)\cdot\mathbf{G}_x(N)$$

4. ***Računanje nove iteracije upravljačkih varijabli*** $(\mathbf{u}_0^{(k+1)},\ \mathbf{u}_1^{(k+1)},\ \ldots,\ \mathbf{u}_{N-1}^{(k+1)})$ *na osnovu gradijentnog algoritma*

$$\mathbf{u}_j^{(k+1)} = \mathbf{u}_j^{(k)} - \eta\frac{\partial J(\mathbf{u}_0^{(k)},\ \mathbf{u}_1^{(k)},\ \ldots,\ \mathbf{u}_{N-1}^{(k)})}{\partial \mathbf{u}_j^{(k)}},$$

za $k = 0,\ 1,\ \ldots,\ M$. *Pomičemo indeks za jedan* $k \leftarrow k+1$ *i vraćamo se na 2. korak.*

Primjer 4.2 Razmotrimo problem optimalnog upravljanja za sustav drugog reda

$$\dot{x}_1 = x_2\,,$$

$$\dot{x}_2 = u$$

sa zadanim početnim i rubnim uvjetima

$$x_1(0) = 1\,, \qquad x_1(2) = 0\,,$$

$$x_2(0) = 1\,, \qquad x_2(2) = 0\,,$$

te s kriterijem optimalnosti

$$J = \min_u 0.5 \int_0^2 u^2 dt\,.$$

Analitičko rješenje ovog problema je

$$x_1(t) = \frac{1}{2}t^3 - \frac{7}{4}t^2 + t + 1, \qquad x_2(t) = \frac{3}{2}t^2 - \frac{7}{2}t + 1,$$

$$u(t) = 3t - \frac{7}{2}$$

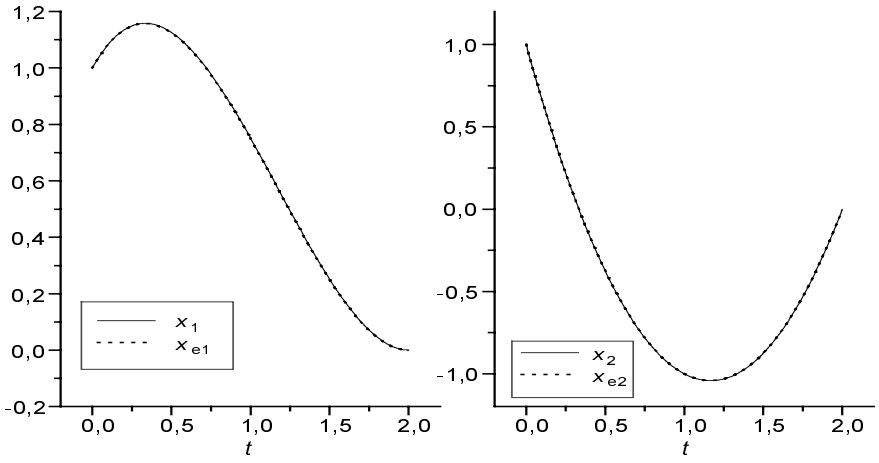
Ako sada primjenimo prethodno izvedeni algoritam na rješavanje ovog problema, imamo

$$\mathbf{X}(j) = \begin{bmatrix} 1 & 0 \\ T & 1 \end{bmatrix}, \qquad \mathbf{U}(j) = \begin{bmatrix} 0 & T \end{bmatrix}, \qquad \mathbf{G}_x(N) = 2K_B \begin{bmatrix} (x_N^1 - x_f^1) \\ (x_N^2 - x_f^2) \end{bmatrix}$$

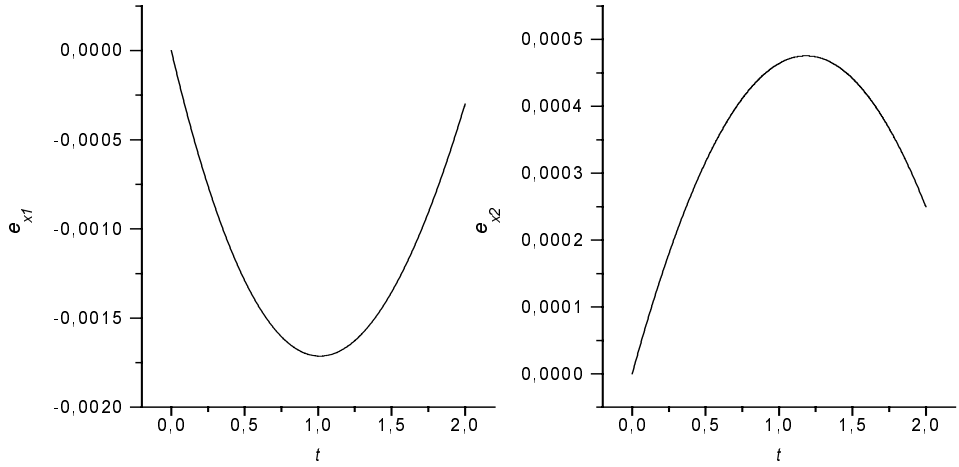
odnosno,

$$\mathbf{F}_x'(j) = \mathbf{0}\,, \qquad \mathbf{F}_u'(j) = u\,,$$

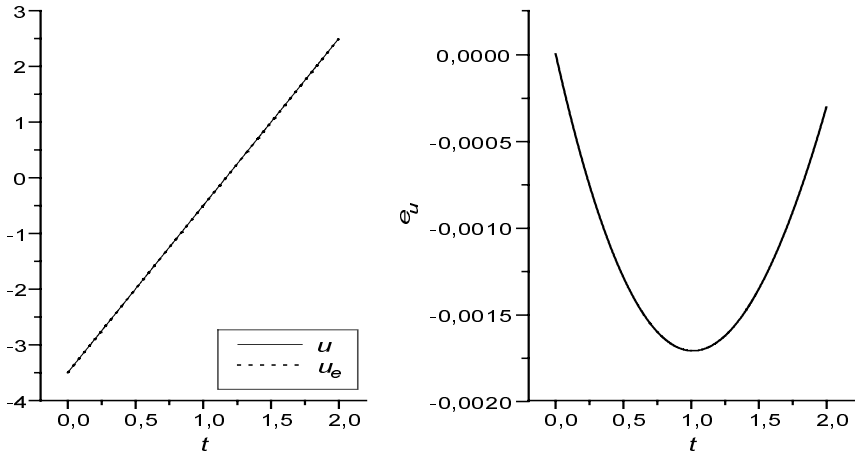
za $j = N-2,\ N-3,\ \ldots, 0$.



Slika 4.10 *Usporedba analitičkih rješenja (x_{e1}, x_{e2}) s numeričkim (x_1, x_2).*



Slika 4.11 *Odstupanja analitičkih rješenja od numeričkih* $\mathbf{e}_x = \mathbf{x}_e - \mathbf{x}$.



Slika 4.12 *Usporedba i međusobno odstupanje analitičkog rješenja (*u_e*) od numeričkog (*u*).*

Na slici 4.10 prikazana su rješenja $x_1(t)$, $x_2(t)$, dobivena BPTT algoritmom, u usporedbi s analitičkim rješenjima $x_{e1}(t)$, $x_{e2}(t)$, dok su na slici 4.11 prikazana odstupanja analitičkih rješenja od numeričkih, $e_{x1} = x_{e1}(t) - x_1(t)$, $e_{x2} = x_{e2}(t) - x_2(t)$. Slika 4.12 prikazuje upravljačku varijablu $u(t)$, dobivenu BPTT algoritmom, u usporedbi s analitičkim rješenjima $u_e(t)$ i odstupanja ta dva rješenja $e_u = u_e(t) - u(t)$.

Vrijednosti numeričkih parametara su

$$N = 1000, \qquad M = 300, \qquad \eta = 0.01, \qquad K_B = 5000.$$

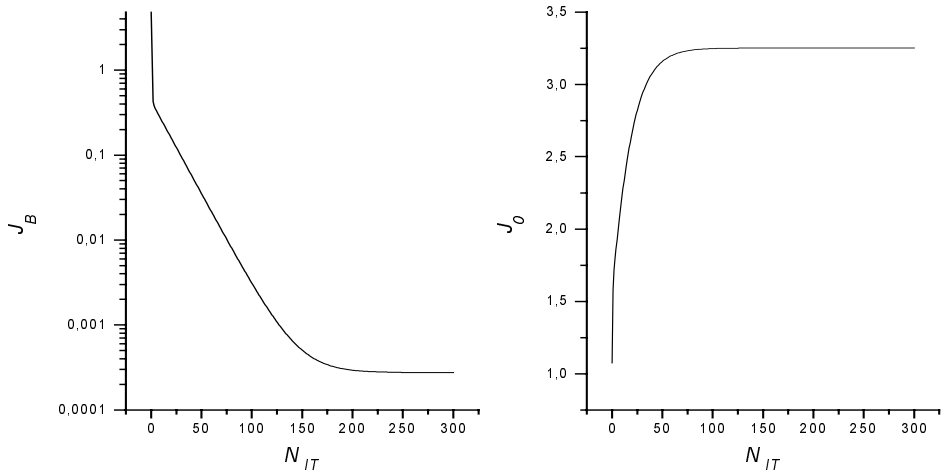
Definirajmo slijedeće funkcije

$$J_B = \frac{1}{2}\left(\left|x_N^1 - x_f^1\right| + \left|x_N^2 - x_f^2\right|\right),$$

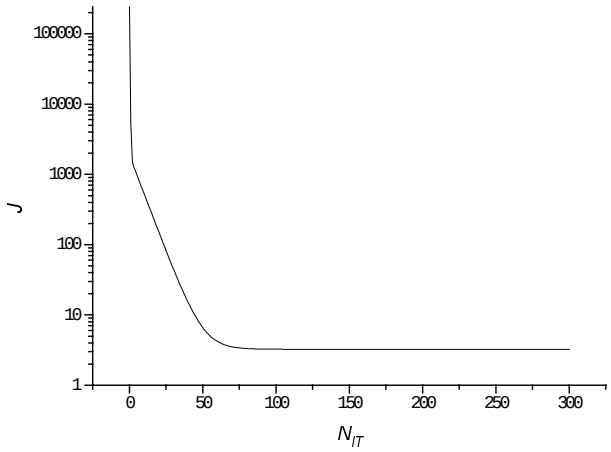
$$J_0 = 0.5T\sum_{j=0}^{N-1} u_j^2,$$

$$J = 0.5T\sum_{j=0}^{N-1} u_j^2 + K_B \sum_{k=1}^{2} (x_N^k - x_f^k)^2 ,$$

koje imaju ista značenja kao u prvom primjeru. Vrijednosti tih funkcija u ovisnosti o broju iteracija gradijentnog algoritma prikazane su na slikama 4.13 i 4.14.



Slika 4.13 *Vrijednosti* J_B, J_0, *u ovisnosti o broju iteracija* N_{IT}.



Slika 4.14 *Vrijednosti* J *u ovisnosti o broju iteracija* N_{IT}.

Ako usporedimo prvi i drugi primjer, primjećujemo da je u drugom primjeru potrebno bitno manje iteracija za dostizanje minimalne vrijednosti ukupne funkcije cilja, zbog toga što u drugom primjeru nema kaznene funkcije za ograničenja tipa nejednakosti, koja bitno usporava konvergenciju optimalnom rješenju.

S druge strane, brzina konvergencije ovisi o međusobnim omjerima koeficijenata kaznenih funkcija, kao i o samom koeficijentu konvergencije η. Npr. ako uzmemo veliki omjer K_B / K_V tada će vrijednost funkcije J_B konvergirati nuli puno brže nego J_V. Vrijednosti tih koeficijenata treba birati tako da postignemo što je moguće veću brzinu konvergencije, a da pri tom ne narušimo stabilnost algoritma.

5. Ubrzanje konvergencije BPTT algoritma

U prethodnom poglavlju izveli smo numerički algoritam za optimalno upravljanje na osnovi gradijentnog algoritma s konstantnim koeficijentom konvergencije. Dobra strana tog pristupa je jednostavnost algoritma; potrebno je samo izračunati gradijent funkcije cilja. Loša strana je relativno spora konvergencija zbog konstantne vrijednosti koeficijenta konvergencije η. S druge strane, stabilnost gradijentnog algoritma s konstantnim koeficijentom konvergencije ovisi direktno o vrijednosti tog koeficijenta. Drugim riječima, postoji gornja granica koeficijenta η iznad koje algoritam postaje nestabilan, što je s druge strane još jedno ograničenje na brzinu konvergencije algoritma.

Da bi rješili gore navedene probleme, primjenili smo metodu konjugiranog gradijenta, koju ćemo prikazati u ovom poglavlju. Metoda konjugiranog gradijenta se vrlo često primjenjuje za ubrzanje konvergencije BP algoritma kod statičkih neuronskih mreža. Tako je u 5.1 prikazana metoda najbržeg spusta koja se bazira na pronalaženju takvog koeficijenta konvergencije (na osnovu jednodimenzionalne minimizacije) za koji dobivamo najmanju vrijednost funkcije cilja u slijedećoj iteraciji gradijentnog algoritma. U 5.2 prikazana je metoda jednodimenzionalne minimizacije sa kvadratičnom konvergencijom, a u 5.3 metoda konjugiranog gradijenta koja je bazirana na metodi najbržeg spusta.

5.1 Metoda najbržeg spusta

Za minimizaciju neke funkcije $f(\mathbf{x})$ gdje je $\mathbf{x} = \begin{bmatrix} x_1 & x_2 & \cdots & x_n \end{bmatrix}^T$, možemo primjeniti gradijentni algoritam s konstantnim koeficijentom konvergencije u obliku

$$\mathbf{x}_{i+1} = \mathbf{x}_i + \eta \mathbf{d}_i, \tag{5.1}$$

gdje je $\mathbf{d}_i$ vektor smjera traženja (*search direction*)

$$\mathbf{d}_i = -\nabla f(\mathbf{x}), \tag{5.2}$$

a $\nabla f(\mathbf{x})$ je gradijent funkcije cilja

$$\nabla f(\mathbf{x}) = \left[\frac{\partial f}{\partial x_1} \quad \frac{\partial f}{\partial x_2} \quad \cdots \quad \frac{\partial f}{\partial x_n} \right]. \tag{5.3}$$

Metoda najbržeg spusta pretpostavlja, za razliku od (5.1), promjenjivu vrijednost koeficijenta η

$$\mathbf{x}_{i+1} = \mathbf{x}_i + \eta_i \mathbf{d}_i, \tag{5.4}$$

gdje se η_i određuje tokom svake iteracije tako da minimizira funkciju $f(\mathbf{x}_i + \eta_i \mathbf{d}_i)$, odnosno

$$\eta_i = \min_{\eta \geq 0} f(\mathbf{x}_i + \eta \mathbf{d}_i). \tag{5.5}$$

Drugim riječima, uzimamo onu vrijednost koeficijenta η_i koja će nam za poznati smjer gradijenta u točki $\mathbf{x}_i$ dati maksimalno smanjenje vrijednosti funkcije $f(\mathbf{x})$.

Ovaj postupak ima bitno bolja konvergencijska svojstva u odnosu na gradijentni algoritam s konstantnim koeficijentom konvergencije. S druge strane, ovom metodom smo automatski rješili problem stabilnosti algoritma. Međutim, cijena koju smo zbog toga platili je jednodimenzionalna minimizacija funkcije $f(\mathbf{x}_i + \eta \mathbf{d}_i)$, u svakom koraku iteriranja, po parametru η.

5.2 Parabolična interpolacija

Minimizaciju funkcije $f(\mathbf{x}_i + \eta \mathbf{d}_i)$ po parametru η provest ćemo paraboličnom interpolacijom, metodom kvadratične konvergencije koja daje vrlo dobre rezultate za mali broj iteracija, što nam je bitno zbog relativne osjetljivosti metode najbržeg spusta u odnosu na male promjene parametra η.

Uz pretpostavku kvadratičnog ponašanja funkcije u okolini minimuma, može se redukcija intervala u kojem leži minimum provesti kvadratičnom (paraboličnom) interpolacijom. Da bi se postupak pokrenio, mora biti poznat interval (a, b) unutar kojeg se nalazi minimum i jedna točka c unutar tog intervala. Kroz točke a, b, c, provlači se parabola i odredi njen minimum kojeg ćemo označiti s d. Redukcija intervala se zatim dobiva odbacivanjem jednog od rubnih subintervala ovisno o tome da li je $d > c$ ili $d < c$ i kakve su vrijednosti funkcija u tim točkama.

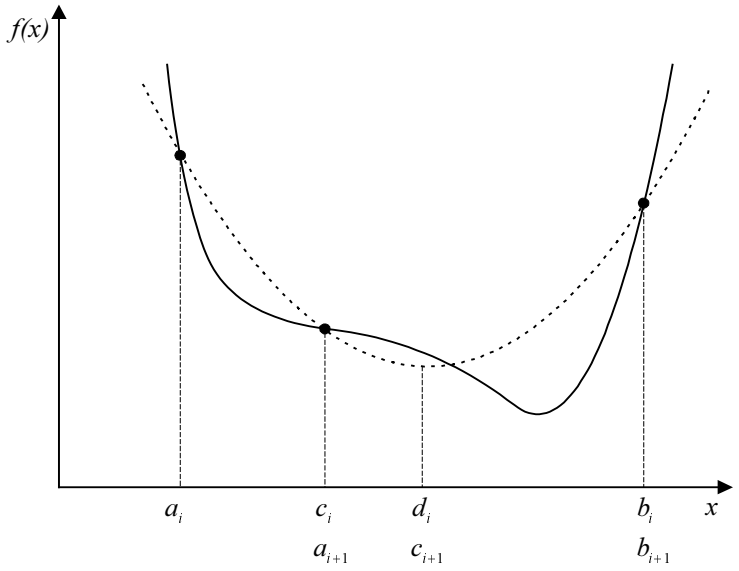
Ako su a_i i b_i, granice intervala a c_i interna točka nakon i-tog koraka redukcije, tada parabola ima minimum u točki

$$d_i = \frac{1}{2} \frac{\left(c_i^2 - b_i^2\right) f(a_i) + \left(a_i^2 - c_i^2\right) f(b_i) + \left(b_i^2 - a_i^2\right) f(c_i)}{\left(c_i - b_i\right) f(a_i) + \left(a_i - c_i\right) f(b_i) + \left(b_i - a_i\right) f(c_i)}. \tag{5.6}$$

Odbacivanje rubnih intervala i formiranje novog, manjeg, provodi se po slijedećem algoritmu [19], [20],

- ako je $\left(d_i < c_i\right) \& \left(f(d_i) < f(c_i)\right)$ tada $a_{i+1} = a_i,\ b_{i+1} = c_i,\ c_{i+1} = d_i$.
- ako je $\left(d_i < c_i\right) \& \left(f(d_i) > f(c_i)\right)$ tada $a_{i+1} = d_i,\ b_{i+1} = b_i,\ c_{i+1} = c_i$.
- ako je $\left(d_i > c_i\right) \& \left(f(d_i) < f(c_i)\right)$ tada $a_{i+1} = c_i,\ b_{i+1} = b_i,\ c_{i+1} = d_i$.
- ako je $\left(d_i > c_i\right) \& \left(f(d_i) > f(c_i)\right)$ tada $a_{i+1} = a_i,\ b_{i+1} = d_i,\ c_{i+1} = c_i$.

Suženje intervala može se se provoditi tako dugo dok se minimum ne ograniči na dovoljno mali interval.



Slika 5.1 *Parabolična interpolacija za slučaj* $\left(d_i > c_i\right) \& \left(f(d_i) < f(c_i)\right)$.

5.3 Metoda konjugiranog gradijenta

Metoda konjugiranog gradijenta (CGM), [19], predstavlja dodatno poboljšanje metode najbržeg spusta. Osnovna shema CGM za minimizaciju funkcije $f(\mathbf{x})$ je generiranje sekvence iteracija $\mathbf{x}_{i+1} = \mathbf{x}_i + \eta_i \mathbf{d}_i$, kao u (5.4), gdje je za $i = 1$, $\mathbf{d}_1 = -\nabla f(\mathbf{x}_1)$, dok je za $i > 1$

$$\mathbf{d}_{i+1} = -\nabla f(\mathbf{x}) + \alpha_i \mathbf{d}_i, \tag{5.7}$$

gdje je α_i tzv. parametar skretanja (*deflection parameter*) čije određivanje karakterizira pojedine CGM .

Zbog jednostavne primjene, vrijednost parametra α_i uzeli smo prema metodi Fletchera i Reevesa [19]

$$\alpha_i = \frac{\left\| \nabla f(\mathbf{x}_{i+1}) \right\|^2}{\left\| \nabla f(\mathbf{x}_i) \right\|^2}. \tag{5.8}$$

Drugim riječima, CGM koristi informaciju o smjeru traženja iz prethodne iteracije, te informaciju o promjeni brzine konvergencije, sadržanu u parametru α_i.

Može se pokazati [19], da za kvadratičnu funkciju $f(\mathbf{x})$, ovaj algoritam daje optimalno rješenje u najviše *n* koraka. Zbog toga, za nekvadratične funkcije vrijednost smjera traženja trebamo resetirati svakih *n* koraka (ili manje) na početnu vrijednost

$$\mathbf{d}_{n+1} = -\nabla f(\mathbf{x}_{n+1}). \tag{5.9}$$

Sada možemo prikazati kompletan algoritam.

Inicijalizacijski korak: Izabere se skalar $\varepsilon > 0$, koji određuje točnost rješenja, i početnu točku $\mathbf{y}_1$. Stavimo $\mathbf{x}_1 = \mathbf{y}_1$, $\mathbf{d}_1 = -\nabla f(\mathbf{x}_1)$, $k = j = 1$ i idemo na glavni korak.

Glavni korak:

1. Ako je $\left\| \nabla f(\mathbf{x}_j) \right\| < \varepsilon$, kraj izvršavanja algoritma. Inače, odredimo

$$\eta_j = \min_{\eta \geq 0} f(\mathbf{x}_j + \eta \mathbf{d}_j)$$

i izračunamo

$$\mathbf{x}_{j+1} = \mathbf{x}_j + \eta_j \mathbf{d}_j.$$

Ako je $j < n$, idemo na korak 2; inače, idemo na korak 3.

2. Stavimo

$$\mathbf{d}_{j+1} = -\nabla f(\mathbf{x}_{j+1}) + \alpha_j \mathbf{d}_j,$$

gdje je

$$\alpha_i = \frac{\left\| \nabla f(\mathbf{x}_{i+1}) \right\|^2}{\left\| \nabla f(\mathbf{x}_i) \right\|^2}.$$

Zamjenimo j s $j+1$ i idemo na korak 1.

3. Stavimo $\mathbf{x}_1 = \mathbf{y}_{k+1} = \mathbf{x}_{n+1}$, $\mathbf{d}_1 = -\nabla f(\mathbf{x}_1)$. Stavimo $j = 1$, zamjenimo k s $k+1$ i idemo na korak 1.

Primjer 5.1 Razmotrimo problem optimalnog upravljanja za sustav drugog reda

$$\dot{x}_1 = x_2,$$

$$\dot{x}_2 = u,$$

sa ograničenjem

$$|u| \leq 1.$$

Treba odrediti funkciju upravljanja $u(t)$, pomoću koje prebacujemo sustav iz početnog stanja

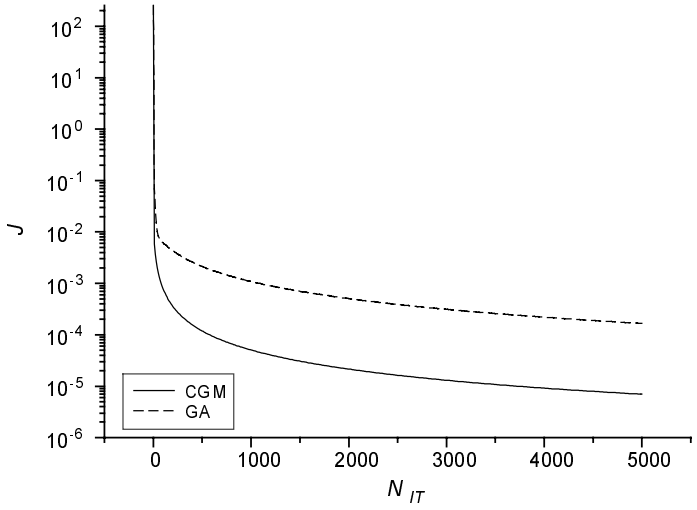
$$x(0) = 1, \qquad x_2(0) = 1,$$

u konačno

$$x_1(t_f) = 0, \qquad x_2(t_f) = 0,$$

unutar vremena $t_f = 3.45$ (koje je minimalno vrijeme, kao što ćemo pokazati u slijedećem poglavlju).

Ovaj problem ćemo detaljnije razmatrati u idućem poglavlju, dok ćemo ovdje samo usporediti brzine konvergencije gradijentne metode s konstantnim koeficijentom konvergencije i metode konjugiranog gradijenta, što je prikazano na slici 5.1.



Slika 5.1 *Usporedba konvergencije metode konjugiranog gradijenta i gradijentne metode s konstantnim koeficijentom konvergencije.*

6. Vremenski optimalno upravljanje nelinearnim sustavima

U četvrtom poglavlju razmatrali smo problem optimalnog upravljanja unutar zadanog vremenskog intervala. Ovdje ćemo razmatrati problem vremenski optimalnog upravljanja (*Time-Optimal Control*, TOC), odnosno, upravljanja u slučaju kada vremenski interval treba minimizirati.

Postoje dva pristupa numeričkom rješavanju tog problema. Kod vremenske diskretizacije ukupni vremenski interval jednak je produktu broja podintervala N s periodom diskretizacije T, tako da varijabilnost vremenskog intervala možemo izraziti jednom od tih veličina uzimajući drugu za konstantu. Dosadašnja primjena BPTT algoritma na vremenski optimalno upravljanje [22], problem tretira uzimajući konstantnim period diskretizacije T i varirajući broj podintervala N. U tom pristupu je problematična cjelobrojnost varijable N kod primjene gradijentne metode. Uzimanjem perioda diskretizacije kao kontinuirane varijable, izbjegavamo gornji problem, međutim, u tom slučaju javlja se problem nestabilnosti koji se može rješiti jedino uzimanjem jako male vrijednosti koeficijenta konvergencije u gradijentnom algoritmu za varijablu T, čime drastično usporavamo konvergenciju algoritma.

U ovom poglavlju prikazat ćemo numeričku metodu rješavanja vremenski optimalnog upravljanja, koja relativno efikasno izbjegava gore navedene probleme. Metoda je bazirana na periodu diskretizacije T kao varijabli po kojoj se minimizira, i koristi svojstva kaznenih funkcija za rubne uvjete i ograničenja.

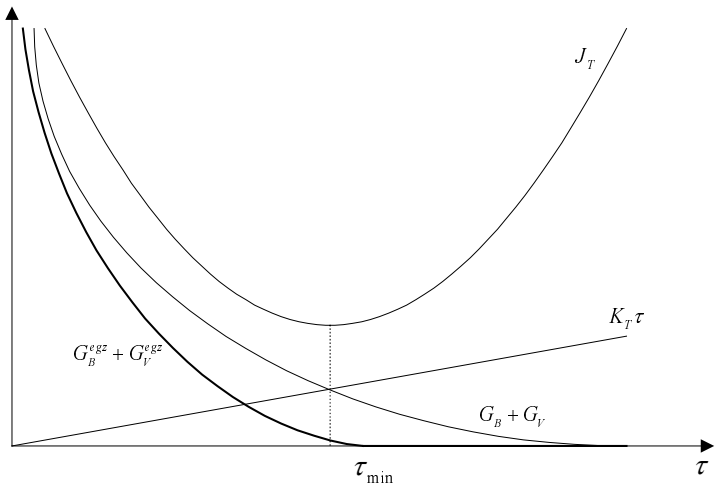
6.1 Direktna primjena BPTT algoritma na TOC

Ovdje ćemo prikazati proširenje BPTT algoritma za vremenski optimalno upravljanje (problem minimalnog vremena) za slučaj varijabilnog perioda diskretizacije, $\tau \equiv T$. Osnovna ideja je u tome da se zadrži algoritam za izračunavanje upravljačkih varijabli za zadani period diskretizacije, dok se slijedeća iteracija perioda diskretizacije izračunava na osnovi gradijenta nove funkcije cilja koja sadrži vremenski kriterij optimalnosti (minimum vremena $t_{\min} = N\tau_{\min}$, za prijelaz iz početnog u konačno stanje) i kaznene funkcije za ograničenja i rubne uvjete

$$J_T = K_T \tau + G_B(\mathbf{x}_N) + G_V(\mathbf{u}_0, \mathbf{u}_1, ..., \mathbf{u}_{N-1}). \tag{6.1}$$

Minimalno vrijeme za prijelaz iz početnog u konačno stanje ima smisla određivati jedino ako su zadana ograničenja na upravljačke varijable. To znači da bi za neko vrijeme manje od minimalnog, $t_f < t_{\min}$, BPTT algoritam za fiksnu vrijednost vremenskog

intervala neizbježno dao vrijednosti upravljačkih varijabli koje bi kršile ograničenja, što bi dovelo do povećanja vrijednosti kaznene funkcije G_V, a time ujedno dolazi do kršenja rubnih uvjeta i povećanja vrijednosti G_B. S druge strane, za vrijeme veće od minimalnog, $t_f > t_{\min}$, upravljačke varijable zadovoljavaju ograničenja i kaznene funkcije teže nuli (za egzaktna, analitička rješenja, vrijednosti kaznenih funkcija su jednake nuli). Kada se kaznenim funkcijama doda linearna ovisnost o vremenu, ukupna funkcija cilja ima minimum koji teži egzaktnoj vrijednosti $t_{\min}$, s povećavanjem koeficijenata kaznenih funkcija. Slika 6.1 daje grafički prikaz naprijed navedenog



Slika 6.1 *Prikaz ovisnosti kaznenih funkcija i J_T o periodu diskretizacije.*

Ako u diskretiziranoj formulaciji problema optimalnog upravljanja, sa specificiranim vremenom trajanja procesa, zamjenimo konstantnu vrijednost perioda diskretizacije T s varijablom $\tau \equiv T$, koja će se određivati na osnovu gradijentnog algoritma, imamo za *k*-ti korak iteracije gradijentnog algoritma

$$\mathbf{x}_{i+1}^{(k)} = \mathbf{x}_i^{(k)} + \tau^{(k)}\, \phi(\mathbf{x}_i^{(k)}, \mathbf{u}_i^{(k)}), \qquad \mathbf{x}_0^{(k)} = \mathbf{x}_0\,, \qquad \mathbf{x}_N^{(k)} = \mathbf{x}_f\,, \tag{6.2}$$

$$g_j(\mathbf{x}_i^{(k)},\ \mathbf{u}_i^{(k)}) \geq 0\,, \qquad j = 1,2,\ldots,p\,, \tag{6.3}$$

$$J(\mathbf{u}_0^{(k)}, \mathbf{u}_1^{(k)}, \ldots, \mathbf{u}_{N-1}^{(k)}) = \min_{\mathbf{u}} \tau^{(k)} \sum_{i=0}^{N-1} F'(\mathbf{x}_0^{(k)}, \mathbf{u}_0^{(k)}) + G_B(\mathbf{x}_N^{(k)}) \tag{6.4}$$

za $i = 0,1,\ldots,N-1$, gdje je $\mathbf{x}_i = \begin{bmatrix} x_i^1 & x_i^2 & \cdots & x_i^n \end{bmatrix}^T$, $\mathbf{u}_i = \begin{bmatrix} u_i^1 & u_i^2 & \cdots & u_i^m \end{bmatrix}^T$.

Gornji problem za fiksni τ rješen je u 4. poglavlju gradijentnom metodom

$$\mathbf{u}_j^{(k+1)} = \mathbf{u}_j^{(k)} - \eta \frac{\partial J(\mathbf{u}_0^{(k)},\ \mathbf{u}_1^{(k)},\ \ldots,\ \mathbf{u}_{N-1}^{(k)})}{\partial \mathbf{u}_j^{(k)}}\,. \tag{6.5}$$

Da bi izračunali slijedeću iteraciju $\tau^{(k+1)}$ primjenom gradijentnog algoritma, moramo funkciju cilja J_T implicitno izraziti u ovisnosti o $\tau^{(k)}$

$$J_T(\tau^{(k)}) = K_T \tau^{(k)} + G_V(\mathbf{u}_0^{(k+1)}, \mathbf{u}_1^{(k+1)}, ..., \mathbf{u}_{N-1}^{(k+1)}) + G_B(\mathbf{x}_N^{(k)}), \tag{6.6}$$

tako da imamo

$$\tau^{(k+1)} = \tau^{(k)} - \mu \frac{\partial J_T(\tau^{(k)})}{\partial \tau^{(k)}}. \tag{6.7}$$

Sada se problem svodi na računanje gradijenta funkcije cilja (6.6)

$$\frac{\partial J_T(\tau^{(k)})}{\partial \tau^{(k)}} = K_T + \frac{\partial G_V}{\partial \tau^{(k)}} + \frac{\partial G_B}{\partial \tau^{(k)}}. \tag{6.8}$$

Počet ćemo s gradijentom kaznene funkcije za rubne uvjete

$$\frac{\partial G_B}{\partial \tau^{(k)}} = \mathbf{G}_x^T(N) \cdot \mathbf{X}_\tau(N), \tag{6.9}$$

gdje je

$$\mathbf{X}_\tau(N) = \left[\frac{\partial x_1^{(k)}(N)}{\partial \tau^{(k)}} \quad \frac{\partial x_2^{(k)}(N)}{\partial \tau^{(k)}} \quad \cdots \quad \frac{\partial x_n^{(k)}(N)}{\partial \tau^{(k)}} \right]^T,$$

s tim da je $x_j^{(k)}(N)$ j-ta komponenta vektora $\mathbf{x}_N^{(k)}$. Rekurzivnu relaciju za gornji vektor dobivamo derivirajući izraz (5.2)

$$\mathbf{X}_\tau(j) = \mathbf{X}_\tau(j-1) + \phi(j-1) + \tau^{(k)} \mathbf{X}^T(j-1) \cdot \mathbf{X}_\tau(j-1), \tag{6.10}$$

za $j = 1, ..., N$, s početnim uvjetom

$$\mathbf{X}_\tau(0) = \mathbf{0}.$$

Sad nam preostaje računanje gradijenta kaznene funkcije ograničenja G_V

$$\frac{\partial G_V}{\partial \tau^{(k)}} = \sum_{j=0}^{N-1} \mathbf{G}_{Vu}^T(j) \cdot \mathbf{U}_\tau(j), \tag{6.11}$$

gdje je

$$\mathbf{G}_{Vu}(j) = \left[\frac{\partial G_V}{\partial u_1^{(k+1)}(j)} \quad \frac{\partial G_V}{\partial u_2^{(k+1)}(j)} \quad \cdots \quad \frac{\partial G_V}{\partial u_m^{(k+1)}(j)} \right]^T,$$

$$\mathbf{U}_\tau(j) = \left[\frac{\partial u_1^{(k+1)}(j)}{\partial \tau^{(k)}} \quad \frac{\partial u_2^{(k+1)}(j)}{\partial \tau^{(k)}} \quad \cdots \quad \frac{\partial u_m^{(k+1)}(j)}{\partial \tau^{(k)}} \right]^T,$$

s tim da je $u_i^{(k+1)}(j)$ i-ta komponenta vektora $\mathbf{u}_j^{(k+1)}$. $\mathbf{x}_N^{(k)}$. Rekurzivnu relaciju za vektor $\mathbf{U}_\tau(j)$ dobivamo derivirajući izraz (5.5), što se svodi, kao što vidimo na derivaciju gradijenta funkcije cilja (5.4) po upravljačkim varijablama. To smo izračunali u prošlom poglavlju i dobili slijedeće rekurzivne relacije koje, prilagođene za ovaj problem poprimaju oblik

$$\mathbf{J}_u(j) = \tau^{(k)}\left(\mathbf{F}'_u(j) + \mathbf{S}(j)\right) + \mathbf{X}_u^T(j)\cdot\mathbf{G}_x(N),$$
$$\mathbf{X}_u(j) = \mathbf{U}(j)\cdot\mathbf{D}\ \ (j),$$
$$\mathbf{D}^T(j) = \mathbf{X}(j+1)\cdot\mathbf{D}^T(j+1),$$
$$\mathbf{S}(j) = \mathbf{U}(j)\cdot\mathbf{P}(j),$$
$$\mathbf{P}(j) = \mathbf{F}'_x(j+1) + \mathbf{X}(j+1)\cdot\mathbf{P}(j+1),$$

sa rubnim uvjetima

$$\mathbf{J}_u(N-1) = \tau^{(k)}\mathbf{F}'_u(N-1) + \mathbf{U}(N-1)\cdot\mathbf{G}_x(N)$$
$$\mathbf{P}(N-1) = \mathbf{0},$$
$$\mathbf{D}(N-1) = \mathbf{I},$$

za $j = N-2,\ N-3,\ \ldots, 0$.

Imamo dakle

$$\mathbf{U}_\tau(j) = -\eta\frac{\partial\mathbf{J}_u(j)}{\partial\tau^{(k)}}, \tag{6.12}$$

odnosno

$$\frac{\partial\mathbf{J}_u(j)}{\partial\tau^{(k)}} = \mathbf{F}'_u(j) + \mathbf{S}(j) + \tau^{(k)}\left(\frac{\partial\mathbf{F}'_u(j)}{\partial\tau^{(k)}} + \frac{\partial\mathbf{S}(j)}{\partial\tau^{(k)}}\right) + \frac{\partial\mathbf{X}_u^T(j)}{\partial\tau^{(k)}}\mathbf{G}_x(N) + \mathbf{X}_u^T(j)\frac{\partial\mathbf{G}_x(N)}{\partial\tau^{(k)}}.$$

Prvo ćemo izračunati

$$\frac{\partial\mathbf{S}(j)}{\partial\tau^{(k)}} = \frac{\partial\mathbf{U}(j)}{\partial\tau^{(k)}}\mathbf{P}(j) + \mathbf{U}(j)\frac{\partial\mathbf{P}(j)}{\partial\tau^{(k)}}, \tag{6.13}$$

odnosno

$$\frac{\partial\mathbf{P}(j)}{\partial\tau^{(k)}} = \frac{\partial\mathbf{F}'_x(j+1)}{\partial\tau^{(k)}} + \frac{\partial\mathbf{X}(j+1)}{\partial\tau^{(k)}}\mathbf{P}(j+1) + \mathbf{X}(j+1)\frac{\partial\mathbf{P}(j+1)}{\partial\tau^{(k)}}. \tag{6.14}$$

Pošto su vrijednosti matrica $\mathbf{U}(j)$, $\mathbf{X}(j)$, koje sadrže parcijalne derivacije funkcije $\mathbf{f}(\mathbf{x}_j^{(k)}, \mathbf{u}_j^{(k)})$ po varijablama $\mathbf{u}_j^{(k)}, \mathbf{x}_j^{(k)}$, ovisne o $\tau^{(k)}$ preko izraza (5.2), na slijedeći način

$$\mathbf{U}(j) = \tau^{(k)}\Phi_u(j), \qquad \mathbf{X}(j) = \mathbf{I} + \tau^{(k)}\Phi_x(j),$$

gdje matrice $\Phi_u(j)$, $\Phi_x(j)$, analogno matricama $\mathbf{U}(j)$, $\mathbf{X}(j)$, sadrže parcijalne derivacije funkcije $\phi(\mathbf{x}_j^{(k)}, \mathbf{u}_j^{(k)})$ po varijablama $\mathbf{u}_j^{(k)}, \mathbf{x}_j^{(k)}$, što znači da nisu ovisne eksplicitno o $\tau^{(\ )}$, možemo napisati

$$\frac{\partial \mathbf{U}(j)}{\partial \tau^{(k)}} = \Phi_u(j) + \tau^{(k)} \frac{\partial \Phi_u(j)}{\partial \tau^{(k)}}. \tag{6.15}$$

Nadalje, imamo

$$\frac{\partial \Phi_u(j)}{\partial \tau^{(k)}} = \sum_{r=1}^{n} X_\tau^r(j)\, \Phi_{ux(r)}(j), \tag{6.16}$$

gdje je

$$\Phi_{ux(r)}(j) = \begin{bmatrix} \dfrac{\partial^2 \phi^1}{\partial u_j^1 \partial x_j^r} & \dfrac{\partial^2 \phi^2}{\partial u_j^1 \partial x_j^r} & \cdots & \dfrac{\partial^2 \phi^n}{\partial u_j^1 \partial x_j^r} \\ \dfrac{\partial^2 \phi^1}{\partial u_j^2 \partial x_j^r} & \dfrac{\partial^2 \phi^2}{\partial u_j^2 \partial x_j^r} & \cdots & \dfrac{\partial^2 \phi^n}{\partial u_j^2 \partial x_j^r} \\ \vdots & \vdots & \ddots & \vdots \\ \dfrac{\partial^2 \phi^1}{\partial u_j^m \partial x_j^r} & \dfrac{\partial^2 \phi^2}{\partial u_j^m \partial x_j^r} & \cdots & \dfrac{\partial^2 \phi^n}{\partial u_j^m \partial x_j^r} \end{bmatrix}.$$

Isto tako, imamo

$$\frac{\partial \mathbf{X}(j+1)}{\partial \tau^{(k)}} = \Phi_x(j+1) + \tau^{(k)} \frac{\partial \Phi_x(j+1)}{\partial \tau^{(k)}}, \tag{6.17}$$

odnosno

$$\frac{\partial \Phi_x(j+1)}{\partial \tau^{(k)}} = \sum_{r=1}^{n} X_\tau^r(j+1)\, \Phi_{xx(r)}(j+1), \tag{6.18}$$

gdje je

$$\Phi_{xx(r)}(j) = \begin{bmatrix} \dfrac{\partial^2 \phi^1}{\partial x_j^1 \partial x_j^r} & \dfrac{\partial^2 \phi^2}{\partial x_j^1 \partial x_j^r} & \cdots & \dfrac{\partial^2 \phi^n}{\partial x_j^1 \partial x_j^r} \\ \dfrac{\partial^2 \phi^1}{\partial x_j^2 \partial x_j^r} & \dfrac{\partial^2 \phi^2}{\partial x_j^2 \partial x_j^r} & \cdots & \dfrac{\partial^2 \phi^n}{\partial x_j^2 \partial x_j^r} \\ \vdots & \vdots & \ddots & \vdots \\ \dfrac{\partial^2 \phi^1}{\partial x_j^n \partial x_j^r} & \dfrac{\partial^2 \phi^2}{\partial x_j^n \partial x_j^r} & \cdots & \dfrac{\partial^2 \phi^n}{\partial x_j^n \partial x_j^r} \end{bmatrix}.$$

Nadalje, imamo

$$\frac{\partial \mathbf{F}_u'(j)}{\partial \tau^{(k)}} = \mathbf{F}_{ux}'(j) \cdot \mathbf{X}_\tau(j), \tag{6.19}$$

$$\frac{\partial \mathbf{F}_x'(j+1)}{\partial \tau^{(k)}} = \mathbf{F}_{xx}'(j+1) \cdot \mathbf{X}_\tau(j+1), \tag{6.20}$$

gdje je

$$\mathbf{F}'_{ux}(j)=\begin{bmatrix}\frac{\partial^2 F'}{\partial u_j^1\partial x_j^1} & \frac{\partial^2 F'}{\partial u_j^1\partial x_j^2} & \cdots & \frac{\partial^2 F'}{\partial u_j^1\partial x_j^n}\\ \frac{\partial^2 F'}{\partial u_j^2\partial x_j^1} & \frac{\partial^2 F'}{\partial u_j^2\partial x_j^2} & \cdots & \frac{\partial^2 F'}{\partial u_j^2\partial x_j^n}\\ \vdots & \vdots & \ddots & \vdots\\ \frac{\partial^2 F'}{\partial u_j^m\partial x_j^1} & \frac{\partial^2 F'}{\partial u_j^m\partial x_j^2} & \cdots & \frac{\partial^2 F'}{\partial u_j^m\partial x_j^n}\end{bmatrix},$$

$$\mathbf{F}'_{xx}(j)=\begin{bmatrix}\frac{\partial^2 F'}{\partial x_j^1\partial x_j^1} & \frac{\partial^2 F'}{\partial x_j^1\partial x_j^2} & \cdots & \frac{\partial^2 F'}{\partial x_j^1\partial x_j^n}\\ \frac{\partial^2 F'}{\partial x_j^2\partial x_j^1} & \frac{\partial^2 F'}{\partial x_j^2\partial x_j^2} & \cdots & \frac{\partial^2 F'}{\partial x_j^2\partial x_j^n}\\ \vdots & \vdots & \ddots & \vdots\\ \frac{\partial^2 F'}{\partial x_j^n\partial x_j^1} & \frac{\partial^2 F'}{\partial x_j^n\partial x_j^2} & \cdots & \frac{\partial^2 F'}{\partial x_j^n\partial x_j^n}\end{bmatrix}.$$

Na kraju, ostaje

$$\frac{\partial \mathbf{G}_x(N)}{\partial \tau^{(k)}}=\mathbf{G}_{xx}(N)\cdot\mathbf{X}_\tau(N), \tag{6.21}$$

$$\frac{\partial \mathbf{X}_u^T(j)}{\partial \tau^{(k)}}=\frac{\partial \mathbf{U}(j)}{\partial \tau^{(k)}}\mathbf{D}^T(j)+\mathbf{U}(j)\frac{\partial \mathbf{D}^T(j)}{\partial \tau^{(k)}}, \tag{6.22}$$

$$\frac{\partial \mathbf{D}^T(j)}{\partial \tau^{(k)}}=\frac{\partial \mathbf{X}(j+1)}{\partial \tau^{(k)}}\mathbf{D}^T(j+1)+\mathbf{X}(j+1)\frac{\partial \mathbf{D}^T(j+1)}{\partial \tau^{(k)}}, \tag{6.23}$$

gdje je

$$\mathbf{G}_{xx}(N)=\begin{bmatrix}\frac{\partial^2 G_B}{\partial x_N^1\partial x_N^1} & \frac{\partial^2 G_B}{\partial x_N^1\partial x_N^2} & \cdots & \frac{\partial^2 G_B}{\partial x_N^1\partial x_N^n}\\ \frac{\partial^2 G_B}{\partial x_N^2\partial x_N^1} & \frac{\partial^2 G_B}{\partial x_N^2\partial x_N^2} & \cdots & \frac{\partial^2 G_B}{\partial x_N^2\partial x_N^n}\\ \vdots & \vdots & \ddots & \vdots\\ \frac{\partial^2 G_B}{\partial x_N^n\partial x_N^1} & \frac{\partial^2 G_B}{\partial x_N^n\partial x_N^2} & \cdots & \frac{\partial^2 G_B}{\partial x_N^n\partial x_N^n}\end{bmatrix},$$

što je zbog oblika kaznene funkcije (4.49) jednako

$$\mathbf{G}_{xx}(N)=2K_B\mathbf{I}.$$

Ovim smo izračunali sve što nam je potrebno za konačan oblik algoritma:

$$\tau^{(k+1)}=\tau^{(k)}-\mu\frac{\partial J_T(\tau^{(k)})}{\partial \tau^{(k)}},$$

$$\frac{\partial J_T(\tau^{(k)})}{\partial \tau^{(k)}}=K_T+\frac{\partial G_V}{\partial \tau^{(k)}}+\frac{\partial G_B}{\partial \tau^{(k)}},$$

$$\frac{\partial G_B}{\partial \tau^{(k)}} = \mathbf{G}_x^T(N) \cdot \mathbf{X}_\tau(N),$$

$$\frac{\partial G_V}{\partial \tau^{(k)}} = \sum_{j=0}^{N-1} \mathbf{G}_{Vu}^T(j) \cdot \mathbf{U}_\tau(j),$$

$$\mathbf{U}_\tau(j) = -\eta \frac{\partial \mathbf{J}_u(j)}{\partial \tau^{(k)}},$$

$$\frac{\partial \mathbf{J}_u(j)}{\partial \tau^{(k)}} = \mathbf{F}_u'(j) + \mathbf{S}(j) + \tau^{(k)} \left(\frac{\partial \mathbf{F}_u'(j)}{\partial \tau^{(k)}} + \frac{\partial \mathbf{S}(j)}{\partial \tau^{(k)}} \right) + \frac{\partial \mathbf{X}_u^T(j)}{\partial \tau^{(k)}} \mathbf{G}_x(N) + \mathbf{X}_u^T(j) \frac{\partial \mathbf{G}_x(N)}{\partial \tau^{(k)}},$$

$$\frac{\partial \mathbf{F}_u'(j)}{\partial \tau^{(k)}} = \mathbf{F}_{ux}'(j) \cdot \mathbf{X}_\tau(j),$$

$$\mathbf{S}(j) = \mathbf{U}(j) \cdot \mathbf{P}(j),$$

$$\frac{\partial \mathbf{S}(j)}{\partial \tau^{(k)}} = \frac{\partial \mathbf{U}(j)}{\partial \tau^{(k)}} \mathbf{P}(j) + \mathbf{U}(j) \frac{\partial \mathbf{P}(j)}{\partial \tau^{(k)}},$$

$$\mathbf{P}(j) = \mathbf{F}_x'(j+1) + \mathbf{X}(j+1) \cdot \mathbf{P}(j+1),$$

$$\frac{\partial \mathbf{P}(j)}{\partial \tau^{(k)}} = \mathbf{F}_{xx}'(j+1) \cdot \mathbf{X}_\tau(j+1) + \frac{\partial \mathbf{X}(j+1)}{\partial \tau^{(k)}} \mathbf{P}(j+1) + \mathbf{X}(j+1) \frac{\partial \mathbf{P}(j+1)}{\partial \tau^{(k)}}$$

$$\frac{\partial \mathbf{G}_x(N)}{\partial \tau^{(k)}} = \mathbf{G}_{xx}(N) \cdot \mathbf{X}_\tau(N),$$

$$\mathbf{X}_u^T(j) = \mathbf{U}(j) \cdot \mathbf{D}^T(j),$$

$$\frac{\partial \mathbf{X}_u^T(j)}{\partial \tau^{(k)}} = \frac{\partial \mathbf{U}(j)}{\partial \tau^{(k)}} \mathbf{D}^T(j) + \mathbf{U}(j) \frac{\partial \mathbf{D}^T(j)}{\partial \tau^{(k)}},$$

$$\mathbf{D}^T(j) = \mathbf{X}(j+1) \cdot \mathbf{D}^T(j+1),$$

$$\frac{\partial \mathbf{D}^T(j)}{\partial \tau^{(k)}} = \frac{\partial \mathbf{X}(j+1)}{\partial \tau^{(k)}} \mathbf{D}^T(j+1) + \mathbf{X}(j+1) \frac{\partial \mathbf{D}^T(j+1)}{\partial \tau^{(k)}},$$

$$\frac{\partial \mathbf{X}(j+1)}{\partial \tau^{(k)}} = \Phi_x(j+1) + \tau^{(k)} \sum_{r=1}^{n} X_\tau^r(j+1) \Phi_{xx(r)}(j+1),$$

$$\frac{\partial \mathbf{U}(j)}{\partial \tau^{(k)}} = \Phi_u(j) + \tau^{(k)} \sum_{r=1}^{n} X_\tau^r(j) \Phi_{ux(r)}(j),$$

za $j = N-2,\ N-3,\ \ldots, 0$, te

$$\mathbf{X}_\tau(j) = \mathbf{X}_\tau(j-1) + \phi(j-1) + \tau^{(k)} \mathbf{X}^T(j-1) \cdot \mathbf{X}_\tau(j-1),$$

za $j = 1, 2, \ldots, N$ i $k = 0, 1, \ldots, M$.

Iz svega ovoga jasno vidimo da je dobiveni algoritam puno složeniji i glomazniji u odnosu na BPTT algoritam sa konstantnim korakom diskretizacije koji smo izveli u prošlom poglavlju, te stoga prilično nepraktičan za primjenu.

S druge strane, aproksimacijom derivacije,

$$\frac{\partial J_T(\tau^{(k)})}{\partial \tau^{(k)}} \approx \frac{J_T(\tau^{(k)}) - J_T(\tau^{(k-1)})}{\tau^{(k)} - \tau^{(k-1)}}, \tag{6.24}$$

dobivamo nestabilna rješenja zbog greške koja se javlja kod oduzimanja bliskih brojeva, što je posljedica računanja s konačnim brojem decimala.

Zbog činjenice da je egzaktno računanje derivacije prekomplicirano za praktičnu primjenu, a numeričko nestabilno, slijedi da problemu vremenski optimalnog upravljanja moramo prići na drugi način, što će biti prikazano u slijedećem podpoglavlju.

6.2 Primjena kaznenih funkcija za TOC

Kao što smo već spomenuli, za vremenski interval koji je veći od $t_{\min}$ biti će zadovoljena sva ograničenja (kaznene funkcije za ograničenja i rubne uvjete jednake su nuli), dok za slučaj $t < t_{\min}$ ograničenja neće biti zadovoljena (kaznene funkcije biti će veće od nule), što je prikazano na slici 6.1. Kod numeričkog izračunavanja to znači da će za $t > t_{\min}$ vrijednost sume kaznenih funkcija konvergirati prema nuli, dok će za $t < t_{\min}$, ta vrijednost konvergirati nekom pozitivnom broju. Što je vrijednost sume kaznenih funkcija za zadani $t > t_{\min}$ manja, to je rješenje točnije (bliže optimalnom).

Međutim, slika 6.1 odgovara vrijednostima kaznenih funkcija nakon određenog broja iteracija za svaki τ, dok se za TOC zahtijeva podešavanje $\tau^{(k)}$ istovremeno s izvršavanjem BPTT algoritma. Drugim riječima, ako na početku izvršavanja algoritma krenemo s nekom vrijednošću $\tau > \tau_{\min}$, vrijednosti kaznenih funkcija će biti velike, iako će s povećanjem broja iteracija težiti nuli. To je glavni izvor problema s konvergencijom i stabilnošću iterativnih metoda tipa $\tau^{(k+1)} = f\left(\tau^{(k)}\right)$, gdje je $f(\tau)$ neprekidna funkcija. Ti problemi se mogu izbjeći nekom vrstom "ireverzibilne" iterativne metode koju ćemo sada izložiti.

Prvo, definiramo funkciju J_G, koja je jednaka sumi kaznenih funkcija za ograničenja i rubne uvjete

$$J_G\left(\tau^{(k)}\right) = G_V(\mathbf{u}_0, \mathbf{u}_1, \ldots, \mathbf{u}_{N-1}) + G_B(\mathbf{x}_N), \tag{6.25}$$

gdje je G_V jednak izrazu (4.55). Minimalna vrijednost funkcije J jednaka je nuli u slučaju zadovoljenja svih ograničenja. Da bi izbjegli eksplicitnu ovisnost funkcije J o τ, za koeficijent kaznene funkcije K_V^*, uzimamo konstantnu vrijednost, neovisnu o τ.

Zatim, definiramo minimalnu vrijednost funkcije J koja je ujedno i mjera točnosti rješenja i označimo je sa $J_{G\min}$. Što je vrijednost $J_{G\min}$ manja, to je rješenje točnije, ali je također potreban veći broj iteracija da se dostigne ta vrijednost.

Sada možemo postaviti slijedeći općeniti algoritam

1. Inicijalizacija $\tau^{(0)}$, takvog da je $\tau^{(0)} > \tau_{\min}$,
2. *ako je* $J_G\left(\tau^{(k)}\right) < J_{G\min}$ *tada* $\tau^{(k+1)} < \tau^{(k)}$,
3. *ako je* $J_G\left(\tau^{(k)}\right) \geq J_{G\min}$ *tada* $\tau^{(k+1)} = \tau^{(k)}$,

za $k = 0,1,\ldots, M$. Pošto ne znamo vrijednost $\tau_{\min}$, početnu vrijednost τ odabiremo na osnovi procjene. U slučaju da stavimo $\tau^{(0)} < \tau_{\min}$, tada J tokom iteriranja neće dostići vrijednost $J_{G\min}$, što je znak da smo izabrali pogrešnu početnu vrijednost.

Drugim riječima, slijedeća iteracija perioda diskretizacije $\tau^{(k)}$, uslijedit će samo kada vrijednost funkcije J dostigne zadanu vrijednost $J_{G\min}$, a to znači da je $\tau^{(k)} > \tau_{\min}$. Sa smanjivanjem $\tau^{(k)}$, J će sve sporije konvergirati prema $J_{G\min}$, dok se ne dođe do vrijednosti $\tau^{(k)}$ za koju J neće moći dostići $J_{G\min}$ ili će mu jako sporo konvergirati, što znači da je $\tau^{(k)} < \tau_{\min}$. U tom slučaju, kao aproksimaciju za $\tau_{\min}$ uzimamo $\tau_{\min} \approx \tau^{(k-1)}$.

Ovakva struktura algoritma nam garantira stabilnost i konvergenciju prema $\tau_{\min}$, zbog toga što ne mjenja vrijednost $\tau^{(k)}$ sve dok vrijednost funkcije J ne padne ispod zadane, dovoljno male, vrijednosti $J_{G\min}$.

Drugi i treći korak navedenog algoritma možemo objediniti slijedećim izrazom

$$\tau^{(k+1)} = \tau^{(k)} - \varphi\left(\tau^{(k)}\right) H^{-}\left(J_G\left(\tau^{(k)}\right) - J_{G\min}\right). \tag{6.26}$$

Brzina konvergencije ovisit će o izboru funkcije $\varphi\left(\tau^{(k)}\right)$. Ako stavimo konstantu, $\varphi\left(\tau^{(k)}\right) = \Delta\tau$, dobivamo

$$\tau^{(k+1)} = \tau^{(k)} - \Delta\tau\, H^{-}\left(J_G\left(\tau^{(k)}\right) - J_{G\min}\right), \tag{6.27}$$

odnosno, svaki put kad je zadovoljeno ograničenje $J_G\left(\tau^{(k)}\right) < J_{G\min}$, τ se smanji za konstantnu vrijednost $\Delta\tau$.

Druga mogućnost je da uzmemo

$$\tau^{(k+1)} = \tau^{(k)} - \mu\left(J_{G\min} - J_G\left(\tau^{(k)}\right)\right) H^{-}\left(J_G\left(\tau^{(k)}\right) - J_{G\min}\right), \tag{6.28}$$

odnosno, promjena varijable τ proporcionalna je 'udaljenosti' funkcije J_G od zadane vrijednosti $J_{G\min}$. Gornjim izrazom omogućujemo kontinuiraniju promjenu varijable τ.

Ova dva pristupa usporediti ćemo međusobno na slijedećem primjeru.

Primjer 5.1 Razmotrimo problem optimalnog upravljanja za sustav drugog reda

$$\dot{x}_1 = x_2,$$

$$\dot{x}_2 = u,$$

sa ograničenjem

$$|u| \le 1.$$

Treba naći najkraće vrijeme, t_f, za prijelaz sustava iz početnog stanja

$$x_1(0) = 1, \qquad x_2(0) = 1,$$

u konačno

$$x_1(t_f) = 0, \qquad x_2(t_f) = 0.$$

Analitičko rješenje ovog problema je

$$x_1(t) = \begin{cases} -0.5t^2 + t + 1 \;\; ; \;\; 0 \le t \le t_s \\ 0.5t^2 + A_1 t + A_2 \;\; ; \;\; t_s \le t \le t_f \end{cases}, \qquad x_2(t) = \begin{cases} -t + 1 \;\; ; \;\; 0 \le t \le t_s \\ t + A_1 \;\; ; \;\; t_s \le t \le t_f \end{cases},$$

$$u(t) = \begin{cases} -1 \;\; ; \;\; 0 \le t \le t_s \\ 1 \;\; ; \;\; t_s \le t \le t_f \end{cases},$$

gdje je

$$t_s = 2.225, \qquad t_f = 3.45, \text{ odnosno } \tau_{\min} = 0.00345 \;\; (N = 1000),$$

$$A_1 = -2t_s + 1, \qquad A_2 = -t_s^2 + (1 - A_1)t_s + 1.$$

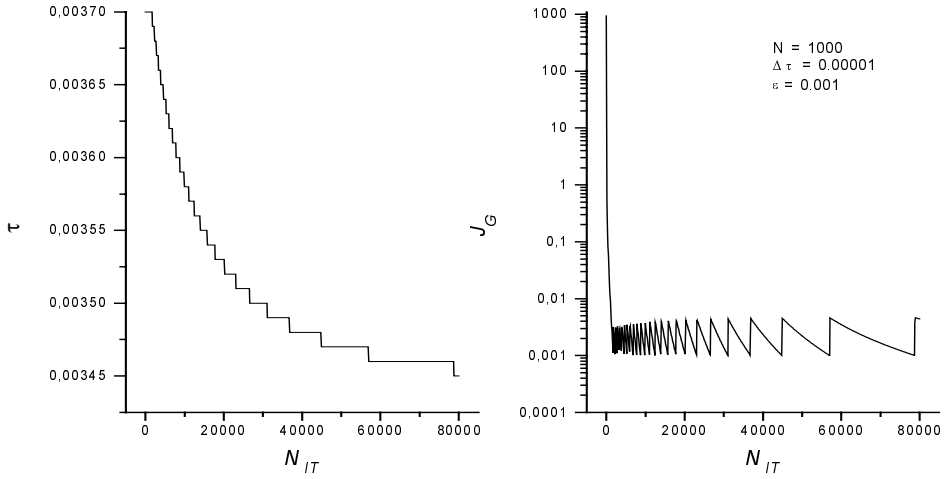
Na slici 6.2 prikazan je iterativni postupak za izračunavanje $\tau_{\min}$ prema izrazu (6.27) za $\varepsilon \equiv J_{G\min} = 0.001$ i $\Delta\tau = 0.00001$. Vidimo da se $\tau^{(k)}$ mijenja skokovito za $\Delta\tau$ svaki put kad se J_G spusti ispod $\varepsilon \equiv J_{G\min}$. Kada dođe do promjene τ , vrijednost funkcije J_G ponovo skoči iznad ε i proces se ponavlja. Što je manji $\Delta\tau$, to su manji skokovi za τ i J_G (slika 6.3). Također što je manja vrijednost ε, to su točniji rezultati, ali i sporija konvergencija.

Na slikama 6.4 i 6.5 prikazan je postupak izračunavanja $\tau_{\min}$ prema izrazu (6.28) za $\varepsilon \equiv J_{G\min} = 0.01$, te $\mu = 0.1$ i $\mu = 1.0$ respektivno. Na slici 6.4 za $N_{IT} = 80000$, dobili smo $t_{\min} = 3.45056$, a na slici 6.5 imamo $t_{\min} = 3.44981$. Vidimo da za manju vrijednost μ dobivamo kontinuiranu ovisnost τ o N_{IT}, ali i nešto sporiju konvergenciju.

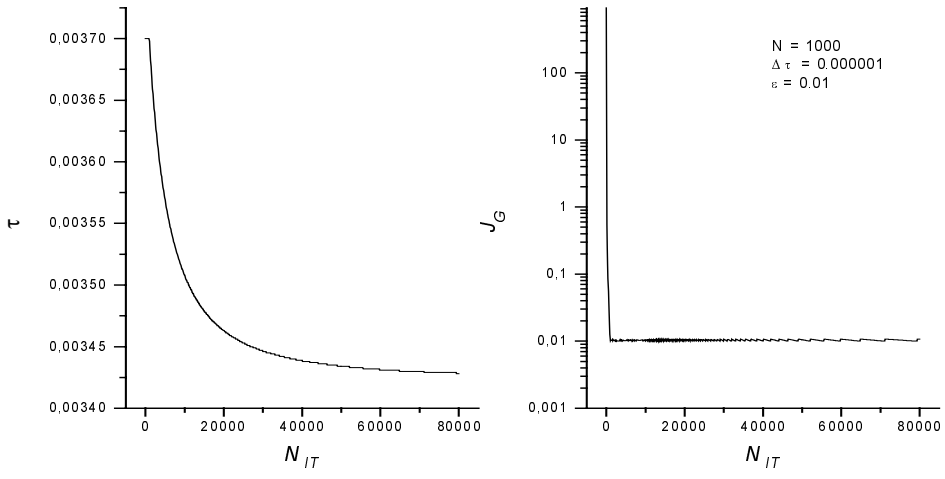
Slike 6.6 do 6.8 prikazuju vrijednosti varijabli stanja i upravljačkih varijabli za parametre prikazane na slici 6.5.

Prednost ovog algoritma, posebno (6.28), je očigledna u odnosu na algoritam opisan u prethodnom podpoglavlju, posebno zbog činjenice da algoritam ne zahtijeva računanje gradijenata i potpuno je neovisan o BPTT algoritmu kojim se računaju vrijednosti upravljačkih varijabli.

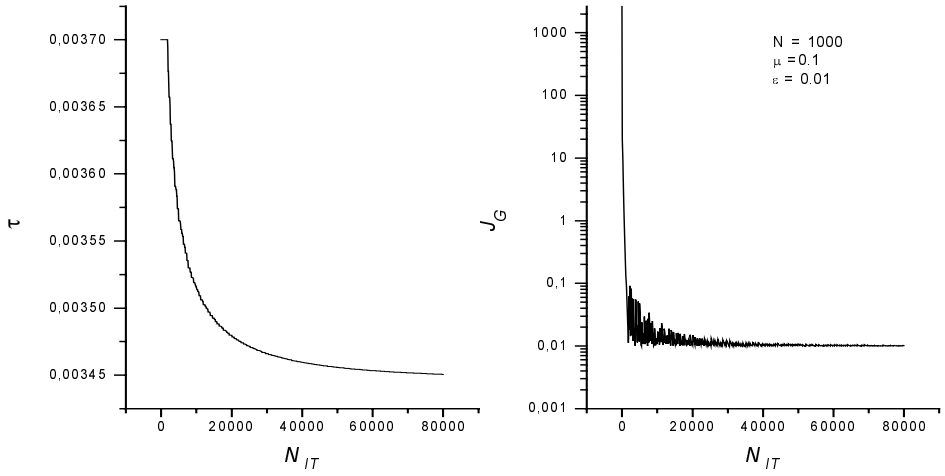
S druge strane, algoritam u obliku (6.27) ili (6.28) analogan je gradijentnom algoritmu i možemo ga lako priključiti strukturi neuronske mreže za BPTT algoritam. Time dobivamo jedinstvenu neuronsku mrežu za rješavanje problema optimalnog upravljanja, kao i vremenski optimalnog upravljanja.



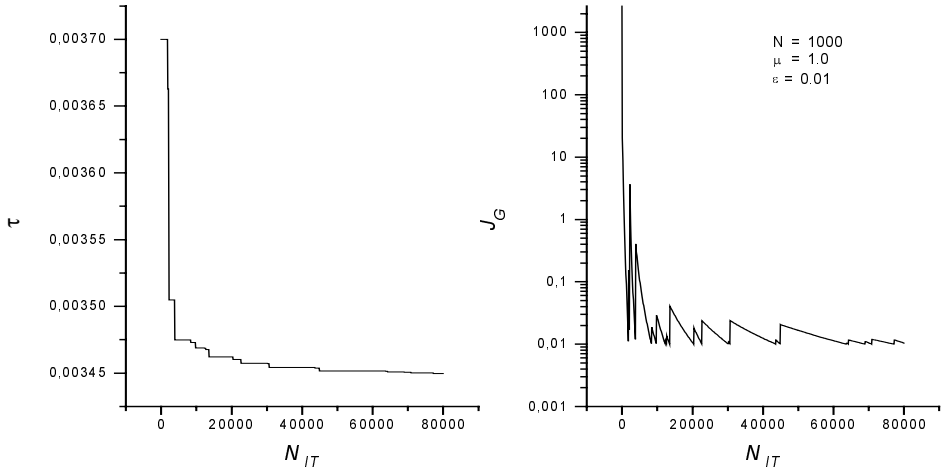
Slika 6.2 *Ovisnost τ i J_G o broju iteracija za (6.27).*



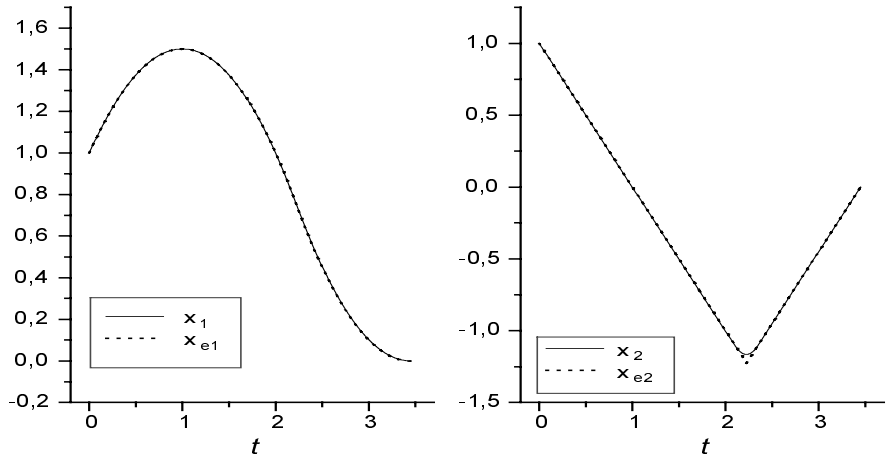
Slika 6.3 *Ovisnost τ i J_G o broju iteracija za (6.27).*



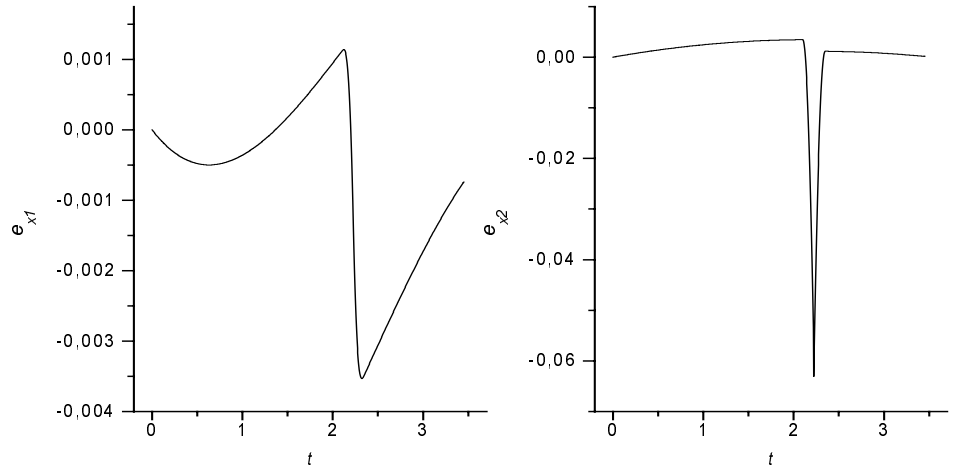
Slika 6.4 *Ovisnost* τ *i* J_G *o broju iteracija za (6.28).*



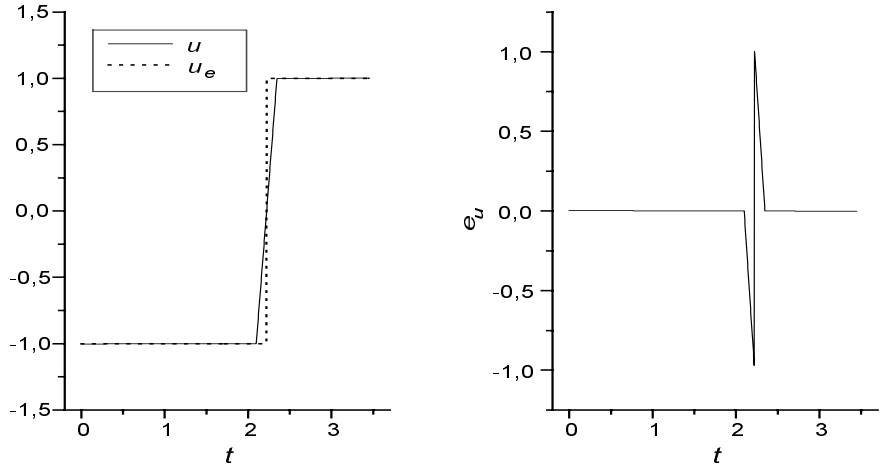
Slika 6.5 *Ovisnost* τ *i* J_G *o broju iteracija za (6.28).*



Slika 6.6 *Usporedba analitičkih rješenja (*x_{e1}, x_{e2}*) s numeričkim (*x_1, x_2*).*



Slika 6.7 *Odstupanja analitičkih rješenja od numeričkih* $\mathbf{e}_x = \mathbf{x}_e - \mathbf{x}$.



Slika 6.8 *Usporedba i međusobno odstupanje analitičkog rješenja* u_e, *od numeričkog* u.

7. Optimalno upravljanje kooperativnim radom dva robota

Optimalno upravljanje nelinearnim modelom robota uz definirani kriterij optimalnosti predstavlja i dalje relativno težak zadatak. Problem se usložnjava kad dva ili više robota rade u kooperaciji na zajedničkom zadatku dijeleći radni prostor, vrijeme, ograničenja i funkciju cilja.

U prethodnim poglavljima prikazali smo numerički algoritam optimalnog upravljanja i ilustrirali ga na linearnim problemima s poznatim analitičkim rješenjima. U ovom poglavlju primjenit ćemo izvedeni algoritam na problem optimalnog upravljanja robotom s dva stupnja slobode, kao i na problem optimalnog koordiniranog rada dva robota.

7.1 Dinamika robota s dva stupnja slobode gibanja

Nelinearni dinamički model manipulatora s dvije slobode gibanja [9], dan je preko cilindričnih koordinata u obliku

$$\begin{bmatrix} M_{11}(\mathbf{q}) & 0 \\ 0 & M_{22}(\mathbf{q}) \end{bmatrix} \begin{bmatrix} \ddot{q}_1 \\ \ddot{q}_2 \end{bmatrix} + \begin{bmatrix} N_1(\mathbf{q},\dot{\mathbf{q}}) \\ N_2(\mathbf{q},\dot{\mathbf{q}}) \end{bmatrix} = \begin{bmatrix} P_1(t) \\ P_2(t) \end{bmatrix}, \tag{7.1}$$

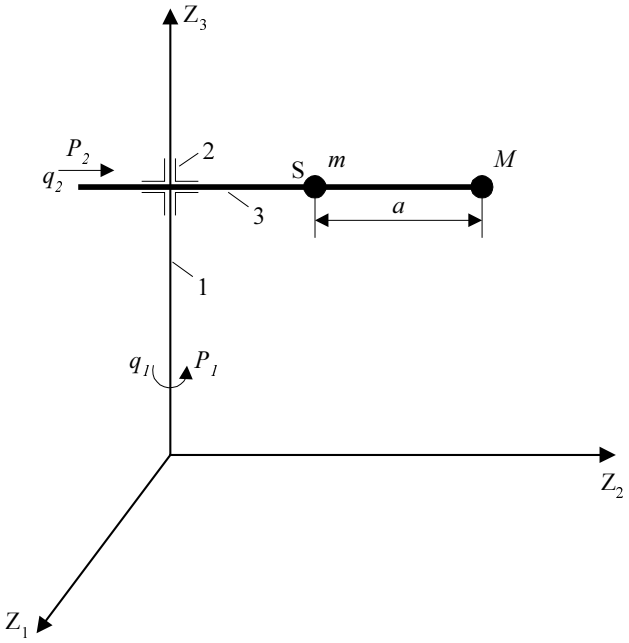
$$\begin{aligned} &M_{11}(\mathbf{q}) = J_1 + J_2 + (m+M)q_2^2 + 2Maq_2 + Ma^2, \qquad M_{22}(\mathbf{q}) = m + M, \\ &N_1(\mathbf{q},\dot{\mathbf{q}}) = 2\left[(m+M)q_2 + Ma\right]\dot{q}_1\dot{q}_2, \\ &N_2(\mathbf{q},\dot{\mathbf{q}}) = -\left[mq_2 + M\left(a+q_2\right)\right]\dot{q}_1^2, \end{aligned} \tag{7.2}$$

gdje su $\mathbf{q} = \begin{bmatrix} q_1 & q_2 \end{bmatrix}^T$ cilindrične koordinate centra mase štapa 3, (slika 7.1), M je totalna masa (prihvatnica plus teret), m je masa štapa, J_1 je totalni moment inercije štapova 1 i 2 u odnosu na os Z_3, J_2 je moment inercije štapa 3 u odnosu na os koja je paralelna osi Z_2 a prolazi kroz točku S, a je udaljenost između centra mase M i točke S. $P_1(t)$ predstavlja upravljački moment rotacijske slobode gibanja q_1, a $P_2(t)$ je upravljačka sila translacijske slobode gibanja q_2.

Numerički iznosi navedenih parametara su:

$M = 50\,kg$; $m = 97\,kg$; $J_1 + J_2 = 193\,kg\,m^2$; $a = 1.1\,m$;

$P_{1\max} = 600\,Nm$; $P_{2\max} = 500\,N$.



Slika 7.1 *Prikaz manipulatora s dva stupnja slobode gibanja (rotacija i translacija).*

Ako želimo gornji sustav diferencijalnih jednadžbi drugog reda prebaciti u sustav jednadžbi prvog reda uvodimo slijedeću smjenu koordinata

$$q_1 = x_1\,;\ \dot{q}_1 = x_2\,;\ q_2 = x_3\,;\ \dot{q}_2 = x_4\ ;\quad P_1(t) = u_1(t)\,;\quad P\ (t) = u\ (t)\,,$$

i zbog preglednijeg zapisa uvodimo slijedeće konstante

$$A_1 = J_1 + J_2 + Ma^2\,;\quad A\ = 2\,Ma\,;\quad A\ = m + M\,,$$

tako da sada imamo

$$\begin{aligned}
\dot{x}_1 &= x_2 \\
\dot{x}_2 &= \frac{-2A_3x_2x_3x_4 - A_2x_2x_4}{A_1 + A_2x_3 + A_3x_3^2} + \frac{u_1(t)}{A_1 + A_2x_3 + A_3x_3^2} \\
\dot{x}_3 &= x_4 \\
\dot{x}_4 &= x_3x_2^2 - \frac{A_2}{2A_3}x_2^2 + \frac{u_2(t)}{A_3}
\end{aligned} \tag{7.3}$$

Ovim smo, dakle, dinamiku robota s dva stupnja slobode gibanja sveli na sustav četiri nelinearne diferencijalne jednadžbe prvog reda, prema (4.35). Nakon diskretizacije, gornji sustav diferencijalnih jednadžbi prelazi u sustav diferencijskih jednadžbi oblika (4.39). Jacobbijeve matrice $\mathbf{X}(j), \mathbf{U}(j)$ za j-ti vremenski trenutak ($t_j = j\tau$, gdje je $\tau = (t_f - t_0)/N$) su

$$\mathbf{X}=\begin{bmatrix} 1 & 0 & 0 & 0 \\ \tau & 1+\tau\dfrac{-C_1x_3x_4-C_2x_4}{A_1+A_2x_3+A_3x_3^2} & 0 & \tau\left(-2\dfrac{D_1}{B}x_2x_3-2\dfrac{D_{21}}{B}x_2\right) \\ 0 & X_{32} & 1 & -\tau\dfrac{D_1}{B}x_2^2 \\ 0 & \tau\dfrac{-C_1x_2x_3-C_2x_2}{A_1+A_2x_3+A_3x_3^2} & \tau & 1 \end{bmatrix}, \tag{7.4}$$

gdje je

$$X_{32}=-\tau\frac{C_1x_2x_4}{A_1+A_2x_3+A_3x_3^2}-\tau\frac{A_2+2A_3x_3}{\left(A_1+A_2x_3+A_3x_3^2\right)^2}\left(-C_1x_2x_3x_4-C_2x_2x_4+u_1(t)\right),$$

i

$$\mathbf{U}=\begin{bmatrix} 0 & \tau\dfrac{1}{A_1+A_2x_3+A_3x_3^2} & 0 & 0 \\ 0 & 0 & 0 & \tau\dfrac{1}{B} \end{bmatrix}. \tag{7.5}$$

7.2 Vremenski optimalno upravljanje robotom s dva stupnja slobode gibanja

Uz poznatu dinamiku robota, preostaje nam još da definiramo kriterij upravljanja na osnovu kojeg ćemo računati upravljačke funkcije $u_1(t)$ i $u\ (t)$. Problem koji ćemo ovdje razmatrati biti će problem najkraćeg vremena potrebnog da robot prijeđe iz početnog u konačno stanje uz zadana ograničenja upravljačkih varijabli. Drugim riječima, treba odrediti upravljačke varijable $u_1(t)$ i $u\ (t)$, za koje će robot prijeći iz početnog stanja

$$x_1(0)=q_1(0)=0\,rad,$$
$$x_2(0)=\dot{q}_1(0)=0\,rad\cdot s^{-1},$$
$$x_3(0)=q_2(0)=0\,m,$$
$$x_4(0)=\dot{q}_2(0)=0\,m\cdot s^{-1},$$

u konačno

$$x_1(t_f)=q_1(t_f)=\frac{\pi}{2}rad,$$
$$x_2(t_f)=\dot{q}_1(t_f)=0\,rad\cdot s^{-1},$$
$$x_3(t_f)=q_2(t_f)=1\,m,$$

$$x_4(t_f) = \dot{q}_2(t_f) = 0\, m \cdot s^{-1},$$

za najkraće vrijeme $t_{\min} \equiv t_f$, uz zadana ograničenja

$$\left| u_1(t) \right| \le u_{1\max},$$

$$\left| u_2(t) \right| \le u_{2\max},$$

gdje je

$$u_{1\max} = 600\, Nm, \qquad u_{2\max} = 500\, N.$$

Ukupna funkcija cilja, J, jednaka je sumi kaznene funkcije za ograničenja, G_V, i kaznene funkcije za rubne uvjete, G_B, $J = G_V + G_B$, gdje je

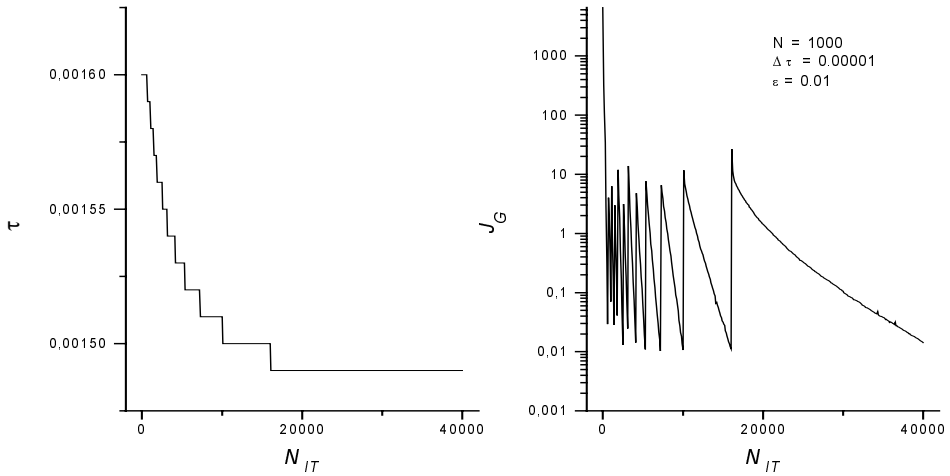
$$G_V = \tau K_V \sum_{i=0}^{N-1} \sum_{k=1}^{2} \left((u_{k\max} - u_i^k)^2 H^-(u_{k\max} - u_i^k) + (u_{k\max} + u_i^k)^2 H^-(u_{k\max} + u_i^k) \right), \tag{7.6}$$

$$G_B = K_B \sum_{k=1}^{4} (x_N^k - x_f^k)^2, \tag{7.7}$$

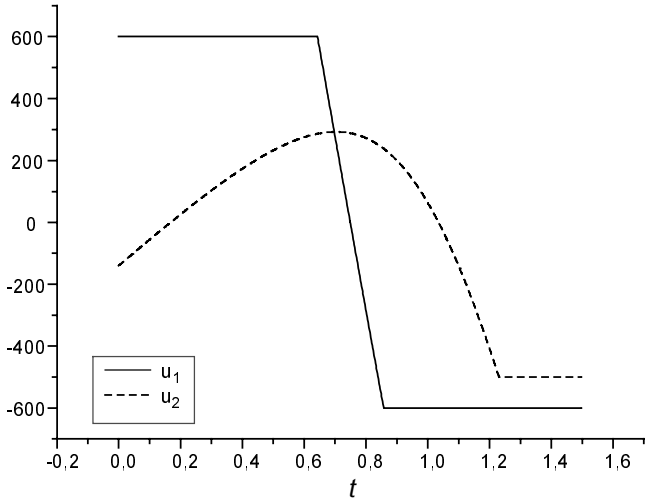
dok smo za funkciju $J_G(\tau)$ uzeli $J_G = \frac{1}{\tau} G_V + G_B$, (faktor $1/\tau$ je zbog toga da bismo izbjegli eksplicitnu ovisnost funkcije $J_G(\tau)$ o τ). Koristili smo metodu konjugiranog gradijenta. Vrijednosti konstanti su

$N = 1000$, $M = 10000$, $K_B = 800$, $K_V = 0.04$.

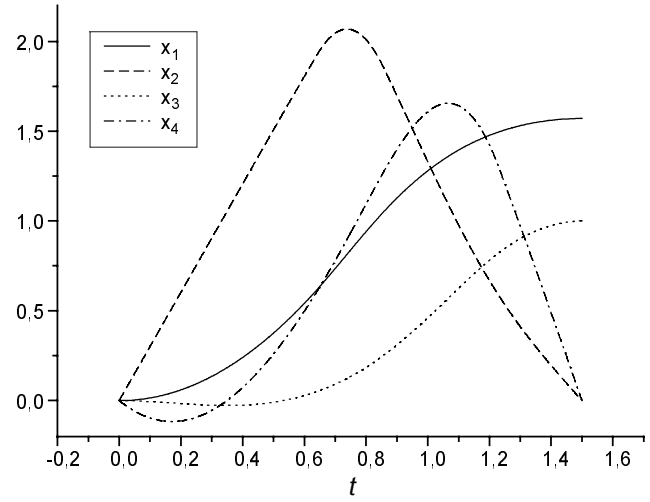
Kao što vidimo na slici 7.2, uz zadanu točnost $\varepsilon \equiv J_{\min} = 0.01$ dobili smo minimalno vrijeme $t_{\min} = 1.5\, s$, odnosno $\tau_{\min} = t_{\min} / N$. Zatim smo za to vrijeme, pomoću BPTT algoritma uz $M = 10000$, dobili rezultate prikazane na slikama 7.3 do 7.5.



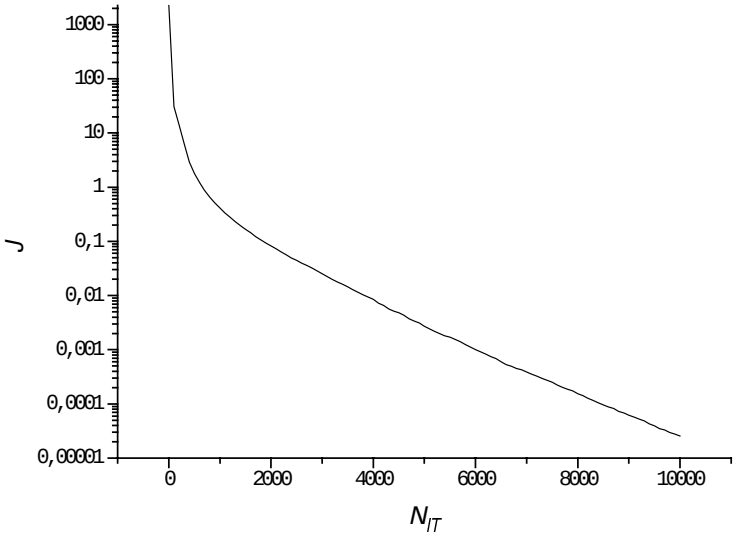
Slika 7.2 *Ovisnost τ i J_G o broju iteracija.*



Slika 7.3 *Vremenska ovisnost funkcija upravljanja.*



Slika 7.4 *Vremenska ovisnost varijabli stanja.*



Slika 7.5 *Ovisnost* J *o broju iteracija.*

7.3 Izbjegavanje prepreka

Jedan od zadataka koji postavljamo robotu jeste zaobilaženje prepreka. Pretpostavlja se poznavanje položaja i oblika prepreke. Zbog jednostavnosti, kao prepreku uzet ćemo kružnicu radijusa R', s centrom u točki (x_0, y_0). Označimo minimalnu udaljenost između kružnice i ruke manipulatora s δR. To je udaljenost do koje dopuštamo ruci manipulatora da se približi. Sada imamo kružnicu radijusa $R = R' + \delta R$ čija je jednadžba

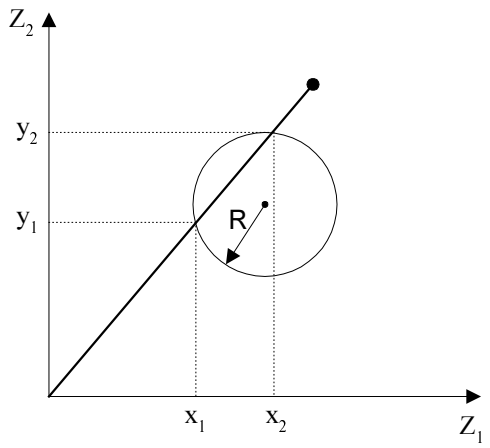
$$(x - x_0)^2 + (y - y_0)^2 = R^2 \ . \tag{7.8}$$

Ruku manipulatora možemo prikazati kao dužinu koja leži na pravcu

$$y = \tan q_1 \cdot x \ , \tag{7.9}$$

tako da iz gornje dvije jednadžbe možemo naći uvjete presjeka kružnice i pravca, kao i same koordinate točaka presjeka. Uvjet da bi pravac sjekao kružnicu jeste da je vrijednost determinante D veća ili jednaka nuli

$$D = 4(x_0 + y_0 \tan q_1)^2 - 4(1 + \tan^2 q_1)(x_0^2 + y_0^2 - R^2) \geq 0 . \tag{7.10}$$



Slika 7.6 *Prikaz presjeka ruke manipulatora i prepreke.*

U tom slučaju, možemo naći koordinate presjeka

$$x_{1,2} = \frac{2\left(x_0 + y_0 \tan q_1\right) \pm \sqrt{D}}{2\left(1 + \tan^2 q_1\right)} \qquad (7.11)$$
$$y_{1,2} = \tan q_1 \cdot x_{1,2}$$

odnosno, udaljenosti točaka presjeka od ishodišta koordinatnog sustava

$$r_1 = \sqrt{x_1^2 + y_1^2}\ , \qquad r_2 = \sqrt{x_2^2 + y_2^2}\ . \qquad (7.12)$$

Ako sa $r_{\min} = \min\left\{r_1, r_2\right\}$ označimo manju od te dvije udaljenosti, tada možemo formulirati uvjet koji će nam garantirati da ruka manipulatora zaobilazi prepreku u *i*-tom vremenskom trenutku

$$a + q_i^2 \le r_{\min}(q_i^1)\,, \qquad (7.13)$$

za $i = 0,1,\ldots,N$, tako da možemo za kaznenu funkciju napisati

$$G_p = K_p \sum_{i=0}^{N} \left(r_{\min}(q_i^1) - q_i^2 - a\right)^2 H^-\left(r_{\min}(q_i^1) - q_i^2 - a\right) H^+\left(D_i\right), \qquad (7.14)$$

gdje je

$$D_i = 4\left(x_0 + y_0 \tan q_i^1\right)^2 - 4\left(1 + \tan^2 q_i^1\right)\left(x_0^2 + y_0^2 - R^2\right), \qquad (7.15)$$

i

$$H^+(x) = H^-(-x)\,.$$

Parcijalne derivacije kaznene funkcije po varijablama stanja su

$$\frac{\partial G_p}{\partial q_i^1} = 2K_p\left(r_{\min}(q_i^1) - q_i^2 - a\right)\frac{\partial r_{\min}(q_i^1)}{\partial q_i^1} H^-\left(r_{\min}(q_i^1) - q_i^2 - a\right) H^+\left(D_i\right), \qquad (7.16)$$

$$\frac{\partial G_p}{\partial q_i^2} = -2K_p\left(r_{\min}(q_i^1) - q_i^2 - a\right) H^-\left(r_{\min}(q_i^1) - q_i^2 - a\right) H^+\left(D_i\right), \qquad (7.17)$$

Ako pretpostavimo da se kružnica nalazi u prvom kvadrantu, odnosno x_0, y_0, $\tan q_1$ su pozitivni, tada je očigledno da je $r_{\min} = r_2$, odnosno

$$r_2 = \sqrt{x_2^2 + y_2^2} = x_2\sqrt{1+\tan^2 q_1} = \\ = \frac{(x_0 + y_0 \tan q_1) - \sqrt{(x_0 + y_0 \tan q_1)^2 - (1+\tan^2 q_1)(x_0^2 + y_0^2 - R^2)}}{\sqrt{1+\tan^2 q_1}}$$

što nagovještava poprilično složenost parcijalne derivacije $\frac{\partial r_{\min}}{\partial q_1} = \frac{\partial r_2}{\partial q_1}$.

Da bi izbjegli komplikacije oko računanja gore navedene parcijalne derivacije, uzet ćemo samo promjenu radijusa uslijed kršenja ograničenja, odnosno

$$\frac{\partial G_p}{\partial q_i^1} = 0, \tag{7.18}$$

$$\frac{\partial G_p}{\partial q_i^2} = -2K_p\left(r_2 - q_i^2 - a\right)H^-\left(r_2 - q_i^2 - a\right)H^+\left(D_i\right), \tag{7.19}$$

ili, u drugom slučaju, još jednostavniji oblik

$$\frac{\partial G_p}{\partial q_i} = 0, \tag{7.20}$$

$$\frac{\partial G_p}{\partial q_i^2} = K_p H^-\left(r_2 - q_i^2 - a\right)H^+\left(D_i\right), \tag{7.21}$$

Kao što ćemo vidjeti, gornje aproksimacije ne utječu na kvalitetu rješenja, osim što eventualno smanjuju brzinu konvergencije optimalnom rješenju. Izrazi (7.19) i (7.20) govore nam da u slučaju presjeka ruke manipulatora s preprekom, kut q ne mjenjamo, dok duljinu ruke, odnosno q_2, smanjujemo konstantnom stopom smanjivanja izraženom pozitivnim koeficijentom K_p. Time smo problem bitno pojednostavili i sveli ga samo na geometrijsko određivanje presjeka pravca na kojem leži ruka manipulatora s rubom prepreke.

Međutim, izbjegavši komplikacije oko egzaktnog računanja parcijalnih derivacija, došli smo do drugog problema. Metoda konjugiranog gradijenta nam u ovom slučaju ne daje dobre rezultate jer gore navedene parcijalne derivacije ne pripadaju kaznenoj funkciji. Zbog toga koristimo gradijentni algoritam s konstantnim koeficijentom konvergencije.

Problem koji smo razmatrali je prijelaz robota iz početnog stanja

$$x_1(0) = q_1(0) = \frac{\pi}{12} rad \quad ,$$

$$x_2(0) = \dot{q}_1(0) = 0\, rad \cdot s^{-1},$$

$$x_3(0) = q_2(0) = 1\, m,$$

$$x_4(0) = \dot{q}_2(0) = 0\, m \cdot s^{-1},$$

u konačno

$$x_1(t_f) = q_1(t_f) = \frac{5\pi}{12} rad\,,$$

$$x_2(t_f) = \dot{q}_1(t_f) = 0\, rad \cdot s^{-1},$$

$$x_3(t_f) = q_2(t_f) = 1\, m,$$

$$x_4(t_f) = \dot{q}_2(t_f) = 0\, m \cdot s^{-1},$$

izbjegavajući prepreku u obliku kružnice radijusa $R' = 0.2\, m$, s koordinatama središta kružnice u $x_0 = 1.0\, m$, $y_0 = 1.0\, m$, te zadanom minimalnom udaljenošću $\delta R = 0.01\, m$, za najkraće vrijeme $t_{\min} \equiv t_f$, uz zadana ograničenja

$$\left| u_1(t) \right| \le u_{1\max},$$

$$\left| u_2(t) \right| \le u_{2\max},$$

gdje je

$$u_{1\max} = 600\, Nm, \qquad u_{2\max} = 500\, N\,.$$

Funkcija cilja jednaka je $J = G_V + G_p + G_B$, gdje je G_V kaznena funkcija za ograničenja upravljačkih varijabli dana sa (7.6), a G_p je kaznena funkcija za ograničenja varijabli stanja koja je jednaka

$$G_p = \tau K_p \sum_{i=0}^{N} (r_{\min} - a - q_i^2)^2\, H^-\left(r_{\min} - a - q_i^2\right) H^+\left(D_i\right), \tag{7.22}$$

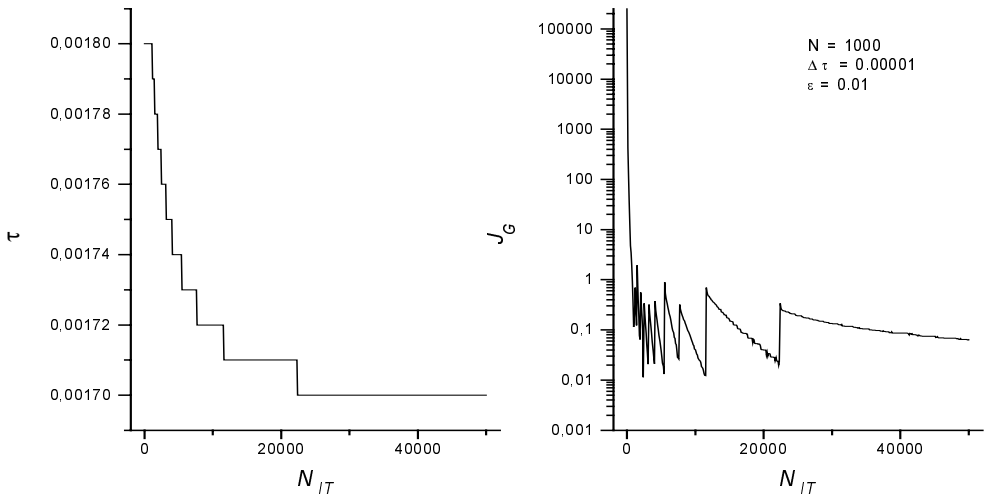
Za funkciju $J_G(\tau)$ uzeli smo isti oblik kao za J, s tom razlikom što smo umjesto promjenjive vrijednosti τ stavili konstantnu τ^0, početnu vrijednost iteracijskog procesa za računanje τ_{in} (zbog izbjegavanja eksplicitne ovisnosti $J_G(\tau)$ o τ). Koristili smo gradijentni algoritam s konstantnim koeficijentom konvergencije η.

Vrijednosti konstanti su

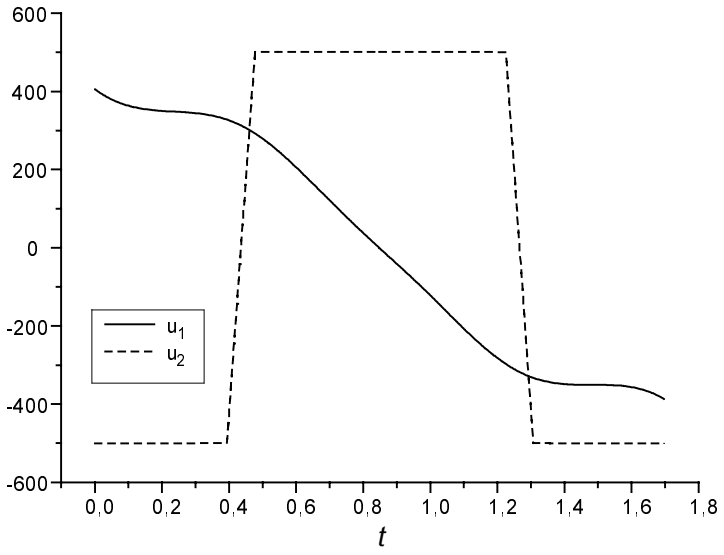
$N = 1000$, $M = 30000$, $K_B = 600$, $K_V = 0.01$, $K_p = 1000$, $\eta = 1000$.

Kao što vidimo na slici 7.7, uz zadanu točnost $\varepsilon \equiv J_{\text{in}} = 0.01$ dobili smo minimalno vrijeme $t_{\min} = 1.71\, s$. Za to vrijeme, $t_{\min} = 1.71\, s$, pomoću BPTT algoritma uz $M = 30000$, dobili smo rezultate prikazane na slikama 7.8 do 7.11.

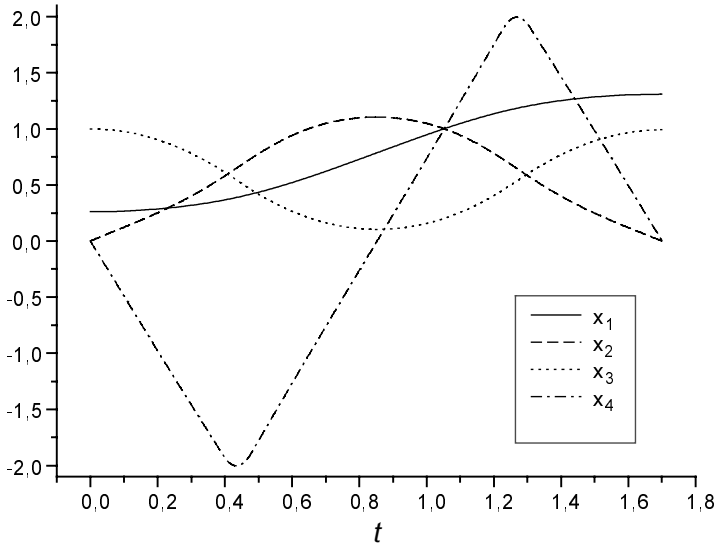
Slika 7.11 prikazuje trajektoriju ruke robota ($q + a$) u ravnini $x - y$. Prepreka je kružnica radijusa $R' = 0.2\, m$ (*prepreka 1*). Sa zadanom minimalnom udaljenošću imamo kružnicu radijusa $R = 0.21\, m$ (*prepreka 2*). Vidimo da optimalna trajektorija (*trajektorija 1*) tangira kružnicu radijusa R, odnosno, zaobilazi prepreku dostigavši minimalnu udaljenost u jednoj točki. Na slici je također prikazana trajektorija, za isto vrijeme $t_{\min} = 1.71\, s$, u slučaju kada nema prepreke.



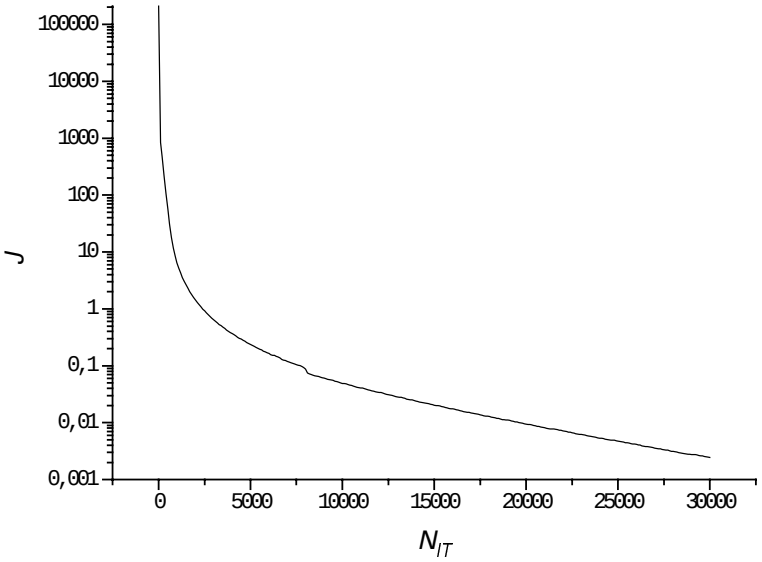
Slika 7.7 *Ovisnost* τ *i* J_G *o broju iteracija.*



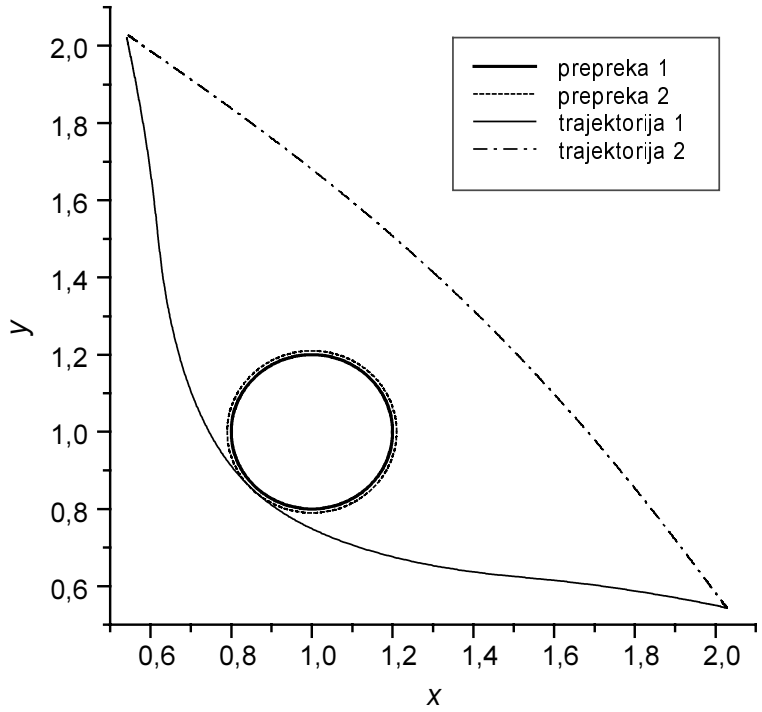
Slika 7.8 *Vremenska ovisnost funkcija upravljanja.*



Slika 7.9 *Vremenska ovisnost varijabli stanja.*



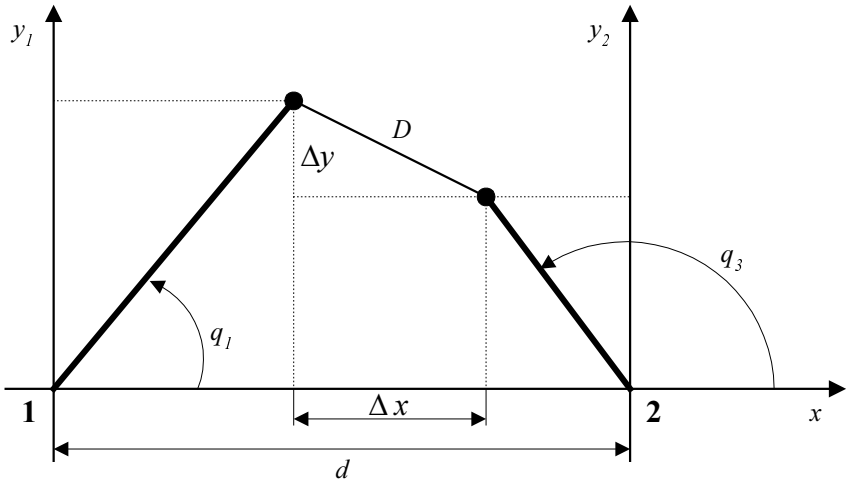
Slika 7.10 *Ovisnost* J *o broju iteracija.*



Slika 7.11 *Prikaz trajektorije s preprekom u koordinatnom sustavu.*

7.4 Kooperativni rad dva robota

Slijedeći problem koji ćemo razmatrati je kooperativni rad dva robota. Pod kooperativnim radom podrazumjevamo međusobno usklađeni rad robota na zajednički definiranom cilju. U našem slučaju razmatrat ćemo problem prenošenja krutog tereta s jednog mjesta na drugo pomoću dva robota. Pretpostavljamo da teret ima dva hvatišta za prenošenje. Pošto se radi o krutom teretu, razmak između hvatišta je konstantan tokom prenošenja tereta, što znači da je uvjet koji moramo postaviti robotima konstantna udaljenost između ruku (prihvatnica) manipulatora za vrijeme prenošenja tereta.



Slika 7.12 *Prijenos krutog tereta duljine* D *pomoću dva robota.*

Jednadžbama gibanja prvog robota (7.3), analogno dodajemo jednadžbe gibanja drugog robota

$$\begin{aligned}
\dot{x}_5 &= x_6 \\
\dot{x}_6 &= \frac{-2A_3x_6x_7x_8 - A_2x_6x_8}{A_1 + A_2x_7 + A_3x_7^2} + \frac{u_3(t)}{A_1 + A_2x_7 + A_3x_7^2} \\
\dot{x}_7 &= x_8 \\
\dot{x}_8 &= x_7x_6^2 - \frac{A_2}{2A_3}x_6^2 + \frac{u_4(t)}{A_3}
\end{aligned} \tag{7.23}$$

gdje je $q_3 = x_5$; $\dot{q}_3 = x_6$; $q_4 = x_7$; $\dot{q}_4 = x_8$ (q_3 i q_4 drugog robota imaju analogna značenja kao q_1 i q_2 za prvog robota, respektivno). Treba naglasiti da se varijable stanja drugog robota odnose na njegov vlastiti koordinatni sustav (u slučaju kada roboti nisu mobilni), na slici 7.12 sustav 2.

Razmak između ruku manipulatora dva robota u *i*-tom vremenskom trenutku možemo dobiti kao

$$r_i = \sqrt{\Delta x_i^2 + \Delta y_i^2}\,, \tag{7.24}$$

gdje je

$$\Delta x_i = d + (x_i^7 + a)\cos x_i^5 - (x_i^3 + a)\cos x_i^1\,, \tag{7.25}$$

$$\Delta y_i = (x_i^7 + a)\sin x_i^5 - (x_i^3 + a)\sin x_i^1\,. \tag{7.26}$$

Uvjet da je razmak između ruku manipulatora dva robota u *i*-tom vremenskom trenutku, r_i, jednak razmaku između hvatišta, *D*, možemo izraziti kao

$$r_i = D\,, \tag{7.27}$$

za $i = 0,1,\ldots,N$. Na osnovu gore navedenog uvjeta možemo formirati kaznenu funkciju čijom minimizacijom zadovoljavamo postavljena ograničenja

$$G_E = K_E\sum_{i=0}^{N}(r_i - D)^2\,. \tag{7.28}$$

Parcijalne derivacije gornje funkcije po varijablama stanja su

$$\frac{\partial G_E}{\partial x_i^k} = 2K_ET(r_i - D)\frac{\partial r_i}{\partial x_i^k}\,, \tag{7.29}$$

za $i = 0,1,\ldots,N$, $k = 1,2,\ldots,8$, gdje je

$$\frac{\partial r_i}{\partial x_i^k} = \frac{1}{r_i}\left(\Delta x_i\frac{\partial \Delta x_i}{\partial x_i^k} + \Delta y_i\frac{\partial \Delta y_i}{\partial x_i^k}\right), \tag{7.30}$$

odnosno

$$\frac{\partial \Delta x_i}{\partial x_i^1} = (x_i^3 - a)\sin x_i^1\,, \qquad \frac{\partial \Delta y_i}{\partial x_i^1} = -(x_i^3 + a)\cos x_i^1\,,$$

$$\frac{\partial \Delta x_i}{\partial x_i^3} = -\cos x_i^1, \qquad \frac{\partial \Delta y_i}{\partial x_i^3} = -\sin x_i^1,$$

$$\frac{\partial \Delta x_i}{\partial x_i^5} = -(x_i^7 - a)\sin x_i^5, \qquad \frac{\partial \Delta y_i}{\partial x_i^5} = (x_i^7 + a)\cos x_i^5,$$

$$\frac{\partial \Delta x_i}{\partial x_i^7} = \cos x_i^5, \qquad \frac{\partial \Delta y_i}{\partial x_i^7} = \sin x_i^5,$$

dok su ostale parcijalne derivacije jednake nuli.

Problem koji smo razmatrali je prenošenje tereta iz početnog položaja, definiranog početnim uvjetima oba robota

$$x_1(0) = q_1(0) = 0\,rad \quad , \qquad x_5(0) = q_3(0) = \pi\,rad \quad ,$$

$$x_2(0) = \dot{q}_1(0) = 0\,rad \cdot s^{-1}, \qquad x_6(0) = \dot{q}_3(0) = 0\,rad \cdot s^{-1},$$

$$x_3(0) = q_2(0) = 0.4\,m, \qquad x_7(0) = q_4(0) = 0.2\,m,$$

$$x_4(0) = \dot{q}_2(0) = 0\,m \cdot s^{-1}, \qquad x_8(0) = \dot{q}_4(0) = 0\,m \cdot s^{-1},$$

u konačni položaj, definiran rubnim uvjetima oba robota

$$x_1(t_f) = q_1(t_f) = \frac{\pi}{2} rad, \qquad x_5(t_f) = q_3(t_f) = 2.62604\,rad,$$

$$x_2(t_f) = \dot{q}_1(t_f) = 0\,rad \cdot s^{-1}, \qquad x_6(t_f) = \dot{q}_3(t_f) = 0\,rad \cdot s^{-1},$$

$$x_3(t_f) = q_2(t_f) = 0.4\,m, \qquad x_7(t_f) = q_4(t_f) = 2.34818\,m,$$

$$x_4(t_f) = \dot{q}_2(t_f) = 0\,m \cdot s^{-1}, \qquad x_8(t_f) = \dot{q}_4(t_f) = 0\,m \cdot s^{-1},$$

uz zadana ograničenja upravljačkih varijabli

$$\left| u_1(t) \right| \le u_{1\max}, \qquad \left| u_3(t) \right| \le u_{1\max},$$

$$\left| u_2(t) \right| \le u_{2\max}, \qquad \left| u_4(t) \right| \le u_{2\max},$$

gdje je

$$u_{1\max} = 600\,Nm, \qquad u_{2\max} = 500\,N,$$

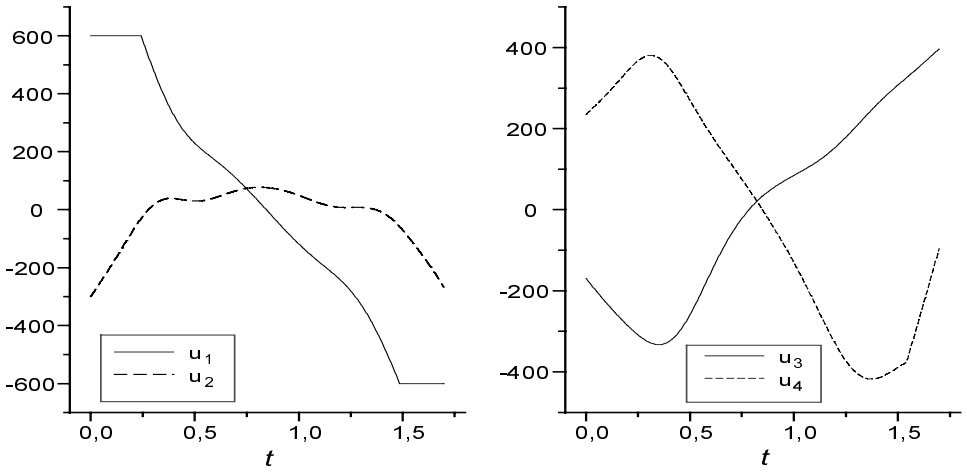
unutar vremenskog intervala $t_f = 1.7\,s$.

Ukupna funkcija cilja, J, jednaka je sumi kaznene funkcije za ograničenja, G_V, kaznene funkcije za rubne uvjete, G_B, te kaznene funkcije ograničenja varijabli stanja, G_E, odnosno $J = G_V + G_B + G_E$.

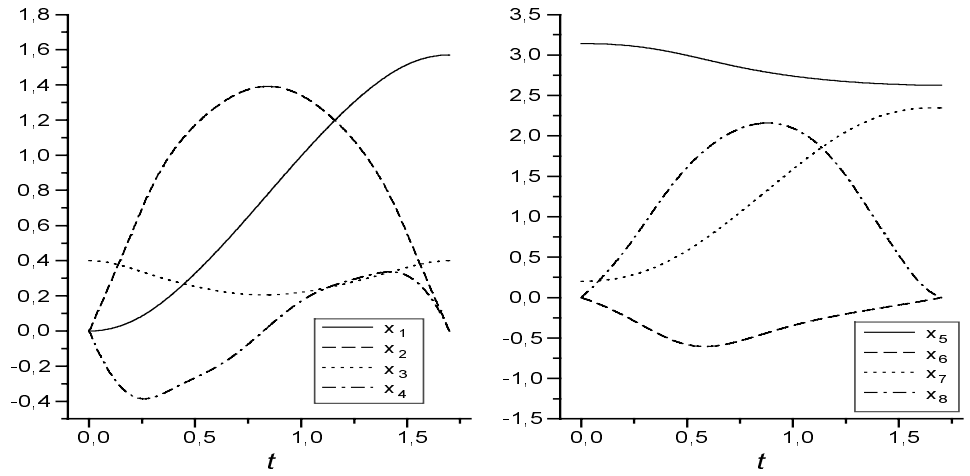
Koristili smo gradijentnu metodu s konstantnim koeficijentom konvergencije. Vrijednosti konstanti koje smo koristili su

$N = 1000$, $M = 30000$, $K_B = 100$, $K_V = 0.01$, $K_E = 1000$, $\eta = 3000$.

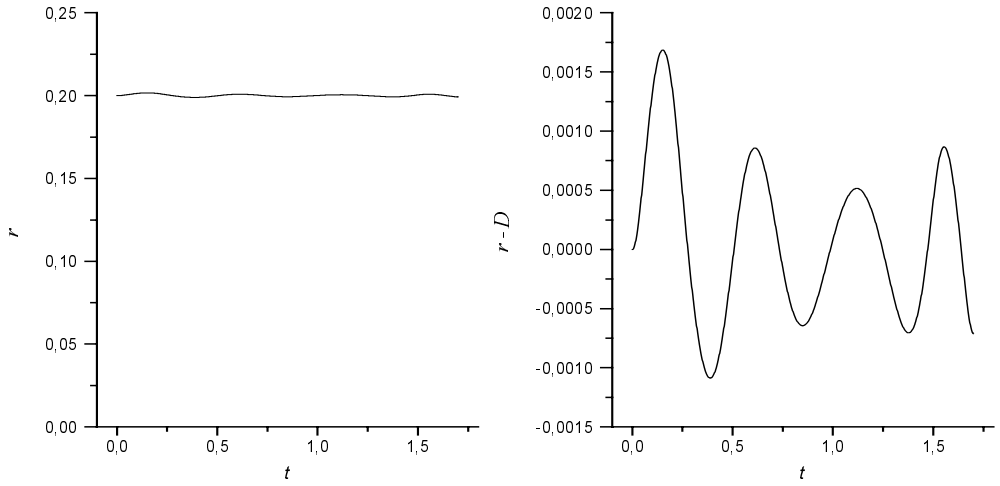
Slike 7.13 i 7.14 prikazuju vremensku ovisnost upravljačkih funkcija, odnosno varijabli stanja, dok slika 7.15 prikazuje vremensku promjenu međusobne udaljenosti prihvatnica dva robota i odstupanje te vrijednosti od definirane udaljenosti među hvatištima tereta koji se prenosi. Vidimo da postoje male oscilacije oko definirane međusobne udaljenosti koje se mogu kompenzirati pomičnim hvatištima tereta ili elastičnošću prihvatnice manipulatora.



Slika 7.13 *Vremenska ovisnost upravljačkih funkcija 1. i 2. robota.*



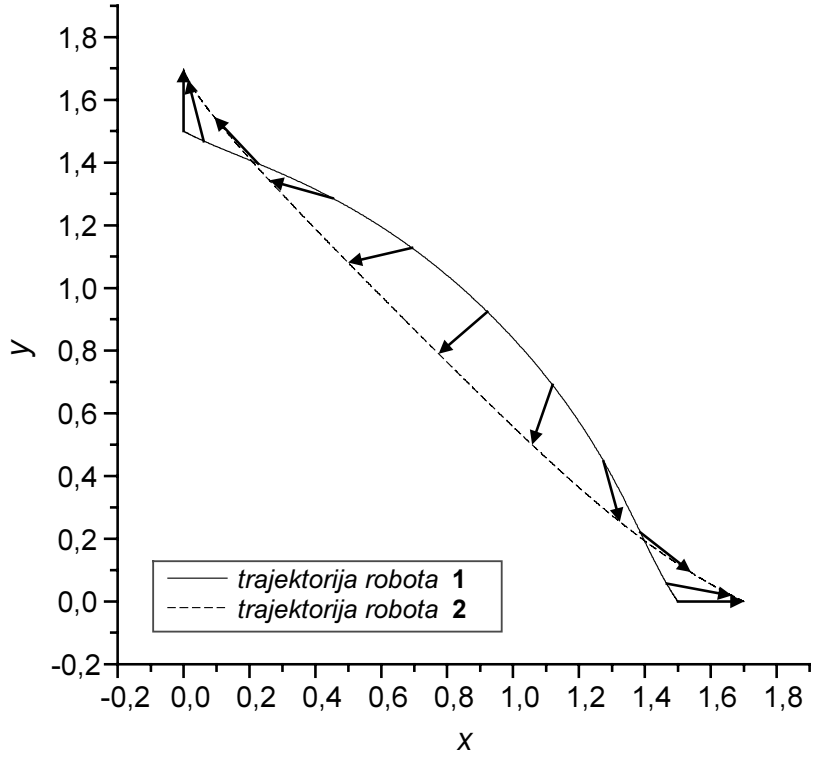
Slika 7.14 *Vremenska ovisnost varijabli stanja 1. i 2. robota.*



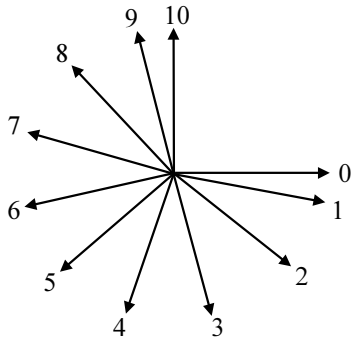
Slika 7.15 *Udaljenost između ruku manipulatora dva robota,* r*, te razlika između te udaljenosti i razmaka između hvatišta krutog tereta,* D.

Slika 7.16 prikazuje trajektorije oba robota u koordinatnom sustavu prvog robota. Također, prikazani su položaji tereta u različitim vremenskim trenucima tokom prijenosa, a na slici 7.17 prikazana je rotacija tereta tokom prijenosa u nepomičnom koordinatnom sustavu. Vidimo da tokom prijenosa iz početnog u krajnji položaj dolazi do rotacije tereta u jednom smjeru za kut od $3\pi/2\,rad$. Iz slike se vidi da u jednom vremenskom intervalu, koji počinje od sredine trajektorije prikazane na slici (odnosno, za kut rotacije tereta od $3\pi/4\,rad$), dolazi do presjeka ruku manipulatora dva robota, što je preciznije prikazano na slici 7.18. Slika 7.19 ilustrira brzinu konvergencije ukupne funkcije cilja.

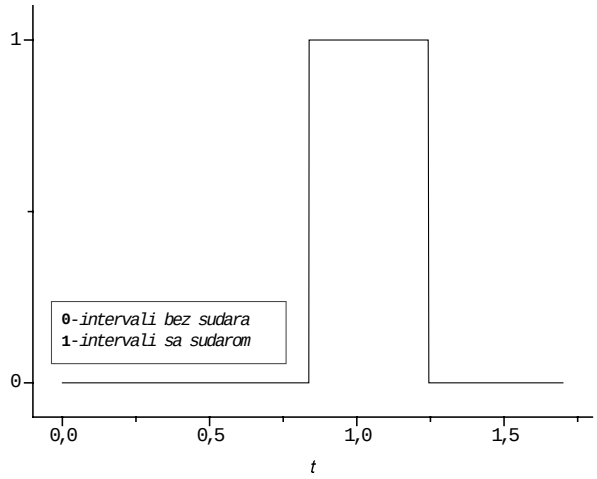
Iz gore navedenog vidimo da, bez obzira što smo dobili dobre rezultate u smislu zadovoljenja postavljenih uvjeta i ograničenja, pojavio se problem međusobnog sudara ruku robota koji naravno u realnoj situaciji moramo izbjeći. Taj problem tretiramo u slijedećem podpoglavlju.



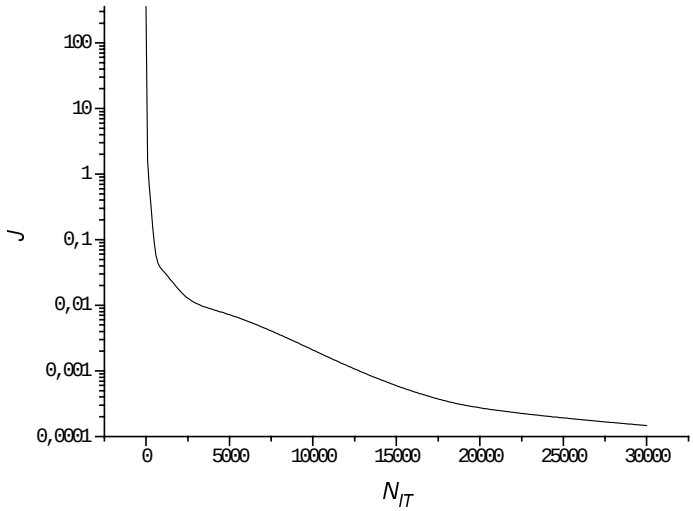
Slika 7.16 *Trajektorije dva robota s prikazom položaja tereta u različitim vremenskim trenucima.*



Slika 7.17 *Prikaz rotacije tereta tokom prijenosa u odnosu na nepomični koordinatni sustav.*



Slika 7.18 *Prikaz vremenskih intervala unutar kojih dolazi do sudara (nivo* 1*), odnosno, ne dolazi do sudara (nivo* 0*).*



Slika 7.19 *Ovisnost ukupne funkcije cilja,* J *, o broju iteracija,* N_{IT} *.*

7.5 Izbjegavanje međusobnih sudara dva robota

Kada imamo dva robota u kooperativnom radu tada je nužno osigurati izbjegavanje međusobnih sudara. Ako imamo dva manipulatora s dvije slobode gibanja, međusobno udaljena za *d* (slika 7.20), rotacionu slobodu gibanja prvog robota označimo sa q_1, a translacionu slobodu gibanja sa q_2, dok ćemo rotacionu slobodu gibanja drugog robota označiti sa q_3, a translacionu slobodu gibanja sa q_4.

Tada ruka manipulatora 1 predstavlja dužinu koja leži na pravcu

$$y = \tan q_1 \cdot x\,, \tag{7.31}$$

dok ruka manipulatora 2 predstavlja dužinu koja leži na pravcu

$$y = \tan q_3 \cdot (x - d). \tag{7.32}$$

Presjek ta dva pravca je u točki (x_p, y_p), gledano iz koordinatnog sustava prvog robota

$$x_p = d \frac{\tan q_3}{\tan q_3 - \tan q_1}, \tag{7.33}$$

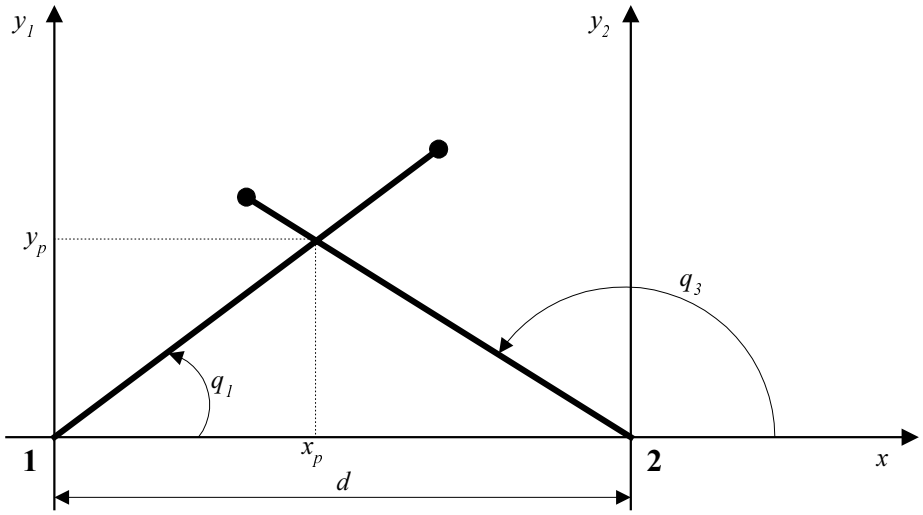
$$y_p = d \frac{\tan q_1 \cdot \tan q_3}{\tan q_3 - \tan q_1}. \tag{7.34}$$

U cilindričnim koordinatama, točka presjeka iz koordinatnog sustava prvog robota je

$$r_{1p} = \sqrt{x_p^2 + y_p^2}, \tag{7.35}$$

a iz koordinatnog sustava drugog robota

$$r_{2p} = \sqrt{(x_p - d)^2 + y_p^2}. \tag{7.36}$$



Slika 7.20 *Prikaz presjeka ruku dva manipulatora.*

Ako sada uzmemo pravac na kome leži ruka manipulatora 1 za *x*-os novog koordinatnog sustava koji se dobije rotacijom sustava $y_1 - x$ za kut q_1 (slika 7.21), tada vidimo da ruka manipulatora leži na pozitivnoj strani *x*-osi od ishodišta do točke $a + q_2$. Ako točku (x_p, y_p) u sustavu $y - x$ transformiramo u sustav $y_1^* - x_1^*$, dobivamo točku $(x_{1p}^*, 0)$. Veličina x_{1p}^* je po apsolutnoj vrijednosti jednaka s r_{1p}, ali može biti negativna i u tom slučaju nema presjeka s rukom manipulatora 1. Analognu situaciju imamo s drugim manipulatorom.

Ruke manipulatora sjeći će se u slučaju da su *istovremeno* zadovoljene slijedeće nejednakosti

$$1.\ 0 \le x_{1p}^{*}(q_1, q_3) \le q_2 + a\,, \tag{7.37}$$

$$2.\ 0 \le x_{2p}^{*}(q_1, q_3) \le q_4 + a\,, \tag{7.38}$$

odnosno

$$1.\ a + q_2 - x_{1p}^{*} \ge 0\,, \tag{7.39}$$

$$2.\ x_{1p}^{*} \ge 0\,, \tag{7.40}$$

$$3.\ a + q_4 - x_{2p}^{*} \ge 0\,, \tag{7.41}$$

$$4.\ x_{2p}^{*} \ge 0\,. \tag{7.42}$$

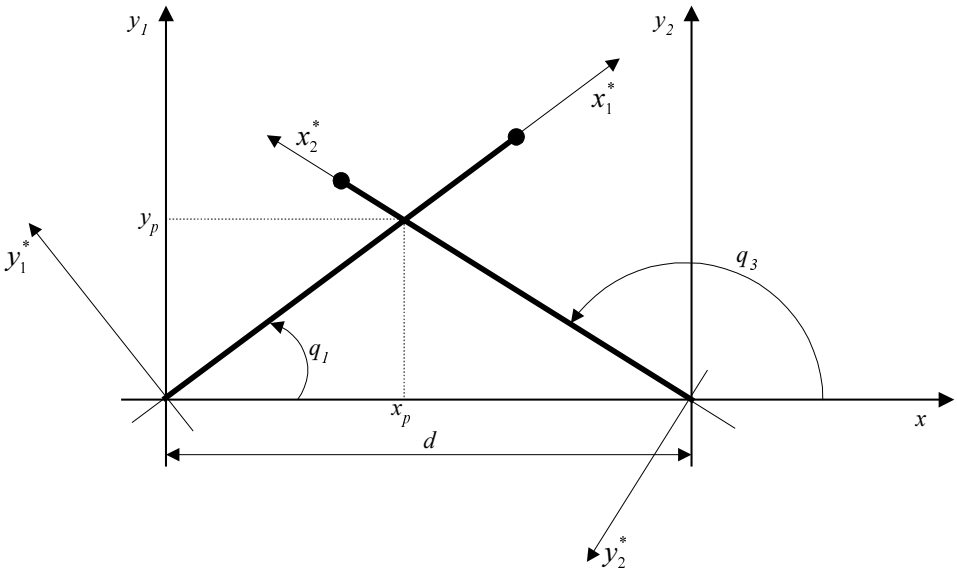
gdje je

$$x_{1p}^{*}(q_1, q_3) = x_p \cos q_1 + y_p \sin q_1\,, \tag{7.43}$$

$$x_{2p}^{*}(q_1, q_3) = (x_p - d)\cos q_3 + y_p \sin q_3\,, \tag{7.44}$$

uključujući nužni uvjet da bi uopće bilo presjeka

$$\tan q_1 \ne \tan q_3\,. \tag{7.45}$$



Slika 7.21 *Prikaz zarotiranih koordinatnih sustava manipulatora.*

Iz svega navedenog slijedi da je dovoljan uvjet izbjegavanja sudara zadovoljenje *bilo koje* od slijedećih nejednakosti u *i*-tom vremenskom trenutku

$$1.\ x_{1p}^{*}(q_i^1, q_i^2) - a - q_i^2 \geq 0, \tag{7.46}$$

$$2.\ x_{1p}^{*}(q_i^1, q_i^2) \leq 0, \tag{7.47}$$

$$3.\ x_{2p}^{*}(q_i^1, q_i^2) - a - q_i^4 \geq 0, \tag{7.48}$$

$$4.\ x_{2p}^{*}(q_i^1, q_i^2) \leq 0, \tag{7.49}$$

za $i = 0,1,\ldots,N$.

Na osnovi gornjih nejednakosti možemo formirati kaznenu funkciju

$$G_S = K_S \sum_{i=0}^{N} F_G(q_i^1, q_i^2, q_i^3, q_i^4) H^*(q_i^1, q_i^2, q_i^3, q_i^4), \tag{7.50}$$

gdje je

$$F_G(q_i^1, q_i^2, q_i^3, q_i^4) = \left(x_{1p}^{*} - q_i^2 - a\right)^2 + \left(x_{2p}^{*} - q_i^4 - a\right)^2 + x_{1p}^{*\,2} + x_{2p}^{*\,2}, \tag{7.51}$$

$$H^*(q_i^1, q_i^2, q_i^3, q_i^4) = H^-(x_{1p}^{*} - q_i^2 - a) H^-(x_{2p}^{*} - q_i^4 - a) H^+(x_{1p}^{*}) H^+(x_{2p}^{*}). \tag{7.52}$$

Kaznenu funkciju oblika (7.52) uveli smo zbog toga što u slučaju zadovoljavanja *bilo koje* nejednadžbe (7.46)-(7.49), jedna od kaznenih funkcija u produktu biti će jednaka nuli, a time će ujedno cijeli izraz (7.52) biti jednak nuli.

Sada ponovo, kao u slučaju zaobilaženja prepreke, dolazimo do problema prekompliciranosti i preglomaznosti izraza za parcijalne derivacije funkcije (7.50) po varijablama stanja. Da bi dobili parcijalne derivacije funkcije (7.50), trebamo uzeti u obzir izraze (7.33), (7.34), (7.43), (7.44) i (7.50), što ilustrira glomaznost postupka i jednadžbi koje bi pri tom dobili. Zbog toga ćemo, analogno kao u slučaju zaobilaženja prepreke, uzeti aproksimaciju u obliku

$$\frac{\partial G_S}{\partial q_i^2} = K_S H^*(q_i^1, q_i^2, q_i^3, q_i^4), \qquad \frac{\partial G_S}{\partial q_i^1} = 0,$$

$$\frac{\partial G_S}{\partial q_i} = K_S H\ (q_i^1, q_i, q_i, q_i), \qquad \frac{\partial G_S}{\partial q_i^3} = 0.$$

Gornji izrazi govore nam da u slučaju međusobnog presjeka ruku manipulatora dva robota, kuteve prvog i drugog robota q_1 i q_3 ne mjenjamo, dok duljine ruku oba robota, odnosno q_2 i q_4, smanjujemo konstantnom stopom smanjivanja, izraženom pozitivnim koeficijentom K_s. Ovim smo, dakle, problem sveli samo na geometrijsko utvrđivanje međusobnog presjeka ruku manipulatora.

Problem koji smo razmatrali je identičan problemu u prethodnom podpoglavlju s tom razlikom da smo za vremenski interval uzeli $t_f = 2.0\,s$ zbog toga što vrijeme od $t_f = 1.7\,s$ nije bilo dovoljno da zadovolji, pored prethodna tri ograničenja, još i uvjet međusobnog izbjegavanja sudara ruku dva robota.

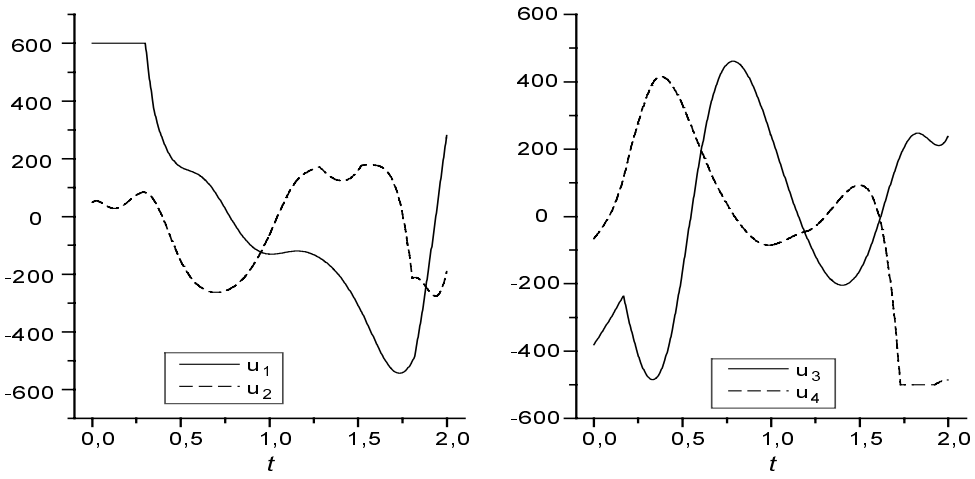
Ukupna funkcija cilja jednaka je $J = G_V + G_B + G_E + G_S$. Koristili smo gradijentnu metodu s konstantnim koeficijentom konvergencije. Vrijednosti konstanti koje smo koristili su

$$N = 1000, \qquad M = 160000, \qquad \eta = 3000,$$

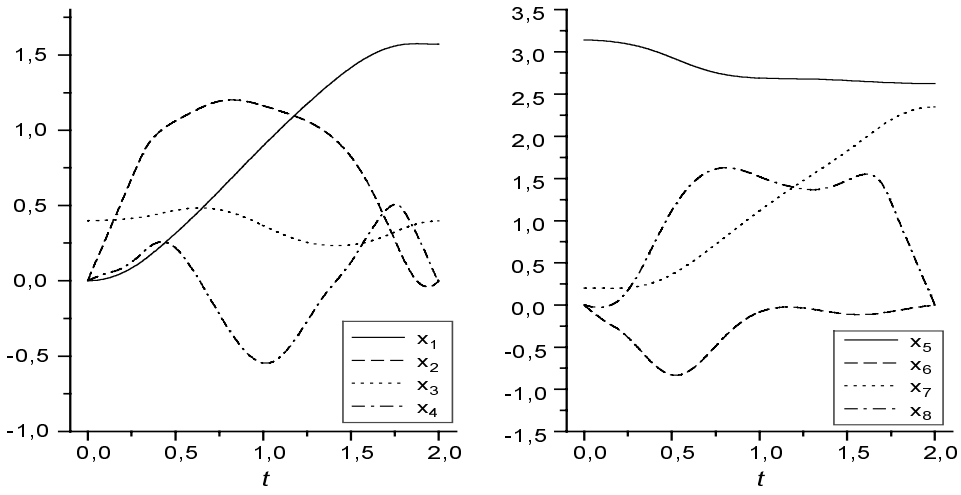
$$K_B = 100, \qquad K_V = 0.01, \qquad K_E = 1000, \qquad K_E = 500.$$

Vidimo da je dodavanje novog uvjeta bitno usporilo brzinu konvergencije algoritma. Slike 7.22 i 7.23 prikazuju vremensku ovisnost upravljačkih funkcija, odnosno varijabli stanja, dok slika 7.24 prikazuje vremensku promjenu međusobne udaljenosti prihvatnica dva robota i odstupanje te vrijednosti od definirane udaljenosti među hvatištima tereta koji se prenosi. Vidimo da postoje male oscilacije oko definirane međusobne udaljenosti koje se mogu kompenzirati pomičnim hvatištima tereta ili elastičnošću prihvatnice manipulatora. Slika 7.25 ilustrira brzinu konvergencije ukupne funkcije cilja.

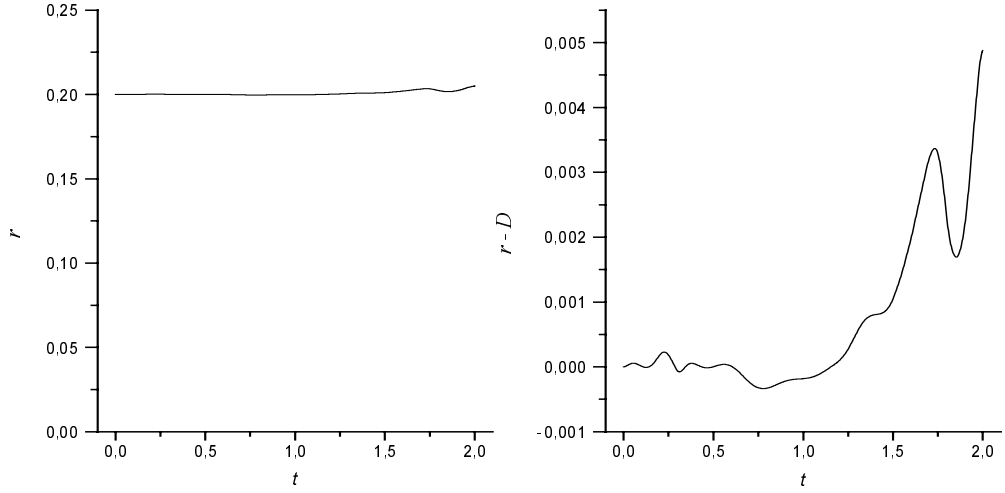
Slika 7.26 prikazuje trajektorije oba robota u koordinatnom sustavu prvog robota. Također, prikazani su položaji tereta u različitim vremenskim trenucima tokom prijenosa, a na slici 7.27 prikazana je rotacija tereta tokom prijenosa u nepomičnom koordinatnom sustavu. Vidimo da tokom prijenosa iz početnog u krajnji položaj dolazi do rotacije tereta u jednom smjeru za kut od $\pi / 2 rad$, a nakon toga dolazi do rotacije tereta u suprotnom smjeru za kut od πrad. Sve skupa, teret je prebrisao kut od $3\pi / 2 rad$, kao u primjeru iz prethodnog podpoglavlja, s tom razlikom što nije došlo do rotacije u jednom smjeru za kritičnih $3\pi / 4 rad$, kada dolazi do sudara ruku manipulatora dva robota.



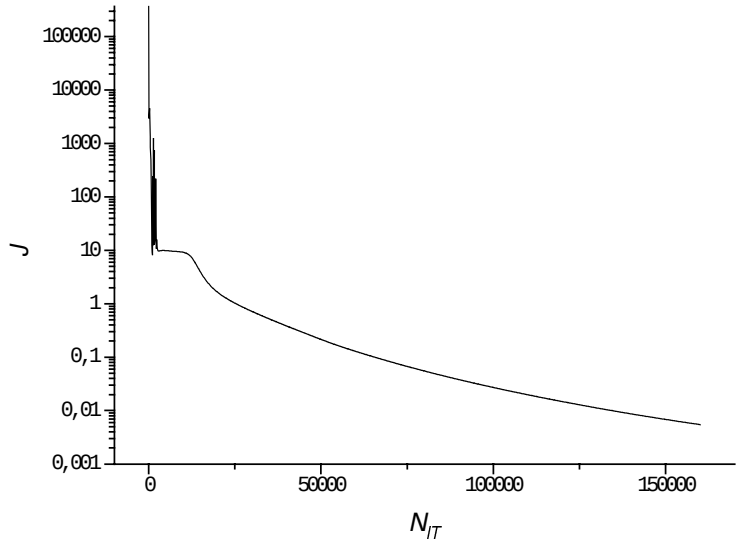
Slika 7.22 *Vremenska ovisnost upravljačkih funkcija 1. i 2. robota.*



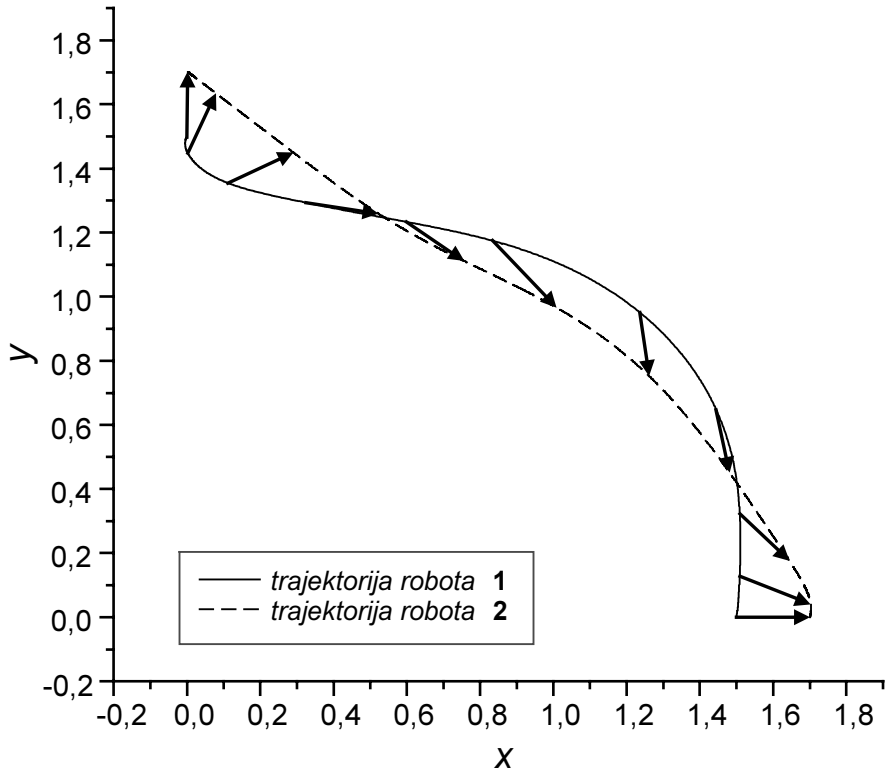
Slika 7.23 *Vremenska ovisnost varijabli stanja 1. i 2. robota.*



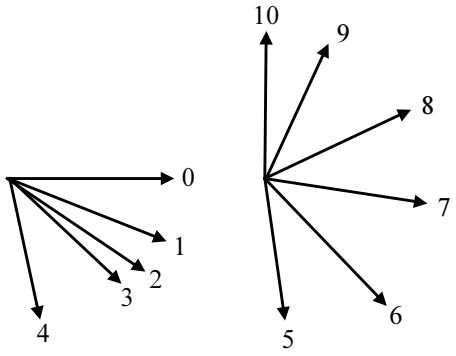
Slika 7.24 *Udaljenost između ruku manipulatora dva robota* r*, te razlika između te udaljenosti i razmaka između hvatišta krutog tereta,* D.



Slika 7.25 *Ovisnost ukupne funkcije cilja,* J *, o broju iteracija,* N_{IT} *.*



Slika 7.26 *Trajektorije dva robota s prikazom položaja tereta u različitim vremenskim trenucima.*



Slika 7.27 *Prikaz rotacije tereta tokom prijenosa u odnosu na nepomični koordinatni sustav.*

Ukupan broj računskih operacija koji nam je potreban da bi dobili zadovoljavajuću točnost rješenja, bitno je veći nego u prethodnim primjerima. Razlog za to je veći broja iteracija s jedne strane (zbog dodatnog uvjeta izbjegavanja sudara) i veći red sustava ($n = 8$) s druge strane. Broj računskih operacija unutar jedne iteracije gradijentnog algoritma proporcionalan je kvadratu reda sustava (broj elemenata Jacobijeve matrice $\mathbf{X}(j)$). Zbog toga je mogućnost realizacije BPTT algoritma primjenom neuronskih mreža vrlo bitna u slučaju sustava velikih dimenzija zbog mogućnosti iskorištavanja svojstva paralelizma, čime bi se bitno smanjio broj računskih operacija (serijskih) po jednoj iteraciji gradijentnog algoritma.

8. Optimalno upravljanje s povratnom vezom primjenom BPTT algoritma

Do sada smo razmatrali problem optimalnog upravljanja kod unaprijednog vođenja, gdje je problem bio u eksplicitnom određivanju funkcija upravljanja koje zadovoljavaju zadani kriterij optimalnosti. U ovom poglavlju razmatrat ćemo problem optimalnog upravljanja s povratnom vezom, gdje će problem biti u određivanju konstantnih koeficijenata pojačanja u petlji povratne veze koji zadovoljavaju zadani kriterij optimalnosti. Formulacija tog problema vrlo je slična formulaciji problema učenja kod dinamičkih neuronskih mreža gdje se trebaju odrediti konstantni koeficijenti (težinski koeficijenti) dinamičkog sustava koji minimiziraju kriterij optimalnosti, obično u obliku kvadrata pogreške, tokom zadanog vremenskog intervala.

Prve primjene BPTT algoritma na dinamičke sustave s povratnom vezom nalazimo u [23], [24], gdje se razmatra problem optimalnog upravljanja bez ograničenja nelinearnim sustavima drugog reda. Ovdje ćemo izvesti numerički algoritam za problem optimalnog upravljanja nelinearnim multivarijabilnim sustavima, s ograničenjima po upravljačkim varijablama.

8.1 Formulacija problema

Dinamiku sustava koji razmatramo možemo prikazati u obliku

$$\dot{x}_j(t) = \phi_j\left(x_1(t), x_2(t), ..., x_n(t), u_1(t), u_2(t), ..., u_m(t)\right). \tag{8.1}$$

gdje je $j = 1, 2, ..., n$. Zadani su početni i rubni uvjeti

$$x_j(t_0) = x_0^j, \qquad x_j(t_f) = x_f^j, \qquad j = 1, 2, ..., n, \tag{8.2}$$

te ograničenja upravljačkih varijabli

$$\left|u_j(t)\right| \le u_g^j, \qquad j = 1, 2, ..., m, \tag{8.3}$$

Upravljačke varijable povezujemo s varijablama stanja linearnom povratnom vezom oblika

$$u_j(t) = w_{j0} + \sum_{k=1}^{n} w_{jk} x_k(t), \qquad j = 1, 2, ..., m, \tag{8.4}$$

ili u matričnom obliku

$$\mathbf{u}(t) = \mathbf{w}_0 + \mathbf{W} \cdot \mathbf{x}(t), \tag{8.5}$$

gdje matrični elementi w_{jk} imaju ulogu koeficijenata pojačanja, a elementi vektora $\mathbf{w}_0$, w_{j0}, imaju ulogu ulaznih, vremenski konstantnih, upravljačkih parametara.

Problem se svodi na određivanje težinskih koeficijenata w_{jk} i w_{j0} koji minimiziraju kriterij optimalnosti

$$J = \int_{t_0}^{t_f} F(x_1(t), x_2(t), \ldots, x_n(t), u_1(t), u_2(t), \ldots, u_m(t))\, dt, \tag{8.6}$$

te zadovoljavaju rubne uvjete (8.2) i ograničenja (8.3).

Da bi osigurali zadovoljenje ograničenja (8.3), linearnu vezu upravljačkih varijabli i varijabli stanja oblika (8.4) zamjenit ćemo nelinearnom vezom oblika

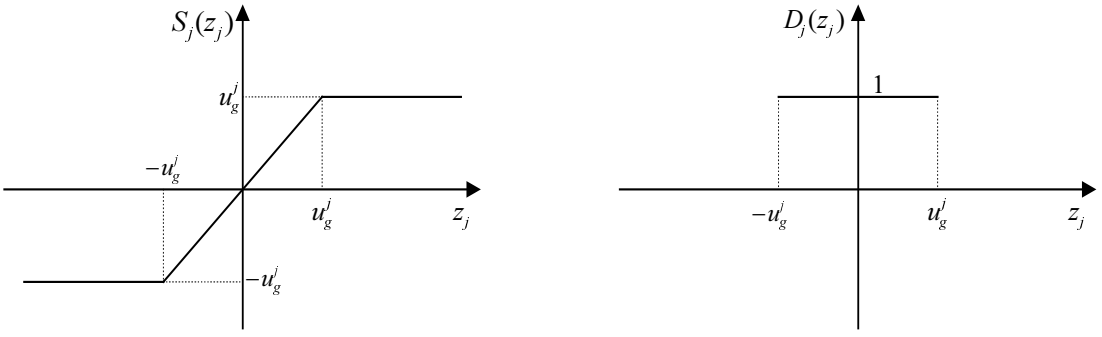
$$u_j(t) = S_j(z_j), \tag{8.7}$$

gdje je

$$z_j(t) = w_{j0} + \sum_{k=1}^{n} w_{jk} x_k(t), \tag{8.8}$$

$$S_j(z_j) = \begin{cases} u_g^j \ ; & z_j > u_g^j \\ z_j \ ; & -u_g^j \le z_j \le u_g^j \\ -u_g^j ; & z_j < -u_g^j \end{cases}. \tag{8.9}$$

Iz oblika funkcije $S_j(z_j)$ slijedi da će izraz (8.7) biti jednak izrazu (8.4) ako su zadovoljena ograničenja (8.3). Ako ograničenja nisu zadovoljena tada će funkcija $u_j(t)$ biti jednaka jednoj od graničnih vrijednosti, $u_j(t) \equiv \pm u_g^j$.



Slika 8.1 *Funkcija* $S_j(z_j)$ *i njena derivacija* $D_j(z_j)$.

8.2 Izvođenje rekurzivnih relacija za koeficijente pojačanja

Diskretizirani oblik ukupne funkcije cilja s kaznenom funkcijom za rubne uvjete je

$$J(\mathbf{W}) = T\sum_{i=0}^{N-1} F\left(\mathbf{x}_i, \mathbf{u}_i\right) + G_B\left(\mathbf{x}_N\right), \tag{8.10}$$

dok jednadžba (8.1) poprima oblik

$$x_{i+1}^j = f^j\left(\mathbf{x}_i, \mathbf{u}_i\right), \qquad j = 1,2,\ldots,n, \tag{8.11}$$

a (8.6) i (8.7) postaju

$$u_i^j = S_j\left(z_i^j\right), \qquad z_i^j = w_{j0} + \sum_{k=1}^{n} w_{jk} x_i^k, \qquad j = 1,2,\ldots,m, \tag{8.12}$$

gdje je $i = 0,1,\ldots,N$.

Pošto funkcija (8.10) implicitno ovisi o w_{jk} i w_{j0} preko (8.7), za minimizaciju gornje funkcije po navedenim koeficijentima možemo koristiti gradijentni algoritam

$$w_{pq}^{(l+1)} = w_{pq}^{(l)} - \eta \frac{\partial J}{\partial w_{pq}^{(l)}}, \tag{8.13}$$

gdje je $p = 1,2,\ldots,m$, $q = 0,1,\ldots,n$.

Gradijent funkcije cilja je

$$\frac{\partial J}{\partial w_{pq}} = T\sum_{i=0}^{N-1}\left(\sum_{r=1}^{n} \frac{\partial F(\mathbf{x}_i, \mathbf{u}_i)}{\partial x_i^r}\frac{\partial x_i^r}{\partial w_{pq}} + \sum_{r=1}^{m} \frac{\partial F(\mathbf{x}_i, \mathbf{u}_i)}{\partial u_i^r}\frac{\partial u_i^r}{\partial w_{pq}}\right) + \sum_{r=1}^{n} \frac{\partial G_B(\mathbf{x}_N)}{\partial x_N^r}\frac{\partial x_N^r}{\partial w_{pq}}.$$

Nadalje, na osnovu (8.11) imamo

$$\frac{\partial x_i^r}{\partial w_{pq}} = \sum_{j=1}^{n} \frac{\partial f^r(\mathbf{x}_{i-1}, \mathbf{u}_{i-1})}{\partial x_{i-1}^j}\frac{\partial x_{i-1}^j}{\partial w_{pq}} + \sum_{j=1}^{m} \frac{\partial f^r(\mathbf{x}_{i-1}, \mathbf{u}_{i-1})}{\partial u_{i-1}^j}\frac{\partial u_{i-1}^j}{\partial w_{pq}}, \tag{8.14}$$

dok na osnovu (8.12) slijedi

$$\frac{\partial u_i^r}{\partial w_{pq}} = D_r\left(z_i^r\right)\frac{\partial z_i^r}{\partial w_{pq}} = D_r\left(z_i^r\right)\left[\frac{\partial w_{r0}}{\partial w_{pq}} + \sum_{j=1}^{n}\left(\frac{\partial w_{rj}}{\partial w_{pq}} x_i^j + w_{rj}\frac{\partial x_i^j}{\partial w_{pq}}\right)\right], \tag{8.15}$$

gdje je $D_r\left(z_i^r\right)$ derivacija funkcije $S_r\left(z_i^r\right)$

$$D_r\left(z_i^r\right) = \begin{cases} 0; & z_i^r > u_g^j \\ 1; & -u_g^j \le z_i^r \le u_g^j \\ 0; & z_i^r < -u_g^j \end{cases}, \tag{8.16}$$

Nadalje, imamo

$$\frac{\partial u_i^r}{\partial w_{pq}} = D_r\left(z_i^r\right)\left[\delta_{rp}\delta_{q0} + \sum_{j=1}^{n}\left(\delta_{rp}\delta_{qj}x_i^j + w_{rj}\frac{\partial x_i^j}{\partial w_{pq}}\right)\right], \tag{8.17}$$

gdje je

$$\delta_{rp} = \begin{cases} 1;\ r = p \\ 0;\ r \neq p \end{cases}. \tag{8.18}$$

Na kraju, imamo

$$\frac{\partial u_i^r}{\partial w_{pq}} = D_r\left(z_i^r\right)\left(\delta_{rp}\delta_{q0} + \xi_q\left(x_i^q\right)\delta_{rp} + \sum_{j=1}^{n} w_{rj}\frac{\partial x_i^j}{\partial w_{pq}}\right), \tag{8.19}$$

gdje je

$$\xi_q(x_i^q) = \begin{cases} 0 & ;\ q = 0 \\ x_i^q & ;\ q > 0 \end{cases}. \tag{8.20}$$

Uvrstimo li gornji izraz u (8.14), dobivamo

$$\begin{aligned}\frac{\partial x_i^r}{\partial w_{pq}} = &\sum_{j=1}^{n}\frac{\partial f^r(\mathbf{x}_{i-1},\mathbf{u}_{i-1})}{\partial x_{i-1}^j}\frac{\partial x_{i-1}^j}{\partial w_{pq}} + \sum_{j=1}^{m}\frac{\partial f^r(\mathbf{x}_{i-1},\mathbf{u}_{i-1})}{\partial u_{i-1}^j}D_j\left(z_{i-1}^j\right)\sum_{k=1}^{n} w_{jk}\frac{\partial x_{i-1}^k}{\partial w_{pq}} + \\ &+ \frac{\partial f^r(\mathbf{x}_{i-1},\mathbf{u}_{i-1})}{\partial u_{i-1}^p}\left(\xi_q(x_{i-1}^q) + \delta_{q0}\right)D_p\left(z_{i-1}^p\right)\end{aligned} \tag{8.21}$$

Izrazi (8.19) i (8.21) predstavljaju rekurzivne relacije potrebne za izračunavanje gradijenta funkcije cilja.

Još preostaje određivanje početnih uvjeta za navedene rekurzivne relacije. S obzirom da početni uvjeti vektora stanja ne ovise o koeficijentima w_{jk}, slijedi da je

$$\frac{\partial x_0^r}{\partial w_{pq}} = 0, \tag{8.22}$$

gdje je $r = 1,2,\ldots,n$, $p = 1,2,\ldots,m$, $q = 0,1,\ldots,n$.

Na osnovu (8.22) i (8.19) dobivamo drugi skup početnih uvjeta

$$\frac{\partial u_0^r}{\partial w_{pq}} = \left(\delta_{rp}\delta_{q0} + \xi(x_0^q)\delta_{rp}\right)D_r\left(z_0^r\right), \tag{8.23}$$

gdje je $r = 1,2,\ldots,m$, $p = 1,2,\ldots,m$, $q = 0,1,\ldots,n$.

8.3 Primjena na upravljanje robotom s dva stupnja slobode gibanja

U podpoglavlju 7.2 razmatrali smo vremenski optimalno upravljanje robotom s dva stupnja slobode gibanja uz zadana ograničenja upravljačkih varijabli. Pošto se radilo o unaprijednom vođenju, problem je bio pronalaženje vremenske ovisnosti upravljačkih funkcija koje zadovoljavaju kriterij optimalnosti uz zadana ograničenja. Ovdje ćemo razmatrati isti problem s tom razlikom što imamo povratnu vezu po varijablama stanja (8.7)-(8.9), tako da se problem svodi na pronalaženje deset konstantnih koeficijenata koji zadovoljavaju zadani kriterij optimalnosti i ograničenja upravljačkih varijabli.

S obzirom da zbog oblika povratne veze, gdje nam funkcija zasićenja $S_j(z_i^j)$ po definiciji garantira zadovoljavanje ograničenja upravljačkih varijabli, suma kaznenih funkcija J_G jednaka je kaznenoj funkciji za rubne uvjete

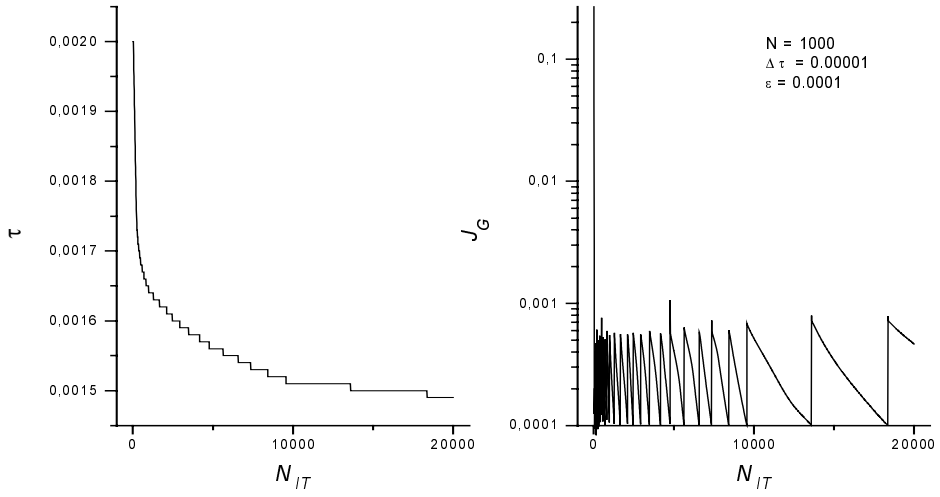
$$J_G = G_B \,. \tag{8.24}$$

Koristili smo metodu konjugiranog gradijenta. Vrijednosti konstanti su

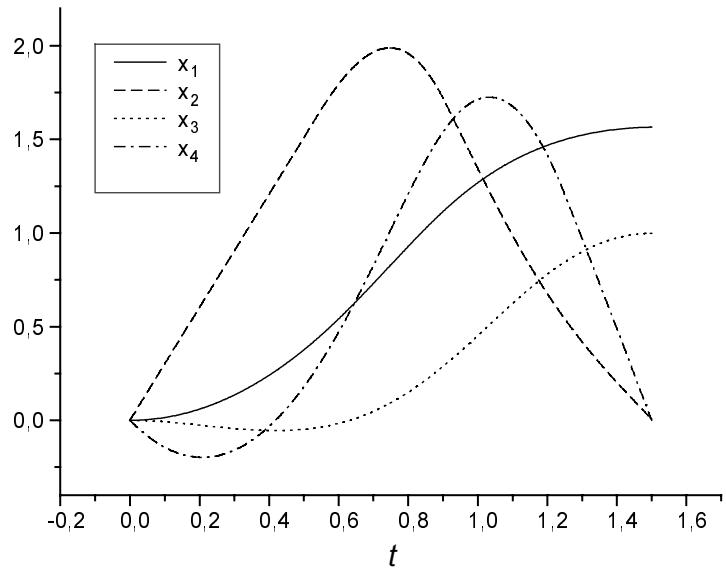
$$N = 1000\,, \qquad M = 20000\,, \qquad K_B = 1\,, \qquad \varepsilon \equiv J_{G\min} = 0.0001\,.$$

Zadana točnost, koja je definirana parametrima $\varepsilon \equiv J_{G\min}$ i K , približno je ista točnosti definiranoj tim parametrima u podpoglavlju 7.2.

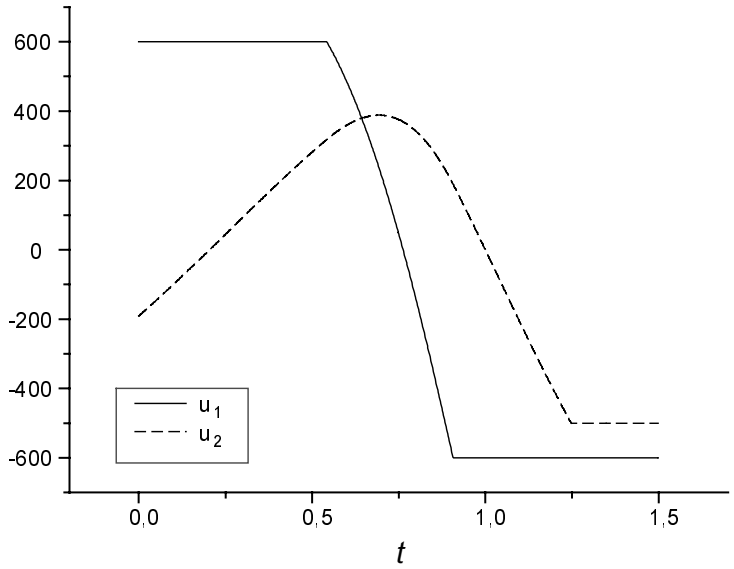
Kao što vidimo na slici 8.2, uz zadanu točnost $\varepsilon \equiv J_{G\min} = 0.0001$ dobili smo, na kraju navedenog broja iteracija, minimalno vrijeme $t_{\min} = 1.5\,s$ (isto kao u podpoglavlju 7.2), odnosno $\tau_{\min} = t_{\min} / N$. Zatim smo to vrijeme stavili konstantnim i pomoću dobivenog algoritma, uz $M = 10000$, dobili rezultate prikazane na slikama 8.3 do 8.6.



Slika 8.2 *Ovisnost τ i J_G o broju iteracija.*

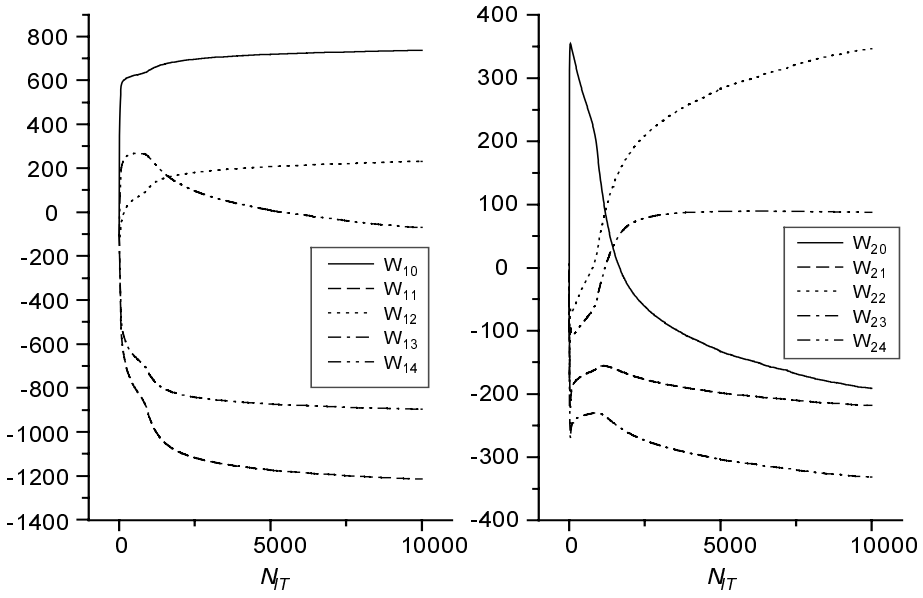


Slika 8.3 *Vremenska ovisnost varijabli stanja.*



Slika 8.4 *Vremenska ovisnost funkcija upravljanja.*

Očigledna je velika sličnost vremenskih ovisnosti varijabli upravljanja a posebno varijabli stanja, na slikama 8.3 i 8.4, sa onima iz podpoglavlja 7.2 (slike 7.3 i 7.4).



Slika 8.5 *Ovisnost vrijednosti težinskih koeficijenata o broju iteracija.*

Vrijednosti težinskih koeficijenata nakon $M = 10000$ iteracija su

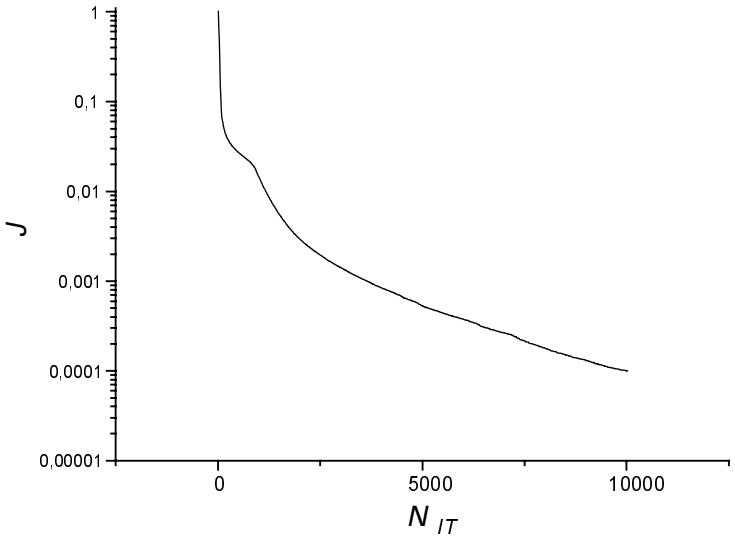
$w_{10} = 738.8\ N \cdot m$, $w_{20} = -191.2\ N$,

$w_{11} = -1211.4\ N \cdot m \cdot rad^{-1}$, $w_{21} = -218.1\ N \cdot rad^{-1}$,

$w_{12} = 233.5\ N \cdot m \cdot rad^{-1} \cdot s$, $w_{22} = 346.6\ N \cdot rad^{-1} \cdot s$

$w_{13} = -894.1\ N$, $w_{23} = -331.5\ N \cdot m^{-1}$,

$w_{14} = -67.4\ N \cdot s$, $w_{24} = 88.0\ N \cdot m^{-1} \cdot s$.



Slika 8.6 *Ovisnost* J *o broju iteracija.*

8.4 Mogućnosti regulacije u realnom vremenu

U prethodnom podpoglavlju razmatrali smo problem optimalnog upravljanja s povratnom vezom. Ovdje ćemo razmotriti problem regulacije, odnosno slučaj kada je barem jedan početni uvjet različit od nule dok su rubni uvjeti (uvjeti u konačnom vremenu) jednaki nuli (problem otklanjanja regulacijskog odstupanja).

Razmotrimo primjer iz prethodnog podpoglavlja u slučaju da zamjenimo početne i rubne uvjete, odnosno

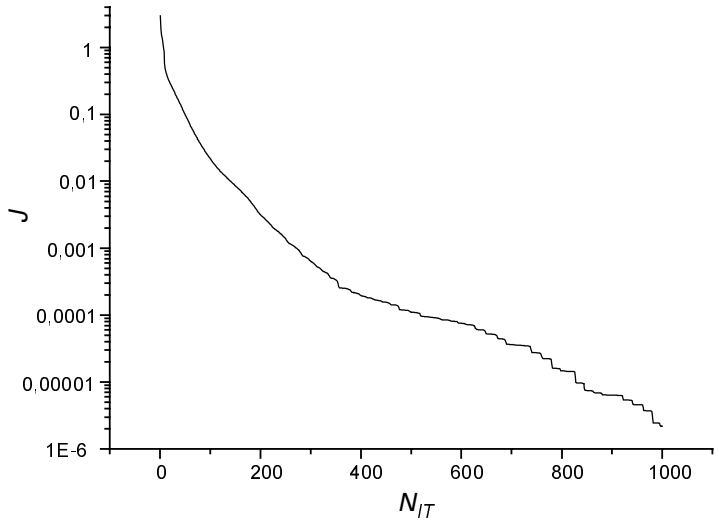
$$x_1(0) = q_1(0) = \frac{\pi}{2} rad, \qquad x_1(t_f) = q_1(t_f) = 0\,rad,$$

$$x\ (0) = \dot{q}_1(0) = 0\,rad \cdot s^{-1}, \qquad x_2(t_f) = \dot{q}_1(t_f) = 0\,rad \cdot s^{-1},$$

$$x_3(0) = q_2(0) = 1\,m, \qquad x_3(t_f) = q\ (t_f) = 0\,m,$$

$$x_4(0) = \dot{q}\ (0) = 0\,m \cdot s^{-1}, \qquad x_4(t_f) = \dot{q}_2(t_f) = 0\,m \cdot s^{-1}.$$

Na slici 8.7 prikazana je brzina konvergencije algoritma. Usporedimo li je sa slikom 8.6, primjećujemo da je brzina konvergencije bitno veća u slučaju regulacije, odnosno, potrebno je puno manje iteracija za istu razinu točnosti rješenja. Razlog za takvu razliku u brzini konvergencije leži u tome što u slučaju regulacije varijable stanja imaju eksponencijalno opadajuće ponašanje, tako da malo odstupanje u parametrima ima za posljedicu vrlo malo odstupanje varijabli stanja u konačnom vremenskom trenutku, t_f.



Slika 8.7 *Ovisnost* J *o broju iteracija.*

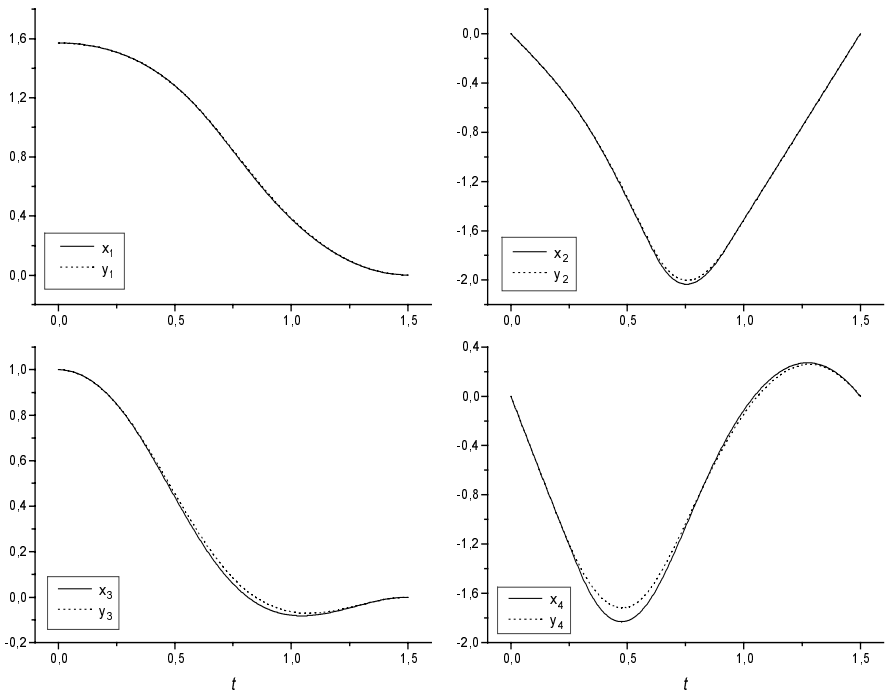
Na osnovu tih svojstava, može se razmatrati mogućnost upravljanja u realnom vremenu koje bi se zasnivalo na izračunavanju gradijenta tokom jednog perioda diskretizacije (perioda sempliranja signala). To bi značilo da u svakom periodu sempliranja signala gradijentnim algoritmom dobivamo vrijednosti koeficijenata pojačanja, odnosno imamo vremenski promjenjive koeficijente pojačanja.

Na slici 8.8 prikazana je usporedba rješenja dobivenog u realnom vremenu sa rješenjem s konstantnim koeficijentima pojačanja dobivenim nakon $N = 1000$ iteracija. Vidimo da su odstupanja relativno mala i to na sredini vremenskog intervala, dok odstupanje na kraju vremenskog intervala iščezava (slika 8.9), što opravdava upotrebljivost upravljanja u realnom vremenu. Na slici 8.10 prikazana je vremenska ovisnost koeficijenata pojačanja.

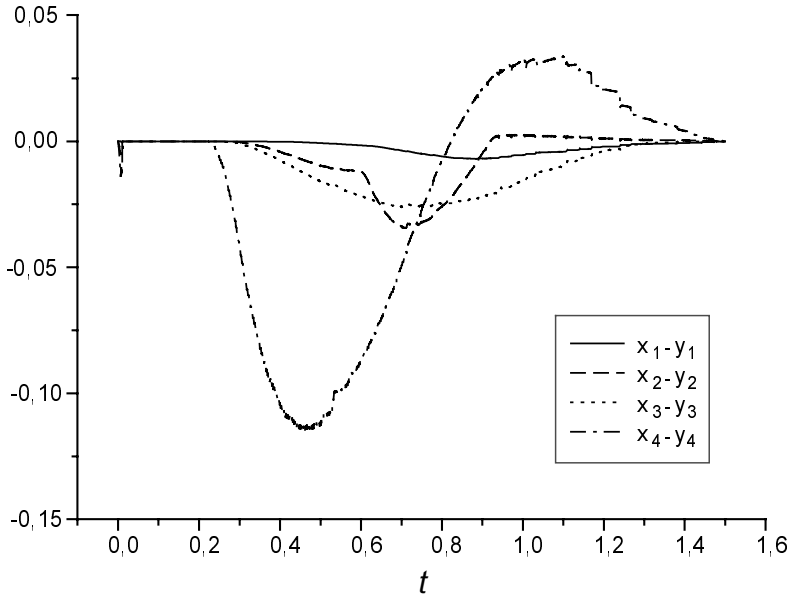
Ostaje jedino pitanje izvršavanja potrebnog broja računskih operacija tokom zadanog perioda diskretizacije. Na osnovi dobivenog algoritma (8.13)-(8.15), može se vidjeti da broj operacija, N_{OP}, ima približnu ovisnost oblika

$$N_{OP} \approx N\left(n^3 m + 2n^2 m^2\right). \tag{8.25}$$

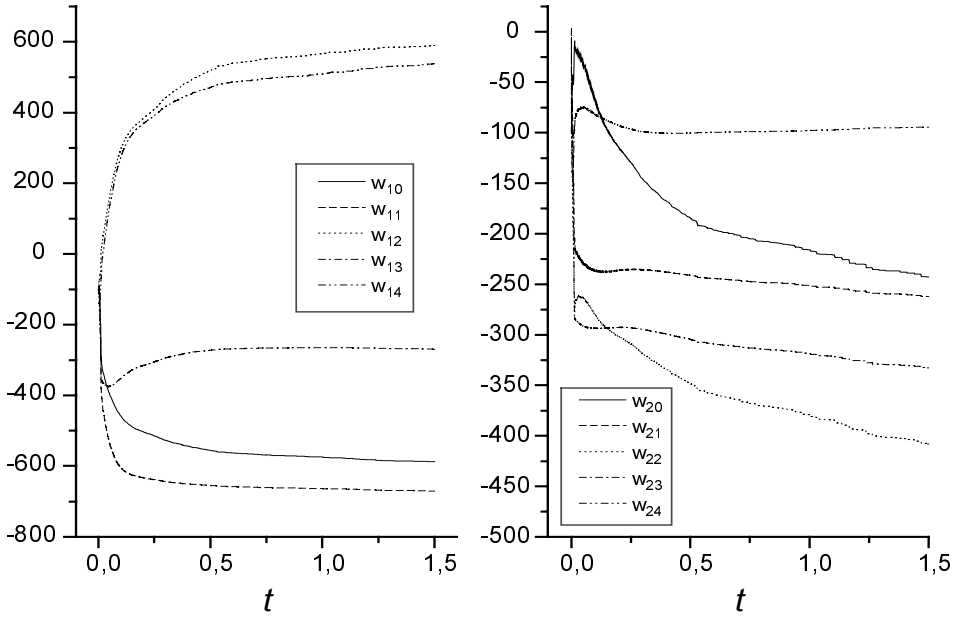
U našem slučaju, za $N = 1000$, $n = 4$, $m = 2$, imamo $N_{OP} \approx 256000$ operacija tokom perioda sempliranja koji je jednak $T = 0.0015\,s$. Na osnovu tih podataka proizlazi da je potrebno vrijeme za izvršavanje jedne instrukcije $T_I \approx 5.8 \cdot 10^{-9}\,s$. Takve računarske zahtijeve zadovoljava procesor TMS320C67x. To je procesor u aritmetici pomičnog zareza sa 1GFLOPS-a na taktu od 167 MHz, čije vrijeme za izvršavanje jedne instrukcije iznosi $T_I \approx 10^{-9}\,s$.



Slika 8.8 *Usporedba rješenja dobivenog u realnom vremenu ($y_1,\ldots,y_4$) sa rješenjem s konstantnim koeficijentima pojačanja ($x_1,\ldots,x_4$).*



Slika 8.9 *Razlika rješenja dobivenog s konstantnim koeficijentima pojačanja ($x_1,...,x_4$) i rješenja u realnom vremenu ($y_1,...,y_4$).*



Slika 8.10 *Vremenska ovisnost koeficijenata pojačanja.*

Pošto smo razmatrali problem regulacije u konačnom vremenskom intervalu s funkcijom cilja (8.6), problem stabilnosti nije posebno razmatran s obzirom da zadovoljenje rubnog uvjeta (funkcija $G_B(\mathbf{x}_N)$ teži nuli) automatski garantira stabilnost u zadanom vremenskom intervalu. Međutim, to ne znači da je dobiveno rješenje stabilno u proizvoljnom vremenskom intervalu (Lyapunovljeva stabilnost). To znači da bi se navedeni algoritam mogao koristiti (bez prethodnog ispitivanja stabilnosti) u procesima kod kojih bi se, nakon zadanog vremenskog intervala, znao daljnji tok upravljanja. Primjer za takav proces je prenošenje tereta robotom iz početnog stanja u konačno uz uvjet minimuma vremena (primjer u podpoglavlju 8.3) i ponovno vraćanje u početni položaj u najkraćem vremenu (primjer u podpoglavlju 8.4). Drugim riječima, nakon zadanog vremenskog intervala, težinska matrica za problem u podpoglavlju 8.3 zamjenila bi se težinskom matricom za problem u podpoglavlju 8.4.

Također, nismo razmatrali ponašanje sustava u prisutnosti stohastičkih poremećaja (šum mjerenja), jer bi to premašilo predviđeni obim ovog rada u kojemu je osnovni naglasak na tretiranju problema ograničenja varijabli stanja i upravljanja.

9. Zaključak

Na osnovi rezultata prikazanih u prethodnim poglavljima mogu se uočiti određene dobre strane kao i određene slabosti dobivenog algoritma. Na primjerima problema s poznatim analitičkim rješenjem dobili smo gotovo savršeno poklapanje numeričkih rezultata sa analitičkim za relativno mali broj iteracija. Korištenjem metode konjugiranog gradijenta brzina konvergencije još se više povećava. U situacijama gdje dobivene numeričke rezultate ne možemo usporediti s analitičkim, kao u slučaju optimalnog upravljanja robotom, mjera točnosti rješenja je ukupna suma kaznenih funkcija. Što je ta suma manja, to je rješenje točnije.

Usporedimo li metodu s konstantnim koeficijentom konvergencije sa metodom konjugiranog gradijenta, dolazimo do slijedećih zaključaka. Metoda s konstantnim koeficijentom konvergencije jednostavnija je za upotrebu; nije potrebna dodatna jednodimenzionalna minimizacija funkcije cilja. Međutim, zbog činjenice da stabilnost algoritma ovisi o koeficijentu konvergencije (odnosno, postoji gornja granica vrijednosti tog koeficijenta iznad koje algoritam postaje nestabilan) imamo i ograničenje brzine konvergencije algoritma. Metodom konjugiranog gradijenta rješavamo pitanje stabilnosti (koeficijent konvergencije određujemo u svakom koraku iteriranja) i bitno ubrzavamo konvergenciju.

S druge strane, kao što smo vidjeli na primjeru zaobilaženja prepreke i primjeru kooperativnog rada dva robota, zbog složenosti izračunavanja parcijalnih derivacija kaznenih funkcija pribjegli smo, može se reći, grubim aproksimacijama tih derivacija što je onemogućilo upotrebu metode konjugiranog gradijenta, ali je zato metoda s konstantnim koeficijentom konvergencije dala dobre rezultate. Razlog za neefikasnost metode konjugiranog gradijenta, u ovom slučaju, leži u raskoraku između aproksimativne vrijednosti gradijenta funkcije cilja i točne vrijednosti funkcije cilja. Sa stanovišta metode s konstantnim koeficijentom konvergencije, aproksimativna vrijednost gradijenta znači određeno skretanje sa smjera najbržeg spusta, odnosno, određeno usporavanje konvergencije.

Vidjeli smo da ograničenja po varijablama stanja mogu biti prilično složena, a derivacije kaznenih funkcija za ta ograničenja izuzetno komplicirane. Stoga se čini prilično značajna činjenica da se grubom aproksimacijom tih derivacija mogu dobiti dobri rezultati, pa makar uz cijenu smanjenja brzine konvergencije. Imajući u vidu specifičnu geometrijsku interpretaciju gradijentnog algoritma, mogli bi zaključiti da je gradijentni algoritam s konstantnim koeficijentom konvergencije možda jedini algoritam koji je u stanju rješavati problem optimalnog upravljanja s navedenim tipom aproksimacija.

Primjenom BPTT (*backpropagation-through-time*) algoritma na problem optimalnog upravljanja s konstantnim konačnim vremenom (odnosno, vremenskim

intervalom upravljanja) dobili smo vrlo elegantan oblik algoritma u kojem se zahtijeva poznavanje samo prvih derivacija odgovarajućih veličina. Primjenom istog principa na vremenski optimalno upravljanje dobili smo vrlo glomazan algoritam, nepraktičan za upotrebu, s derivacijama drugog reda. Čak i kad bi dobiveni algoritam implementirali, ostalo bi otvoreno pitanje stabilnosti algoritma. Osnovni problem u realizaciji vremenski optimalnog upravljanja koje je bazirano na simultanom iteriranju varijabli upravljanja i vremenskog intervala, leži u činjenici da za dani vremenski interval nemamo dovoljno točno rješenje varijabli upravljanja. Taj problem smo rješili tako da smo vrijednost vremenskog intervala držali konstantnom sve dok ne bi postigli odgovarajuću točnost rješenja, definiranu sumom kaznenih funkcija. Taj pristup garantira stabilnost algoritma i, kao što smo vidjeli, daje vrlo dobre rezultate.

Pošto je dokazivanje egzistencije rješenja problema optimalnog upravljanja s ograničenjima vrlo težak problem u općenitom slučaju, tretiranje ograničenja primjenom kaznenih funkcija vrlo je korisno i kao indikacija rješivosti problema. Ako kaznena funkcija ne konvergira prema nuli (za stabilan algoritam s konstantnim koeficijentom konvergencije) to je siguran znak da problem nema rješenje u okviru zadanih ograničenja. Upravo to svojstvo kaznenih funkcija je iskorišteno u algoritmu za vremenski optimalno upravljanje.

Brzina konvergencije algoritma ne ovisi bitno o redu sustava, koliko o broju ograničenja, odnosno kaznenih funkcija, po varijablama stanja i upravljanja. Na osnovi dobivenih rezultata možemo zaključiti da je optimalna primjena algoritma na nelinearne dinamičke sustave sa zadanim rubnim uvjetima, ograničenjima upravljačkih varijabli i jednim tipom ograničenja po varijablama stanja. Već dva tipa ograničenja po varijablama stanja, kao što smo vidjeli (međusobno izbjegavanje sudara i održavanje konstantne udaljenosti ruku manipulatora), mogu značajno smanjiti brzinu konvergencije algoritma.

Analogija BPTT algoritma za optimalno upravljanje sa strukturom dinamičke neuronske mreže, odnosno unaprijedne (statične) neuronske mreže s *N-1* skrivenih slojeva, korisna je zbog mogućnosti korištenja svojstava paralelizma neuronskih mreža, što bi omogućilo (primjenom paralelnog procesiranja) bitno smanjenje ovisnosti brzine računanja o redu dinamičkog sustava.

Na kraju još nekoliko riječi o mogućim pravcima daljnjeg rada i istraživanja. Interesantno bi bilo ispitati razne moguće modifikacije dobivenog algoritma. Problem modifikacije funkcije cilja u obliku integrala možemo transformirati u problem minimizacije dodatne varijable stanja u konačnom vremenu. Zatim, umjesto da kaznene funkcije dodamo funkciji cilja, moguće je uvesti dodatne varijable stanja s rubnim uvjetima jednakim nuli. Međusobnom usporedbom raznih modifikacija utvrdile bi se verzije algoritma koje daju najbolje rezultate pod određenim okolnostima. Također, interesantno bi bilo ispitati razne heurističke gradijentne algoritme s promjenjivim koeficijentom konvergencije u cilju dobivanja boljih konvergencijskih svojstava.

Algoritam za optimalno upravljanje dinamičkim sustavima s povratnom vezom i s ograničenjima upravljačkih varijabli (sinteza regulatora) izveden je za slučaj statičkog regulatora. Nadalje, trebalo bi provesti analizu stabilnosti i ispitati osjetljivost na stohastičke poremećaje (šum mjerenja). Daljnje poboljšanje ostvarilo bi se primjenom BPTT algoritma za slučaj dinamičkog regulatora. Također, valjalo bi ispitati mogućnosti povratne veze u slučaju ograničenja po varijablama stanja.

Interesantna bi bila primjena dobivenog algoritma na problem diferencijalnih igara. U formulaciji problema diferencijalne igre imamo dinamički sustav s dva vektora upravljanja, gdje jedan vektor upravljanja nastoji maksimizirati određenu funkciju cilja, dok drugi vektor upravljanja nastoji minimizirati istu funkciju cilja. Tako formuliran problem bitno je složeniji od standardnog problema optimalnog upravljanja. Proučavanje diferencijalnih igara za sada se svodi ili na opće teoreme ili na rješavanje jednostavnih modela. Teško se nalazi optimalno rješenje koje bi slijedilo iz općih principa teorije optimalnog upravljanja. Proširenje primjene gradijentnog algoritma na problem diferencijalne igre bilo bi relativno jednostavno. Za one upravljačke varijable koje minimiziraju funkciju cilja stavio bi se negativan predznak ispred gradijenta, dok bi se za one upravljačke varijable koje maksimiziraju funkciju cilja stavio pozitivan predznak. Mogućnost rješenja diferencijalne igre vrlo je korisna, jer bi razne neodređenosti u sustavu koje ne možemo modelirati, poput stohastičkih smetnji, promjenjivih parametara sustava, nemodelirane dinamike, mogli tretirati kao dodatni vektor stanja koji nastoji maksimizirati definiranu funkciju cilja (ako je u pitanju problem minimizacije). Sve što bi trebali učiniti je da te dodatne varijable stanja dodamo na odgovarajuća mjesta u modelu dinamike sustava i definiramo interval unutar kojeg se te varijable mogu naći. S druge strane, u ekonomskim primjenama optimalnog upravljanja rješavanje problema u obliku diferencijalne igre ima izuzetan značaj.

Problem nemodelirane dinamike mogli bi također riješiti primjenom neuronske mreže koja bi na osnovu odgovarajućeg broja različitih ulazno-izlaznih podataka dovoljno dobro “naučila” dinamiku sustava kojim upravljamo.

10. Prilozi

10.1 Popis korištenih oznaka

$\mathbf{u}(t)$ – vektor upravljanja
$\mathbf{x}(t)$ – vektor stanja sustava
t – vrijeme
t_0 – početni vremenski trenutak
t_f – konačni vremenski trenutak
J – kriterij optimalnosti, funkcija cilja
H – Hamiltonian
T – konstantni period diskretizacije
$\tau^{(k)}$ – varijabilni period diskretizacije
η – konstantni koeficijent konvergencije
N – broj diskretnih vremenskih podintervala
M – broj iteracija gradijentnog algoritma
n – dimenzija vektora stanja
m – dimenzija vektora upravljanja
$\mathbf{X}(j)$ – Jacobijeva matrica po vektoru stanja
$\mathbf{U}(j)$ – Jacobijeva matrica po vektoru upravljanja
G_B – kaznena funkcija za ograničenja tipa rubnih uvjeta
G_V – kaznena funkcija za ograničenja tipa nejednakosti
G_E – kaznena funkcija za ograničenja tipa jednakosti
K_B – koeficijent kaznene funkcije G_B
K_V – koeficijent kaznene funkcije G_V
K_E – koeficijent kaznene funkcije G_E
J_G – ukupna suma kaznenih funkcija
$\mathbf{q}(t)$, $\dot{\mathbf{q}}(t)$ – vektori generaliziranih koordinata i brzina robota
$\mathbf{W}$ – matrica težinskih koeficijenata, matrica koeficijenata pojačanja
S_j – funkcija zasićenja, aktivacijska funkcija
D_j – derivacija funkcije zasićenja
$\mathbf{I}$ – jedinična matrica
$\mathbf{0}$ – nul-matrica

Ostale oznake, kao i navedene oznake s drugim značenjem, objašnjene su u tekstu.

10.2 Program za BPTT algoritam

```
// OptimalControl.c
// BPTT algoritam za rjesavanje problema vremenski optimalnog upravljanja
// dinamikom robota s dva stupnja slobode (rotaciona i translaciona).
// Koristi se metoda konjugiranog gradijenta.

#include <stdio.h>
#include <math.h>
#include "Nrutil.h"                 // procedure za alokaciju memorije
#include "Dat1.h"                   // procedure za rad s datotekama
#include "matrix.h"                 // procedure za matricne operacije
#include "Dim1Min.h"                // procedure za 1-D  minimizaciju

#define Ndat 10                     // broj datoteka
#define N 1000                      // broj vremenskih intervala
#define Noff 2000                   // broj iteracija za OFF-line
#define NMOD 100                    // broj iteracija za resetiranje
#define NDIM (N-1)                  // broj varijabli fun. cilja
#define Nn 4                        // broj komponenti vektora stanja
#define Nm 2                        // broj komponenti vektora upravljanja
#define T ((tf-to)/N)               // period diskretizacije
#define to 0.0                      // pocetno vrijeme
#define tf 1.5                      // konacno vrijeme
#define x01 0.0                     // pocetna vrijednost x1(0)=x01
#define x02 0.0                     // pocetna vrijednost x2(0)=x02
#define x03 0.0                     // pocetna vrijednost x3(0)=x03
#define x04 0.0                     // pocetna vrijednost x4(0)=x04
#define x1f 1.571                   // konacna vrijednost x1(tf)=x1f
#define x2f 0.0                     // konacna vrijednost x2(tf)=x2f
#define x3f 1.0                     // konacna vrijednost x3(tf)=x3f
#define x4f 0.0                     // konacna vrijednost x4(tf)=x4f
#define U1g 600.0                   // granicna vrijednost upravljacke varijable  'u1'
#define U2g 500.0                   // granicna vrijednost upravljacke varijable  'u2'

#define TOL 1.0e-4                  // tolerancija rjesenja
#define uo 100.0                    // pocetna vrijednost u(0)=uo
#define uf 200.0                    // konacna vrijednost u(tf)=uf
#define S 800.0                     // koeficijent uz kaznenu funkciju (g=0)
#define K 0.04                      // koeficijent uz kaznenu funkciju (g>=0)
#define eps 0.01                    //minimalna vrijednost funkcije cilja

#define J12 193.0                   // ukupni moment inercije
#define M1 5.0                      // masa 'M'
#define M2 97.0                     // masa 'm'
#define Aa 1.1                      // duljina ruke
#define A1 (J12+M1*Aa*Aa)           // konstanta dinamickog modela
#define A2 (2*M1*Aa)                // konstanta dinamickog modela
#define A3 (M1+M2)                  // konstanta dinamickog modela


double **Uu, **Ju, dt;
```

```
void Vrijeme(double *t, double h, int n)        // vrijeme
{
     int i;

     for(i=0;i<=n;i++) t[i]=i*h;
}

void Init_u0(double **u, double *t, int n)                    // pocetna vrijednost  u
{
     int i, j;

     for(j=1;j<=Nm;j++)
          for(i=0;i<=n;i++)
               u[j][i]=(uf-uo)*((double)(i)/n)+uo;
}

void func(double *f, double **x, double **u, int i)            //  x' = f(x, u)
{
     double x1, x2, x3, x4, u1, u2, naz;

     x1 = x[1][i]; x2 = x[2][i];x3 = x[3][i]; x4 = x[4][i];
     u1 = u[1][i]; u2 = u[2][i];
     naz = (A1 + A2*x3 + A3*x3*x3);

     f[1] = x2;
     f[2] = (-2*A3*x2*x3*x4 - A2*x2*x4 + u1)/naz;
     f[3] = x4;
     f[4] = (A3*x3*x2*x2 - A2*x2*x2/2 + u2)/A3;
}

void vec_x(double **x, double **u, double h, int n)          // Eulerov metod
{
     int i, j;
     double *f;

     f=dvector(1, Nn);
     x[1][0]=x01; x[2][0]=x02; x[3][0]=x03; x[4][0]=x04;
     for(i=0;i<=n;i++){
          for(j=1;j<=Nn;j++){
               func(f, x, u, i);
               x[j][i+1]=x[j][i]+h*f[j];
          }
     }
     free_dvector(f, 1);
}

double sn4(double x)
{
     double r1;

     if (x >= 0.0) r1=0.0;
     if (x < 0.0) r1=1.0;
     return(r1);
}
```

```
double Fu(double u1, double ug)              // parcijalna derivacija F po u
{
    double w1, w2, ret;

    w1 = u1 + ug;
    w2 = ug - u1;

    ret = 2*K*(w1*sn4(w1) - w2*sn4(w2));
    return(ret);
}

void JaccobX(double **X, double **x, double **u, double h, int i)        // Jacobijan po x
{
    double x1, x2, x3, x4, u1, u2, naz;

    x1 = x[1][i]; x2 = x[2][i];x3 = x[3][i]; x4 = x[4][i];
    u1 = u[1][i]; u2 = u[2][i];
    naz = (A1 + A2*x3 + A3*x3*x3);


    X[1][1] = 1.0; X[1][2] = 0.0; X[1][3] = 0.0; X[1][4] = 0.0;

    X[2][1] = h; X[2][2] = 1.0 + h*(-2*A3*x3*x4 - A2*x4)/naz; X[2][3] = 0.0;
    X[2][4] = h*(2*A3*x3*x2 - A2*x2*x2)/A3;

    X[3][1] = 0.0;
    X[3][2] = h*(-2*A3*x2*x4/naz - (A2+2*A3*x3)*(-2*A3*x2*x3*x4 - A2*x2*x4 + u1)/naz/naz);
    X[3][3] = 1.0; X[3][4] = h*x2*x2;

    X[4][1] = 0.0; X[4][2] = h*(-2*A3*x2*x3 - A2*x2)/naz; X[4][3] = h; X[4][4] = 1.0;
}

void JaccobU(double **U, double **x, double **u, double h, int i)        // Jacobijan po  u
{
    double x3, naz;

    x3 = x[3][i];
    naz = (A1 + A2*x3 + A3*x3*x3);

    U[1][1] = 0.0; U[1][2] = h/naz; U[1][3] = 0.0; U[1][4] = 0.0;
    U[2][1] = 0.0; U[2][2] = 0.0; U[2][3] = 0.0; U[2][4] = h/A3;
}

void vecFx(double *vFx, double   **x, double **u, int j)                 // parcijalna derivacija F po x
{
    vFx[1] = 0.0;
    vFx[2] = 0.0;
    vFx[3] = 0.0;
    vFx[4] = 0.0;
}

void vecFu(double *vFu, double   **x, double **u, int j)                 // parcijalna derivacija F po u
{
    vFu[1] = Fu(u[1][j], U1g);
    vFu[2] = Fu(u[2][j], U2g);
}
```

```
void vecGx(double *Gx, double **x)                    // parcijalna derivacija G_B po x
{
    Gx[1] = 2*S*(x[1][N]-x1f);
    Gx[2] = 2*S*(x[2][N]-x2f);
    Gx[3] = 2*S*(x[3][N]-x3f);
    Gx[4] = 2*S*(x[4][N]-x4f);
}

double J(double **u)                                  // ukpna funkcija cilja
{
    int i;
    double r1, r2=0.0, ret, w1, w2, w3, w4, **x;
    double s1, s2, s3, s4;

    x=dmatrix(1, Nn, 0, N);

    vec_x(x, u, dt, NDIM);
    s1 = (x[1][N]-x1f)*(x[1][N]-x1f);
    s2 = (x[2][N]-x2f)*(x[2][N]-x2f);
    s3 = (x[3][N]-x3f)*(x[3][N]-x3f);
    s4 = (x[4][N]-x4f)*(x[4][N]-x4f);
    r1 = S*(s1+s2+s3+s4);
     for(i=0;i<=NDIM;i++){
        w1 = (u[1][i] + U1g); w2 = (U1g - u[1][i]);
        w3 = (u[2][i] + U2g); w4 = (U2g - u[2][i]);
        r2 = r2 + dt*K*(w1*w1*sn4(w1)+w2*w2*sn4(w2)+w3*w3*sn4(w3)+w4*w4*sn4(w4));
    }
    ret = r1 + r2;
    free_dmatrix(x, 1, Nn, 0, N);
    return(ret);
}

double Jfun(double z)                                 // metoda najbrzeg spusta
{
    double r1, **y;
    int i, j;

    y=dmatrix(1, Nm, 0, NDIM);

    for(i=1;i<=Nm;i++)
        for(j=0;j<=NDIM;j++)
            y[i][j] = Uu[i][j] - z*Ju[i][j];
    r1 = J(y);
    free_dmatrix(y, 1, Nm, 0, NDIM);
    return(r1);
}

double J0(double **x)                                 // kaznena funkcija za rubne uvjete
{
    double r1, s1, s2, s3, s4;

    s1 = (x[1][N]-x1f)*(x[1][N]-x1f);
    s2 = (x[2][N]-x2f)*(x[2][N]-x2f);
    s3 = (x[3][N]-x3f)*(x[3][N]-x3f);
    s4 = (x[4][N]-x4f)*(x[4][N]-x4f);
    r1 = S*(s1+s2+s3+s4);
    return(r1);
}
```

```
double J1(double **u, int n)                    // kaznena funkcija za ograničenja vektora u
{
     int i;
     double r2=0.0, w1, w2, w3, w4;

     for(i=0;i<=n;i++){
          w1 = (u[1][i] + U1g); w2 = (U1g - u[1][i]);
          w3 = (u[2][i] + U2g); w4 = (U2g - u[2][i]);
          r2 = r2 + K*(w1*w1*sn4(w1)+w2*w2*sn4(w2)+w3*w3*sn4(w3)+w4*w4*sn4(w4));
     }
     return(r2);
}

void Grad(double **Ju, double **x, double **u, double h)        // gradijent funkcije cilja
{
     double **X, **U, *P, *P1, *Fq, *Fw, *S1, *Gx, *Gu, *J0;
     double **Xt, **Ut, **Dj, **D1, **Xq, **Xqt;
     int j;

     X = dmatrix(1,Nn,1,Nn);
     U = dmatrix(1,Nm,1,Nn);
     P = dvector(1,Nn);
     S1 = dvector(1,Nm);
     P1 = dvector(1,Nn);
     Fq = dvector(1,Nn);
     Fw = dvector(1,Nm);
     Gx = dvector(1,Nn);
     J0 = dvector(1,Nm);
     Gu = dvector(1,Nm);
     Xt = dmatrix(1,Nn,1,Nn);
     Ut = dmatrix(1,Nn,1,Nm);
     Dj = dmatrix(1,Nn,1,Nn);
     D1 = dmatrix(1,Nn,1,Nn);
     Xq = dmatrix(1,Nn,1,Nm);
     Xqt = dmatrix(1,Nm,1,Nn);

     IdentMat(Dj, Nn);
     NullVector(P, Nn);

     vecGx(Gx, x);
     vecFu(Fw, x, u, N-1);
     JaccobU(U, x, u, h, N-1);
     MultVec(Gu, U, Gx, Nn, Nm);
     Storage(Ju, Fw, Gu, h, N-1);

     for(j=N-2;j>=0;j--){

          JaccobX(X, x, u, h, j+1);
          JaccobU(U, x, u, h, j);
          vecFx(Fq, x, u, j+1);
          vecFu(Fw, x, u, j);

          MultVec(P1, X, P, Nn, Nn);
          SummVec(P, Fq, P1, Nn);
          MultVec(S1, U, P, Nn, Nm);
          SummVec(J0, Fw, S1, Nm);

          TranMat(Xt, X, Nn, Nn);
          TranMat(Ut, U, Nm, Nn);
```

```
        MultMat(D1, Dj, Xt, Nn, Nn, Nn);
        MoveMat(Dj, D1, Nn, Nn);
        MultMat(Xq, Dj, Ut, Nn, Nn, Nm);
        TranMat(Xqt, Xq, Nn, Nm);
        MultVec(Gu, Xqt, Gx, Nn, Nm);

        Storage(Ju, J0, Gu, h, j);
    }
    free_dmatrix(X,1,Nn,1,Nn);
    free_dmatrix(U,1,Nm,1,Nn);
    free_dvector(P, 1);
    free_dvector(P1, 1);
    free_dvector(S1, 1);
    free_dvector(Fq, 1);
    free_dvector(Fw, 1);
    free_dvector(Gx, 1);
    free_dvector(Gu, 1);
    free_dvector(J0, 1);
    free_dmatrix(Xt,1,Nn,1,Nn);
    free_dmatrix(Ut,1,Nn,1,Nm);
    free_dmatrix(Dj,1,Nn,1,Nn);
    free_dmatrix(D1,1,Nn,1,Nn);
    free_dmatrix(Xq,1,Nn,1,Nm);
    free_dmatrix(Xqt,1,Nm,1,Nn);
}

void Opt_Upr(double **u, double *t)          // gradijentni algoritam + vremenski optimalno upravljanje
{
    double **x, **D, fnc, qmin, bren, br, naz, alf, fc0, fc1, fc, Dt;
    int i, j, k;

    x=dmatrix(1, Nn, 0, N);
    Ju=dmatrix(1, Nm, 0, N-1);
    D=dmatrix(1, Nm, 0, N-1);

    dt=T;

    for(k=0;k<=Noff;k++){

        if (k>0) naz = AbsVel(Ju, NDIM);
        vec_x(x, u, dt, N-1);
        Grad(Ju, x, u, dt);

        fc0 = J0(x);
        fc1 = J1(u, NDIM);
        fc = fc0 + fc1;
        Dt = (fc - eps);

        if (Dt<0.0) dt = dt - 0.00001;

        naz = AbsVel(Ju, NDIM);
        if (k%NMOD==0){
            for(j=1;j<=Nm;j++)
                for(i=1;i<=NDIM;i++)
                    D[j][i] = -Ju[j][i];
        }
        bren=Brent1(Jfun, TOL, &qmin);
        for(j=1;j<=Nm;j++)
            for(i=0;i<=N-1;i++)
```

```
                u[j][i]=u[j][i]-qmin*Ju[j][i];
        br = AbsVel(Ju, NDIM);
        if (k==0) alf = 0.0;
        else alf = br/naz;

        for(j=1;j<=Nm;j++)
            for(i=1;i<=NDIM;i++)
                D[j][i] = -Ju[j][i] + alf*D[j][i];
        naz = br;

        if ((k%100)==0){
            fnc = J(u);
            printf("%d J=%lf fc=%lf tf=%lf\n", k, fnc, fc, N*dt);
            puni_toc((double)(k), fc0, 7);
            puni_toc((double)(k), fc1, 8);
            puni_toc((double)(k), fc, 9);
            puni_toc((double)(k), fnc, 10);
        }
    }
    free_dmatrix(x, 1, Nn, 0, N);
    free_dmatrix(Ju, 1, Nm, 0, N-1);
    free_dmatrix(D, 1, Nm, 0, N-1);
}

void main(void)                    // glavni program
{
    double *t, **x;

    t=dvector(0, N);
    x=dmatrix(1, Nn, 0, N);
    Uu=dmatrix(1, Nm, 0, NDIM);

    datot(Ndat);                   // otvaranje datoteka
    Vrijeme(t, T, N);              // t - vrijeme
    Init_u0(Uu, t, N-1);           // Uu - pocetni vektor upravljanja
    Opt_Upr(Uu, t);                // Uu - optimalni vektor upravljanja
    Vrijeme(t, dt, N);
    vec_x(x, Uu, dt, N-1);         // x - optimalni vektor stanja

    puni_dat1(t, x[1], 0, N, 1);
    puni_dat1(t, x[2], 0, N, 2);
    puni_dat1(t, x[3], 0, N, 3);
    puni_dat1(t, x[4], 0, N, 4);
    puni_dat1(t, Uu[1], 0, N-1, 5);
    puni_dat1(t, Uu[2], 0, N-1, 6);

    free_dvector(t, 0);
    free_dmatrix(x, 1, Nn, 0, N);
    free_dmatrix(Uu, 1, Nm, 0, NDIM);

    datzt(Ndat);                   // zatvaranje datoteka
}
```

```
//  matrix.h

void Greska(double **z, double **x, double **y, int n)            // razlika 2 vektora
{
     int i, j;

     for(j=1;j<=Nn;j++)
          for(i=0;i<=n;i++)
               z[j][i]=x[j][i]-y[j][i];
}

void MultMat(double **z, double **x, double **y, int n, int m, int l)          // množenje matrica
// Z=X*Y       X-n*m, Y-m*l,  Z-n*l
{
 int i, j, k;

 for(i=1; i<=n; i++){
     for(j=1; j<=l; j++){
       z[i][j] = 0.0;
       for(k=1; k<=m; k++)
          z[i][j] += x[i][k]*y[k][j];
     }
 }
}

void MultVec(double *z, double **x, double *y, int n, int m)          // množenje matrice s vektorom
//  z=X*y    z-m*1   X-m*n   y-n*1
{
 int i, j;

 for(i=1; i<=m; i++){
     z[i] = 0.0;
     for(j=1; j<=n; j++)
       z[i] += x[i][j]*y[j];
 }
}

void SummVec(double *z, double *x, double *y, int n)            // zbroj dva vektora
{
 int i;

 for(i=1; i<=n; i++) z[i] = x[i] + y[i];
}

void TranMat(double **y, double **x, int n, int m)                  // transponiranje matrice
//  x-n*m   y-m*n
{
 int i, j;

 for(i=1; i<=n; i++)
     for(j=1; j<=m; j++)
       y[j][i] = x[i][j];
}
```

```
void MoveMat(double **y, double **x, int n, int m)              // prebacivanje jedne matrice u drugu
{
 int i, j;

 for(i=1; i<=n; i++)
     for(j=1; j<=m; j++)
      y[i][j] = x[i][j];
}

void NullVector(double *x, int n)                  // nul vektor
{
     int i;

   for(i=1; i<=n; i++)       x[i] = 0.0;
}

void IdentMat(double **x, int n)                   // jedinicna matrica
{
     int i, j;

   for(i=1; i<=n; i++)
          for(j=1; j<=n; j++){
               if (i==j) x[i][j] = 1.0;
               else x[i][j] = 0.0;
          }
}

void Storage(double **Ju, double *J0, double *Gu, double h, int j)
{
     int i;

     for(i=1; i<=Nm; i++)
          Ju[i][j] = h*J0[i] + Gu[i];
}

double AbsVel(double **d, int n)             // skalarni produkt dva vektora
{
     int i, j;
     double r=0.0;

     for(i=1; i<=Nm; i++)
          for(j=1; j<=n; j++)
               r = r + d[i][j]*d[i][j];

     return(r);
}
```

// **Dim1Min.h**

```
#define GOLD 1.618034
#define GLIMIT 100.0
#define TINY 1.0e-20
#define MAX(a,b) ((a) > (b) ? (a) : (b))
#define SIGN(a,b) ((b) > 0.0 ? fabs(a) : -fabs(a))
#define SHFT(a,b,c,d) (a)=(b);(b)=(c);(c)=(d);

void mnbrak(double *ax, double *bx, double *cx, double *fa, double *fb, double *fc, double
(*func)(double))
{
    double ulim,u,r,q,fu,dum;

    *fa=(*func)(*ax);
    *fb=(*func)(*bx);
    if (*fb > *fa) {
        SHFT(dum,*ax,*bx,dum)
        SHFT(dum,*fb,*fa,dum)
    }
    *cx=(*bx)+GOLD*(*bx-*ax);
    *fc=(*func)(*cx);
    while (*fb > *fc) {
        r=(*bx-*ax)*(*fb-*fc);
        q=(*bx-*cx)*(*fb-*fa);
        u=(*bx)-((*bx-*cx)*q-(*bx-*ax)*r)/
            (2.0*SIGN(MAX(fabs(q-r),TINY),q-r));
        ulim=(*bx)+GLIMIT*(*cx-*bx);
        if ((*bx-u)*(u-*cx) > 0.0) {
            fu=(*func)(u);
            if (fu < *fc) {
                *ax=(*bx);
                *bx=u;
                *fa=(*fb);
                *fb=fu;
                return;
            } else if (fu > *fb) {
                *cx=u;
                *fc=fu;
                return;
            }
            u=(*cx)+GOLD*(*cx-*bx);
            fu=(*func)(u);
        } else if ((*cx-u)*(u-ulim) > 0.0) {
            fu=(*func)(u);
            if (fu < *fc) {
                SHFT(*bx,*cx,u,*cx+GOLD*(*cx-*bx))
                SHFT(*fb,*fc,fu,(*func)(u))
            }
        } else if ((u-ulim)*(ulim-*cx) >= 0.0) {
            u=ulim;
            fu=(*func)(u);
        } else {
            u=(*cx)+GOLD*(*cx-*bx);
            fu=(*func)(u);
        }
        SHFT(*ax,*bx,*cx,u)
        SHFT(*fa,*fb,*fc,fu)
    }
```

```
}

#undef GOLD
#undef GLIMIT
#undef TINY
#undef MAX

#define ITMAX 100
#define CGOLD 0.3819660
#define ZEPS 1.0e-10

double brent(double ax, double bx, double cx, double (*f)(double), double tol, double *xmin)
{
	int iter;
	double a,b,d,etemp,fu,fv,fw,fx,p,q,r,tol1,tol2,u,v,w,x,xm;
	double e=0.0;
	void nrerror();

	a=((ax < cx) ? ax : cx);
	b=((ax > cx) ? ax : cx);
	x=w=v=bx;
	fw=fv=fx=(*f)(x);
	for (iter=1;iter<=ITMAX;iter++) {
//		printf("iter=%d\n",iter);
		xm=0.5*(a+b);
		tol2=2.0*(tol1=tol*fabs(x)+ZEPS);
		if (fabs(x-xm) <= (tol2-0.5*(b-a))) {
			*xmin=x;
			return fx;
		}
		if (fabs(e) > tol1) {
			r=(x-w)*(fx-fv);
			q=(x-v)*(fx-fw);
			p=(x-v)*q-(x-w)*r;
			q=2.0*(q-r);
			if (q > 0.0) p = -p;
			q=fabs(q);
			etemp=e;
			e=d;
			if (fabs(p) >= fabs(0.5*q*etemp) || p <= q*(a-x) || p >= q*(b-x))
				d=CGOLD*(e=(x >= xm ? a-x : b-x));
			else {
				d=p/q;
				u=x+d;
				if (u-a < tol2 || b-u < tol2)
					d=SIGN(tol1,xm-x);
			}
		} else {
			d=CGOLD*(e=(x >= xm ? a-x : b-x));
		}
		u=(fabs(d) >= tol1 ? x+d : x+SIGN(tol1,d));
		fu=(*f)(u);
		if (fu <= fx) {
			if (u >= x) a=x; else b=x;
			SHFT(v,w,x,u)
			SHFT(fv,fw,fx,fu)
		} else {
			if (u < x) a=u; else b=u;
			if (fu <= fw || w == x) {
```

```
                    v=w;
                    w=u;
                    fv=fw;
                    fw=fu;
                } else if (fu <= fv || v == x || v == w) {
                    v=u;
                    fv=fu;
                }
            }
        }
        nrerror("Too many iterations in BRENT");
        *xmin=x;
        return fx;
}

#undef ITMAX
#undef CGOLD
#undef ZEPS
#undef SIGN

double Brent1(double (*func)(double), double tol, double *xm)
{
        double ax,bx,cx,fa,fb,fc,br,xmin;

        ax=1.0; bx=2.0; cx=3.0;
        mnbrak(&ax,&bx,&cx,&fa,&fb,&fc,func);
        br=brent(ax,bx,cx,func,tol,&xmin);
        *xm=xmin;
        return(br);
}
```

// **Dat1.h**

```
//#include <stdlib.h>
//#include <string.h>

FILE *fp[28];

void datot(int N_dat)                    // otvaranje N_dat datoteka
{
    int ibr;
    char init_name[8], file_name[12], ibr_str[5];

    printf ("\nUnesite inicijalno ime datoteke: ");
    scanf ("%s", init_name);
    for (ibr=1; ibr<=N_dat; ibr++){
        strcpy (file_name, init_name);
        if (ibr < 10)
            sprintf (ibr_str, "0%d", ibr);
        else if (ibr < 100)
            sprintf (ibr_str, "%d", ibr);
        else
            sprintf (ibr_str, "%d", ibr);
        strcat (file_name, ibr_str);
        strcat (file_name, ".dat");
        fp[ibr] = fopen (file_name, "w");
    }
}

void datzt(int how)                      // zatvaranje N_dat datoteka
{
    int ibr;

    for (ibr=1; ibr<=how; ibr++){
        fclose(fp[ibr]);
        printf ("Datoteka %d je zatvorena\n", ibr);
    }
}

void puni_dat1(double *x, double *y, int n1, int n2, int j)        // ponjenje datoteke vektorima x[i], y[i]
{
    int i;

    for(i=n1;i<=n2;i++)
        fprintf (fp[j],"%10.6lf  %10.6lf\n", x[i], y[i]);
}

void puni_toc(double x, double y, int j)                           // ponjenje datoteke skalarima x, y
{
    fprintf (fp[j], "%12.10lf %12.10lf\n", x, y);
}
```

// **Nrutil1.h**

```
#include <malloc.h>
#include <stddef.h>
#include <stdlib.h>

void nrerror(char error_text[])
{
	void exit();

	fprintf(stderr,"Numerical Recipes run-time error...\n");
	fprintf(stderr,"%s\n",error_text);
	fprintf(stderr,"...now exiting to system...\n");
	exit(0);
}

double *dvector(int nl,int nh)
{
	double *v;

	v=(double *)malloc((unsigned) (nh-nl+1)*sizeof(double));
	if (!v) nrerror("allocation failure in dvector()");
	return v-nl;
}

double **dmatrix(int nrl,int nrh,int ncl,int nch)
{
	int i;
	double **m;

	m=(double **) malloc((unsigned) (nrh-nrl+1)*sizeof(double*));
	if (!m) nrerror("allocation failure 1 in dmatrix()");
	m -= nrl;

	for(i=nrl;i<=nrh;i++) {
		m[i]=(double *) malloc((unsigned) (nch-ncl+1)*sizeof(double));
		if (!m[i]) nrerror("allocation failure 2 in dmatrix()");
		m[i] -= ncl;
	}
	return m;
}

void free_dvector(double *v,int nl)
{
	free((char*) (v+nl));
}


void free_dmatrix(double **m, int nrl, int nrh, int ncl, int nch)
{
	int i;

	for(i=nrh;i>=nrl;i--) free((char*) (m[i]+ncl));
	free((char*) (m+nrl));
}
```

Literatura

[1] L. S. Pontryagin, V. G. Boltyanskii, R.V. Gamkrelidze, E. F. Mishchenko, *The Mathematical Theory of Optimal Processes*, John Wiley, New York, 1962.

[2] R. Bellman, *Dynamic Programming*, Princeton University Press, Princeton, 1957.

[3] A. P. Sage, C. C. White, *Optimum System Control*, Prentice-Hall, New Jersey, 1977.

[4] A. E. Bryson, Y. C. Ho, *Applied Optimal Control*, John Wiley, New York, 1969.

[5] P. N. V. Tu, *Introductory Optimization Dynamics*, Springer-Verlag, Berlin, 1991.

[6] D. Leonard, N. V. Long, *Optimal Control Theory and Static Optimization in Economics*, Cambridge University Press, Cambridge, 1992.

[7] M. Ruth, B. Hannon, *Modeling Dynamic Economic Systems*, Springer-Verlag, New York, 1997.

[8] M. D. Mesarovic, D. Macko, Y. Takahara, *Teorija hijerarhijskih sistema sa više nivoa*, Informator, Zagreb, 1972.

[9] B. Novaković, *Metode vođenja tehničkih sistema*, Školska knjiga, Zagreb, 1990.

[10] L. T. Fan, *The Continuous Maximum Principle*, John Wiley, New York, 1966.

[11] R. Bellman, *Adaptive Control Processes: A Guided Tour*, Princeton University Press, New Jersey, 1961.

[12] R. Bellman, S. Dreyfus, *Applied Dynamic Programming*, Princeton University Press, Princeton, 1962.

[13] P. J. Werbos, "Backpropagation through time: What it does and how to do it", *Proc. IEEE*, vol. 78, pp. 1550-1560, 1990.

[14] P. J. Werbos, "Maximaizing long-term gas industry profits in two minutes in Lotus using neural network methods", *IEEE Trans. Syst., Man, Cybern.*, Mar./Apr. 1989.

[15] P. J. Werbos, "Generalization of backpropagation with application to a recurrent gas market model", *Neural Networks*, pp. 339-356, 1988.

[16] D. E. Rumelhart and J. L. McClelland, *Parallel Distributed Processing,* Vol. 1., MA: The MIT Press, 1986.

[17] S. W. Piche, " Steepest Descent Algorithms for Neural Network Controllers and Filters", *IEEE Trans. on Neural Networks*, vol. 5, no. 2, pp. 198-212, 1994.

[18] Y. M. Park and K. Y. Lee, "An Optimal Tracking Neuro-Controller for Nonlinear Dynamic Systems", *IEEE Trans. on Neural Networks*, vol. 7, no. 5, pp. 1099-1110, 1996.

[19] M. S. Bazaraa, H. D. Sherali, C. M. Shetty, "Nonlinear Programming", John Wiley, 1993.

[20] S. Turk, L. Budin, *Analiza i projektiranje računalom*, Školska knjiga, Zagreb, 1989.

[22] E. S. Plumer, "Optimal Control of Terminal Processes Using Neural Networks", *IEEE Trans. on Neural Networks*, vol. 7, no. 2, pp. 408-418, 1996.

[23] K. S. Narendra, K. Parthasarathy, “Gradient Methods for the Optimization of Dynamical Systems Containing Neural Networks”, *IEEE Trans. on Neural Networks*, vol. 2, no. 2, pp. 252-262, 1991.

[24] K. S. Narendra, K. Parthasarathy, “Identification and Control of Dynamical Systems Using Neural Networks”, *IEEE Trans. on Neural Networks*, vol. 1, pp. 4-27, Mar 1990.

Sažetak

U radnji je prikazan izvod numerickog algoritma za optimalno upravljanje nelinearnim multivarijabilnim sustavima sa ogranicenjima varijabli stanja i upravljanja. Izvod algoritma zasnovan je na BPTT (*backpropagation-through-time*) algoritmu koji se primjenjuje kao algoritam ucenja za dinamicke neuronske mreže. Izveden je takoder algoritam za vremenski optimalno upravljanje koji je zasnovan na svojstvima kaznenih funkcija za ogranicenja varijabli stanja i upravljanja. Dobiveni algoritmi primijenjeni su na razne probleme optimalnog upravljanja robotom s dva stupnja slobode. Najprije je razmatran problem vremenski optimalnog upravljanja robotom s ogranicenjima varijabli upravljanja. Zatim je razmatran isti problem s dodatnim uvjetom; zaobilaženjem prepreke, odnosno ogranicenjima varijabli stanja. Nakon toga razmatra se problem optimalnog upravljanja kooperativnim radom dva robota. Problem se sastoji u zajednickom prenošenju krutog tereta s jednog mjesta na drugo uz ogranicenja varijabli upravljanja, te uz uvjet medusobnog izbjegavanja sudara i održavanja konstantne udaljenosti izmedu prihvatnica manipulatora. Na kraju je prikazan izvod algoritma za optimalno upravljanje s povratnom vezom za nelinearne multivarijabilne sustave s ogranicenjima varijabli upravljanja. Izvod algoritma takoder je zasnovan na principu BPTT algoritma.

Summary

“Optimal Control of Nonlinear Systems Using Neural Networks”

The master's thesis presents the derivation of the numerical algorithm for optimal control of nonlinear multivariable systems with limited state and control vectors. The algorithm derivation is based on the BPTT (backpropagation-through-time) algorithm which is used as a learning algorithm for dynamic neural networks. Presented is also the derivation of the algorithm for time optimal control which is based on the characteristics of penalty functions for constraints of state and control vectors. The derived algorithms are used for various problems concerning optimal robot control with two degrees of freedom. The thesis first deals with the problem of time optimal robot control with control vector constraints. It also considers the same problem with an additional condition - avoidance of obstacles, that is, constraint of state vectors. The thesis further deals with the problem of optimal control of cooperative work of two robots, that is, of joint transfer of solid material from one place to another including control vector constraint and conditions of mutual avoidance of clashes and maintaining of constant distance between the hands of the robot. Derivation of the optimal control algorithm with a feedback for nonlinear multivariable systems with control vector constraint is presented at the end. This derivation is also based on the BPTT algorithm.

Životopis

Josip Kasač rođen je 31.07.1969. godine u Vinkovcima, Republika Hrvatska. Srednju školu elektrotehničkog usmjerenja završio je u Vinkovcima. Maturirao je 1988. godine i iste godine upisuje se na Prirodoslovno matematički fakultet, smjer: inžinjer fizike, koji je počeo pohađati nakon odsluženog vojnog roka. Diplomirao je 1995. godine na temi "Solitoni u sredstvu s prigušenjem", te stekao zvanje diplomiranog inžinjera fizike. Iste godine upisuje se na sveučilišni poslijediplomski studij "Vođenje i upravljanje pokretnim objektima". Do sada je objavio jedan rad na međunarodnoj konferenciji ITI '98 u Puli.