NUMEERISIA MENETELMIÄ SINGULAARISEN STRATEGIAN RATKAISEMISEKSI POPULAATIOMALLEISSA

Sami Kallio

Pro gradu -tutkielma
Helmikuu 2020

MATEMATIIKAN JA TILASTOTIETEEN LAITOS
TURUN YLIOPISTO

TURUN YLIOPISTO
Matematiikan ja tilastotieteen laitos

KALLIO, SAMI: Numeerisia menetelmiä singulaarisen strategian ratkaisemiseksi populaatiomalleissa
Pro gradu -tutkielma, 45 s., 5 liites.
Sovellettu matematiikka
Helmikuu 2020

Tässä tutkielmassa käydään läpi erilaisia numeerisia menetelmiä singulaarisen strategian ratkaisemiseksi matemaattisissa populaatiomalleissa. Menetelmät on tarkoitettu käytettäväksi tilanteissa, joissa singulaarista strategiaa ei ole mahdollista ratkaista analyyttisesti.

Menetelmien soveltamista ja vertailua varten johdetaan esimerkkimallina diskreettiaikainen resurssi-kuluttajamalli. Ennen menetelmien läpikäymistä tutustutaan lyhyesti adaptiivisen dynamiikan teoriaan soveltaen sitä samalla resurssi-kuluttajamalliin. Tutkielma etenee lähtien idealtaan yksinkertaisimmasta menetelmästä, kelpoisuusgradienttimenetelmästä, päätyen lopulta toistettuun kvadraattiseen optimointiin. Suurin osa tarkastelluista menetelmistä ovat sakkofunktiomenetelmiä, jotka eroavat toisistaan moniulotteisen rajoitteettoman optimointitehtävän ratkaisualgoritmin valinnan osalta.

Jokaisen menetelmän toiminnan perustelemiseksi esitetään siihen kuuluva teoria. Lisäksi menetelmät esitetään myös algoritmisessa muodossa, josta niitä on helppo hyödyntää omiin tarpeisiin. Liiteosiossa vertaillaan kootusti algoritmien tehokkuutta esimerkkimallin singulaarisen strategian ratkaisemisessa. Tutkielman algoritmit ja kuvat on toteutettu käyttäen Mathematica-ohjelmistoa.

Asiasanat: adaptiivinen dynamiikka, singulaarinen strategia, matemaattinen optimointi.

Sisältö

1 Johdanto

Yksi adaptiivisen dynamiikan keskeisimmistä käsitteistä on singulaarinen strategia. Singulaariset strategiat ovat mahdollisia evolutiivisia päätepisteitä tai haarautumispisteitä, joissa residenttipopulaation strategioiden määrä kasvaa. Yksinkertaisissa matemaattisissa malleissa singulaariset strategiat voidaan laskea analyyttisesti, mutta yleisesti tämä ei ole aina mahdollista. Tällöin on turvauduttava numeerisiin menetelmiin, joita tarkastellaan tässä tutkielmassa. Esiteltävät menetelmät hyödyntävät kelpoisuusgradienttia ja siksi edellyttävät kelpoisuusfunktion differentioituvuutta.

Luvussa 2 johdetaan Geritzin ja Kisdin artikkeliin [5] perustuva resurssi-kuluttajamalli. Lisäksi johdetaan mallin kausien välinen dynamiikka ja lasketaan mallin kiintopisteet sekä selvitetään niiden stabiilisuus. Luvun 2 jälkimmäisellä puoliskolla perehdytään adaptiiviseen dynamiikkaan ja käydään läpi sille ominaisia käsitteitä, kuten kelpoisuus, singulaarinen strategia, voittamattomuus ja attrahoivuus. Tässä luvussa käytetyt lähteet ovat [4], [7], [8] ja [11].

Luvussa 3 esitellään singulaarisen strategian ratkaisemiseksi numeerisia menetelmiä, jotka eivät suoranaisesti ole optimointimenetelmiä. Aluksi esiteltävä kelpoisuusgradienttimenetelmä on räätälöity nimenomaan luvussa 2 johdettuun resurssi-kuluttajamalliin. Sen jälkeen menetelmä yleistetään käytettäväksi kaikkiin muihinkin mahdollisiin malleihin ja sitä kutsutaan yleiseksi kelpoisuusgradienttimenetelmäksi. Sen toimintaa testataan käyttämällä sitä kahteen testifunktioon. Luvun 3 lopussa esitellään paranneltu versio kelpoisuusgradienttimenetelmästä, joka pohjautuu generoitujen suuntien välisiin kulmiin. Huomataan, että tällä tavoin menetelmää saadaan tehostettua merkittävästi.

Luvussa 4 tarkastellaan lukuisia optimointimenetelmiä singulaarisen strategian ratkaisemiseksi. Kunkin menetelmän kohdalla aloitetaan ensin teorian käsittelyllä ja lopuksi havainnollistetaan menetelmän toimintaa sovellettaessa sitä luvussa 2 johdettuun resurssi-kuluttajamalliin. Luku perustuu lähteisiin [1], [2], [3], [6], [9] ja [10].

Luvussa 5 tehdään aiheeseen liittyviä päätelmiä ja tutkielman lopuksi esite-

tään menetelmillä saatu data liitteenä.

2 Evoluution matemaattinen mallintaminen

Tässä luvussa johdetaan jollekin eläinpopulaatiolle (esimerkiksi hyönteisille) resurssi-kuluttajamalli, jota tullaan käyttämään esimerkkimallina myöhemmin tarkasteltaville numeerisille menetelmille. Lisäksi käydään läpi matemaattiseen mallintamiseen ja adaptiiviseen dynamiikkaan liittyvää teoriaa.

2.1 Esimerkkimallin johtaminen

Tarkastellaan resurssi-kuluttajamallia, jossa on kaksi resurssia R_1 ja R_2 sekä m kuluttajaa x_j. Jatkuva-aikaisen lisääntymiskauden aikana ($t \in [0,1]$) kuluttajat syövät resursseja massavaikutuksen lain mukaan intensiteetillä $\beta_i^j, i = 1,2$, ja tuottavat munia E_j intensiteetillä γ_j, joka on resurssien kulutuksen suhteen vähenevä funktio (katso kaava (2)). Tällöin resurssien kulutuksen kasvattaminen vähentää munien tuottamista, mikä tekee resurssien liiallisen kuluttamisen kannattamattomaksi. Tälle voidaan löytää myös luonnollinen tulkinta: mitä enemmän kuluttaja käyttää aikaa resurssien hankkimiseen, sitä vähemmän sillä on aikaa tuottaa jälkeläisiä. Edellä mainitut populaatiokoot tulkitaan populaatiotiheyksinä. Kauden aikana ei esiinny aikuiskuolleisuutta, mutta tuotetut munat kuolevat vakiointensiteetillä δ. Kauden lopuksi kaikki aikuiset kuolevat ja jäljellä olevista munista kuoriutuu seuraavan kauden aikuiset selviytymistodennäköisyydellä σ. Kauden $n = 1, 2, ...$ sisäinen dynamiikka on

$$\begin{cases} \dot{R}_1^n(t) = \alpha R_1^n(t) f_1(R_1^n(t)) - R_1^n(t) \sum_j \beta_1^j x_j^n(t) \\ \dot{R}_2^n(t) = \alpha R_2^n(t) f_2(R_2^n(t)) - R_2^n(t) \sum_j \beta_2^j x_j^n(t) \\ \dot{E}_j^n(t) = \gamma_j x_j^n(t)(\beta_1^j R_1^n(t) + \beta_2^j R_2^n(t)) - \delta E_j^n(t), j = 1, ..., m \\ \dot{x_j^n}(t) = 0, j = 1, ..., m. \end{cases}$$

Resurssin kasvussa ilman kuluttajia α on skaalausparametri ja funktiot f_1 ja f_2 ovat jatkuvia ja monotonisesti väheneviä välillä $(0, \infty)$. Lisäksi oletetaan, että $\lim_{R_1 \to 0} R_1 f_1(R_1)$ ja $\lim_{R_2 \to 0} R_2 f_2(R_2)$ ovat äärellisiä ja on olemassa sellainen positiivinen luku K_i, että $f_i(R_i) > 0$ kun $R_i < K_i$ ja $f_i(R_i) < 0$ kun $R_i > K_i$, $i = 1, 2$.

Oletetaan, että resurssien dynamiikka on hyvin nopeaa verrattuna kuluttajien ja munien dynamiikkaan. Tällöin resurssipopulaatiot ovat *kvasitasapainotilassa*

$$\hat{R}_i^n(t) = \left(f_i^{-1}\left(\frac{1}{\alpha} \sum_j \beta_i^j x_j^n(t) \right) \right)^+,$$

missä $(f_i^{-1}(\frac{1}{\alpha}\sum_j \beta_i^j x_j^n))^+ := \max\{0, f_i^{-1}(\frac{1}{\alpha}\sum_j \beta_i^j x_j^n)\}$. Resurssipopulaatioiden kvasitasapainotilat ovat vakioita kauden aikana, sillä aikuiskuolleisuutta ei ole $(x_j^n(t) = x_j^n(0), t \in [0,1])$. Kun kvasitasapainotilat sijoitetaan munien dynamiikkaa kuvaavaan yhtälöön, saadaan alkuehdolla $E_j^n(0) = 0$ ratkaistuksi munien määrä $E_j^n(t)$. Yhtälöstä $x_j^{n+1}(0) = \sigma E_j^n(1)$ kausien väliseksi dynamiikaksi saadaan

$$x_j^{n+1}(0) = \lambda_j x_j^n(0) \sum_{i=1}^{2} \beta_i^j \left(f_i^{-1}\left(\frac{1}{\alpha} \sum_j \beta_i^j x_j^n(0) \right) \right)^+, \tag{1}$$

missä $\lambda_j = \frac{1}{\delta}\sigma\gamma_j(1 - e^{-\delta})$.
Oletetaan, että resurssien kasvu ilman kuluttajia on jatkuva sisäänvirtaus kuolleisuudella eli

$$\dot{R}_i^n(t) = \alpha - \frac{\alpha R_i^n(t)}{K_i}.$$

Toisin sanoen ilman kuluttajia resurssia virtaa systeemiin sisään tasaisesti ja sitä poistuu vakiokuolleisuudella. Oletetaan lisäksi, että munien tuottonopeus vähenee eksponentiaalisesti resurssien kulutusnopeuden funktiona. Tällöin $f_i(R_i) = \frac{1}{R_i} - \frac{1}{K_i}$ eli $f_i^{-1}(R_i) = \frac{K_i}{K_i R_i + 1} > 0$ ja

$$\gamma_j = e^{-(\beta_1^j + \beta_2^j)}. \tag{2}$$

Kausien välinen dynamiikka voidaan nyt kirjoittaa muodossa

$$g(x_j^n) := x_j^{n+1} = \lambda_j x_j^n \alpha \left[\frac{\beta_1^j K_1}{K_1 \sum_j \beta_1^j x_j^n + \alpha} + \frac{\beta_2^j K_2}{K_2 \sum_j \beta_2^j x_j^n + \alpha} \right], \tag{3}$$

missä $\lambda_j = \lambda(\beta_1^j, \beta_2^j) = \frac{1}{\delta}\sigma e^{-(\beta_1^j + \beta_2^j)}(1 - e^{-\delta})$.

2.2 Kiintopisteet ja stabiilisuus

Diskreettiaikaisessa mallissa *kiintopisteitä* ovat pisteet x^*, joille $g(x^*) = x^*$. Kun $j = 1$, kuvauksen $g(x)$ kiintopisteet ovat $x = 0$ tai

$$x = \alpha\left(\lambda - \frac{K_1\beta_1 + K_2\beta_2}{2K_1K_2\beta_1\beta_2} \pm \sqrt{\left(\frac{K_1\beta_1 - K_2\beta_2}{2K_1K_2\beta_1\beta_2}\right)^2 + \lambda^2}\right).$$

Jotta

$$x = \alpha\left(\lambda - \frac{K_1\beta_1 + K_2\beta_2}{2K_1K_2\beta_1\beta_2} - \sqrt{(\frac{K_1\beta_1 - K_2\beta_2}{2K_1K_2\beta_1\beta_2})^2 + \lambda^2}\right) \geq 0$$

eli se olisi sallittu kiintopiste, niin saadaan $\lambda(K_1\beta_1 + K_2\beta_2) \leq 1$. Toisaalta, jotta kiintopiste voisi olla ylipäätään positiivinen, on oltava

$$\lambda > \frac{K_1\beta_1 + K_2\beta_2}{2K_1K_2\beta_1\beta_2}.$$

Tällöin kuitenkin $\lambda(K_1\beta_1 + K_2\beta_2) > 1$. Siispä kiintopiste ei ole milloinkaan sallittu. Kiintopiste

$$x^* = \alpha\left(\lambda - \frac{K_1\beta_1 + K_2\beta_2}{2K_1K_2\beta_1\beta_2} + \sqrt{(\frac{K_1\beta_1 - K_2\beta_2}{2K_1K_2\beta_1\beta_2})^2 + \lambda^2}\right)$$

on positiivinen, kun

$$\lambda > \frac{1}{K_1\beta_1 + K_2\beta_2}, \tag{4}$$

missä $\lambda = \frac{1}{\delta}\sigma\gamma(1 - e^{-\delta})$.

Yleisesti kiintopisteen x^* läheisyydessä saadaan approksimaatio $g(x_n) \approx g(x^*) + g'(x^*)(x_n - x^*) = x^* + g'(x^*)(x_n - x^*)$, joka voidaan esittää lineaarikuvauksena $y_{n+1} = g'(x^*)y_n$, kun merkitään $y_n = x_n - x^*$. Kuvausta iteroimalla saadaan $y_{n+1} = (g'(x^*))^{n+1}y_0$. Täten kiintopisteeseen konvergoiminen riippuu kuvauksen g derivaatan arvosta kiintopisteessä. Kun $|g'(x^*)| < 1$, iteraatiopisteet x_n lähestyvät pistettä x^*, jolloin sanotaan, että kiintopiste x^* on stabiili. Jos $0 < g'(x^*) < 1$, kiintopiste x^* on monotonisesti stabiili eli pisteet x_n lähestyvät sitä samalta puolelta. Jos $-1 < g'(x^*) < 0$, kiintopiste x^* on alternoivasti stabiili, jolloin pisteet x_n lähestyvät sitä oskilloivasti eli

heilahdellen puolelta toiselle.
Esimerkkimallin tilanteessa

$$g'(x) = \alpha\lambda\left[\frac{K_1\beta_1}{\alpha + K_1\beta_1 x} + \frac{K_2\beta_2}{\alpha + K_2\beta_2 x} - x\left(\left(\frac{K_1\beta_1}{\alpha + K_1\beta_1 x}\right)^2 + \left(\frac{K_2\beta_2}{\alpha + K_2\beta_2 x}\right)^2\right)\right],$$

johon sijoittamalla kiintopiste $x^* = 0$ saadaan

$$g'(0) = (K_1\beta_1 + K_2\beta_2)\lambda.$$

Huomataan, että $g'(0) > 0$ ja lisäksi $g'(0) < 1$, kun $\lambda < \frac{1}{K_1\beta_1 + K_2\beta_2}$. Positiivisen kiintopisteen tapauksessa derivaatan $g'(x^*)$ lauseke on jo huomattavasti mutkikkaampi. Sitä tutkimalla voidaan vastaavasti todeta, että $g'(x^*) > 0$ ja $g'(x^*) < 1$, kun $\lambda > \frac{1}{K_1\beta_1 + K_2\beta_2}$.
Saadaan siis tapaukset

- Kun $\lambda < \frac{1}{K_1\beta_1 + K_2\beta_2}$, kiintopiste $x = 0$ on yksikäsitteinen ja (monotonisesti) stabiili.
- Kun $\lambda > \frac{1}{K_1\beta_1 + K_2\beta_2}$, kiintopiste $x = 0$ on epästabiili ja kiintopiste $x^* = \alpha\left(\lambda - \frac{K_1\beta_1 + K_2\beta_2}{2K_1K_2\beta_1\beta_2} + \sqrt{(\frac{K_1\beta_1 - K_2\beta_2}{2K_1K_2\beta_1\beta_2})^2 + \lambda^2}\right)$ on (monotonisesti) stabiili.

2.3 Kelpoisuus ja singulaariset strategiat

Jotta adaptiivisen dynamiikan työkaluja voidaan käyttää, mallista on tehtävä joitakin lisäoletuksia. Oletetaan nyt, että yksilöt lisääntyvät suvuttomasti, jolloin jälkeläinen on geneettisesti identtinen vanhempansa kanssa, jollei mutaatiota tapahdu. Tällöin jälkeläisen strategia on yleensä sama kuin vanhemmalla. Mutaatioita syntyy satunnaisesti ja ne ovat harvinaisia ja pieniä. Tällöin niin sanotun putkilauseen (eng. Tube Theorem) nojalla mutantti perii residentin attraktorin [4].
Kausien välisestä dynamiikasta voidaan johtaa kelpoisuutta kuvaava keskimääräinen lisääntymisluku, joka on kuluttajan x_j odotettu jälkeläisten lukumäärä seuraavassa sukupolvessa yhtä kuluttajaa kohti edeltävässä sukupolvessa. Oletetaan, että kuluttajapopulaatio koostuu yhdestä residenttipopulaatiosta x strategianaan $\beta = (\beta_1, \beta_2)$ ja erittäin pienestä määrästä mutantteja x^m, joiden strategia $\beta^m = (\beta_1^m, \beta_2^m)$ poikkeaa hieman residentistä.

Yhtälöstä (1) nähdään, että

$$x_j^{n+1} = \lambda_j x_j^n \sum_{i=1}^{2} \beta_i^j R_i^n,$$

missä R_1^n ja R_2^n ovat vakioita kauden aikana. Mutantin *kelpoisuus* residentin määräämässä ympäristössä E_{res} on

$$w(\beta^m, E_{res}) = w(\beta^m, (R_1, R_2)) = \lambda_m \sum_{i=1}^{2} \beta_i^m R_i.$$

Käyttämällä yhtälöä (3) ja valitsemalla $j = 2$ kelpoisuus voidaan esittää muodossa

$$w(\beta^m, \beta) = \lambda_m \alpha \left[\frac{\beta_1^m K_1}{K_1 \beta_1 x + \alpha} + \frac{\beta_2^m K_2}{K_2 \beta_2 x + \alpha} \right],$$

missä $\lambda_m = \lambda(\beta_1^m, \beta_2^m) = \frac{1}{\delta} \sigma e^{-(\beta_1^m + \beta_2^m)}(1 - e^{-\delta})$. Jos $w(\beta^m, \beta) < 1$, mutantti ei voi tehdä invaasiota ja kuolee pois. Jos $w(\beta^m, \beta) > 1$, mutantti voi tehdä invaasion ja mahdollisesti syrjäyttää residentin. Erityisesti $w(\beta, \beta)$=1.
Kelpoisuussuureelle w pätee likimäärin

$$w(\beta^m, \beta) \approx w(\beta, \beta) + (\beta^m - \beta)^T D(\beta), \tag{5}$$

missä $D(\beta)$ on *kelpoisuusgradientti*, joka koostuu komponenteista

$$D_i(\beta) = \left. \frac{\partial w(\beta^m, \beta)}{\partial \beta_i^m} \right|_{\beta^m = \beta}. \tag{6}$$

Yhtälöstä (5) nähdään, että mutantti voi tehdä invaasion, jos

$$(\beta^m - \beta)^T D(\beta) > 0. \tag{7}$$

Likimääräistyksen kautta tehtävä kelpoisuuden tarkastelu kelpoisuusgradienttien avulla on mielekäs, sillä mutaatiot oletettiin pieniksi. Strategiaa, jolle $D(\beta) = 0$, kutsutaan *singulaariseksi*. Singulaarinen strategia on siis isokliinien $D_i(\beta) = 0$ leikkauspiste.
Singulaarisen strategian β^* läheisyydessä pätee likimäärin

$$w(\beta^m, \beta^*) \approx w(\beta^*, \beta^*) + \frac{1}{2}(\beta^m - \beta^*)^T H(\beta^*)(\beta^m - \beta^*), \tag{8}$$

missä $H(\beta^*)$ on niin sanottu *Hessen matriisi*, joka koostuu alkioista

$$H(\beta^*)_{ij} = \left. \frac{\partial^2 w(\beta^m, \beta)}{\partial \beta_i^m \partial \beta_j^m} \right|_{\beta^m = \beta = \beta^*}.$$

Yhtälöstä (8) huomataan, että jos Hessen matriisi $H(\beta^*)$ on negatiividefiniitti, niin yksikään mutantti ei voi tehdä invaasiota, kun residentillä on singulaarinen strategia β^*. Kyseistä strategiaa kutsutaan *voittamattomaksi*. Vastaavasti matriisin $H(\beta^*)$ ollessa positiividefiniitti mikä tahansa mutantti voi tehdä invaasion ja syrjäyttää residentin. Tällaista singulaarista strategiaa sanotaan *häviäväksi*. Hessen matriisi voi olla myös indefiniitti, jolloin singulaarinen strategia on satulapiste.

2.4 Attrahoivuus

Koska mutaatiot oletettiin pieniksi ja harvinaisiksi, voidaan strategian β dynamiikkaa kuvata niin kutsutulla *kanonisella yhtälöllä*

$$\frac{d\beta}{dt} = \frac{1}{2}\mu(\beta)\sigma^2(\beta)n(\beta)D(\beta), \tag{9}$$

missä $\mu(\beta)$ on mutaationopeus, $\sigma^2(\beta)$ mutaatiojakauman varianssi-kovarianssimatriisi, $n(\beta)$ populaatiokoko tasapainotilassa ja $D(\beta)$ kelpoisuusgradientti. Yhtälö (9) voidaan esittää myös muodossa

$$\frac{d\beta}{dt} = C(\beta)D(\beta),$$

missä $C(\beta)$ on symmetrinen positiividefiniitti matriisi. Strategian β dynamiikka riippuu siis sekä mutationaalisesta prosessista että luonnonvalinnasta. Singulaarisen strategian β^* lähistöllä voidaan käyttää kelpoisuusgradientista approksimatiota

$$D(\beta) \approx D(\beta^*) + J(\beta^*)(\beta - \beta^*) = J(\beta^*)(\beta - \beta^*),$$

missä $J(\beta^*)$ on kelpoisuusgradientin Jacobin matriisi ja se koostuu alkioista

$$J(\beta^*)_{ij} = \left. \frac{\partial D_i(\beta)}{\partial \beta_j} \right|_{\beta = \beta^*}.$$

Merkitsemällä $C = C(\beta^*)$, $J = J(\beta^*)$ ja käyttäen äskeistä approksimaatiota voidaan muodostaa linearisoitu kanoninen yhtälö

$$\frac{d}{dt}(\beta - \beta^*) = CJ(\beta - \beta^*). \tag{10}$$

Approksimaation käyttäminen on jälleen mielekästä, sillä mutaatiot oletettiin pieniksi. Linearisoitu yhtälö (10) on asymptoottisesti stabiili pisteessä β^*, jos matriisin CJ ominaisarvojen reaaliosat ovat kaikki negatiivisia ja muulloin epästabiili. Koska matriisi C on symmetrinen positiividefiniitti matriisi, stabiilisuusehto voidaan esittää kelpoisuusgradientin Jacobin matriisin J avulla: Singulaarinen strategia on vahvasti konvergoiva, jos kelpoisuusgradientin Jacobin matriisi on negatiividefiniitti. Yleisesti singulaarista strategiaa sanotaan vahvasti konvergoivaksi, jos kanoninen yhtälö (9) suppenee kaikilla symmetrisillä positiividefiniiteillä varianssi-kovarianssimatriiseilla $\sigma^2(\beta)$ singulaarisen strategian lähistöllä.

Vahva konvergenssi ei takaa sitä, että populaatio päätyisi singulaariseen strategiaan sen läheisyydessä kaikissa mahdollisissa tilanteissa. Tätä varten voidaan määritellä absoluuttinen konvergenssi, jossa singulaariseen strategiaan päädytään aina sen läheisyydessä. Ehdot absoluuttiseen konvergenssiin ovat luonnollisesti paljon tiukemmat. Jos kelpoisuusgradientti voidaan esittää muodossa

$$D(\beta) = r(\beta)\nabla F(\beta), \tag{11}$$

missä $r(\beta)$ on positiivinen ja $F(\beta)$ on funktio, jolla on paikallinen maksimi pisteessä β^*, niin tällöin piste β^* on absoluuttisesti konvergoiva. Jos kelpoisuusgradientti on muotoa (11), niin sen Jacobin matriisi on symmetrinen singulaarisessa pisteessä. Välttämätön ehto absoluuttiselle konvergenssille on se, että Jacobin matriisi on symmetrinen pisteessä β^*. Riittävä ehto on, että Jacobin matriisi on symmetrinen ja negatiividefiniitti [7].

3 Kelpoisuusgradientin suuntaa hyödyntävät menetelmät

Useissa tilanteissa singulaaristen strategioiden ratkaiseminen analyyttisesti on hyvin hankalaa. Tässä luvussa tarkastellaan näiden strategioiden löytämiseksi numeerisia menetelmiä, joissa ei käytetä optimoitavana olevaa kohdefunktiota (vrt. luku 4).

3.1 Kelpoisuusgradienttimenetelmä (KGM)

Kuten aiemmin todettiin, ehto mutantin kelpoisuudelle voitiin esittää muodossa (7). Kelpoisuusgradienttimenetelmän (KGM) idea pohjautuu tähän. Kussakin pisteessä lasketaan kelpoisuusgradientti ja seuraavan pisteen laskemiseksi muutetaan mutantin strategiaa kelpoisuusgradientin suuntaan. Singulaarisessa strategiassa kelpoisuusgradientti häviää ja tätä voidaan käyttää lopetuskriteerinä. Jos sisäsingulariteetteja ei ole, algoritmi pysähtyy lopulta sallitun alueen reunalle tai kulmapisteeseen. Menetelmä on diskreetti eli se säätelee itse askelpituutensa. Menetelmä on ahne eikä se huomioi sitä, mihin suuntaan mutaatiot todellisuudessa tapahtuisivat. Tarkasteltava algoritmi palauttaa singulaarisen strategian, jos se on kelpoisuusmaksimi. Algoritmin vaiheet voidaan esittää seuraavasti:

1. Aseta lopetusta varten $\varepsilon > 0$. Olkoon $\Delta \geq \varepsilon$ askelpituus alussa. Valitse aloituspiste x^1. Jos populaatio ei ole elinkelpoinen, niin lopeta (esimerkkimallissa ehto (4) ei ole voimassa). Muutoin aseta $k = 1$ ja siirry vaiheeseen 2.

2. Jos $|D_i(x^k)| < \varepsilon$ tai jos $x_i^k \notin (\varepsilon, 1-\varepsilon)$, $i = 1, 2$, niin lopeta ja x^k on ratkaisu. Muutoin aseta $j = 1$ ja $\rho_1 = \rho_2 = \phi = \Delta$ ja siirry vaiheeseen 3.

3. Jos $D_i(x^k)D_i(x^k + \phi D(x^k)) > 0$ ja $x_i^k + \phi D_i(x^k) \in (0, 1)$, $i = 1, 2$, niin siirry vaiheeseen 5. Muussa tapauksessa siirry vaiheeseen 4.

4. Jos $j < m$, niin aseta $\phi \leftarrow \frac{1}{2}\phi$ ja $j \leftarrow j + 1$ ja siirry vaiheeseen 3.
   Jos $j = m$, niin siirry vaiheeseen 5.

5. Jos $x_i^k + \phi D_i(x^k) \in (0, 1)$, $i = 1, 2$, niin aseta $x^{k+1} = x^k + \phi D(x^k)$.
   Aseta lisäksi $k \leftarrow k + 1$ ja siirry vaiheeseen 2.
   Jos $x_1^k + \phi D_1(x^k) \notin (0, 1)$, niin siirry vaiheeseen 6.
   Jos $x_2^k + \phi D_2(x^k) \notin (0, 1)$, niin siirry vaiheeseen 7.

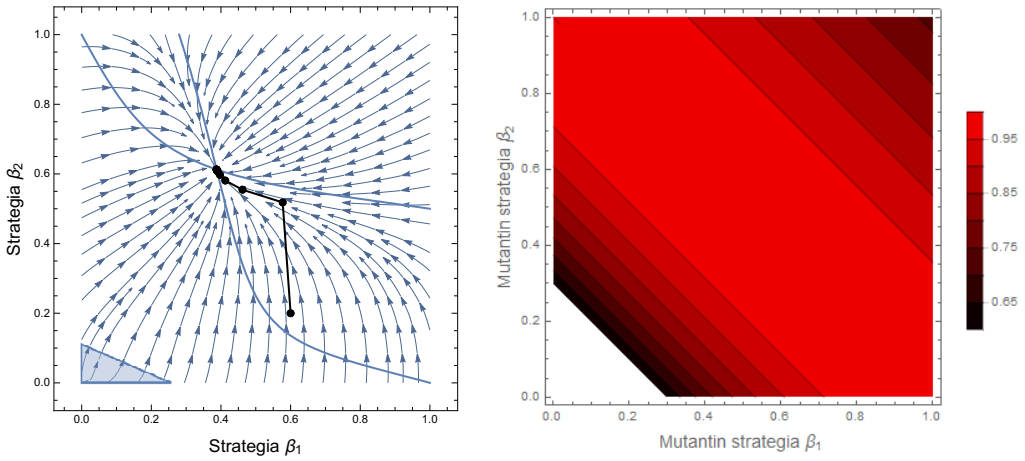
6. Jos $x_2^k + \rho_2 \cdot \text{sgn}(D_2(x^k)) \in (0, 1)$, niin aseta $x_2^{k+1} = x_2^k + \rho_2 \cdot \text{sgn}(D_2(x^k))$ ja $x_1^{k+1} = x_1^k$. Aseta lisäksi $k \leftarrow k + 1$ ja siirry vaiheeseen 2.
   Muulloin aseta $\rho_2 = \frac{1}{2}\rho_2$ ja toista vaihe 6.

7. Jos $x_1^k + \rho_1 \cdot \text{sgn}(D_1(x^k)) \in (0, 1)$, niin aseta $x_1^{k+1} = x_1^k + \rho_1 \cdot \text{sgn}(D_1(x^k))$ ja $x_2^{k+1} = x_2^k$. Aseta lisäksi $k \leftarrow k + 1$ ja siirry vaiheeseen 2.
   Muulloin aseta $\rho_1 = \frac{1}{2}\rho_1$ ja toista vaihe 7.

Algoritmissa esiintyvä etumerkkifunktio $\text{sgn}(x)$ määritellään

$$\text{sgn}(x) = \frac{x}{|x|}, x \neq 0.$$

Vaihe 2 tarkastaa lopetuskriteerien toteutumisen. Lopetuskriteerinä käytetään kelpoisuusgradientin häviämistä tai strategia-avaruuden reunalle ajautumista. Vaihe 3 (askelpituusehto) selvittää, onko kelpoisuusgradientin suuntaan edetty seuraava piste sallitulla alueella ja vaihtaako jokin kelpoisuusgradientin komponenteista etumerkkinsä kyseisessä pisteessä. Lisäksi voitaisiin valvoa elinkelpoisuusehdon toteutumista, mutta sitä nyt ei ole huomioitu tässä. Jos uusi piste ei ole sallittu tai jollakin komponentilla etumerkki vaihtuu, lyhennetään askelpituutta vaiheessa 4. Vaihe 4 estää myös algoritmin juuttumisen paikoilleen isokliiniin $D_i(x) = 0$ loikkaamalla tarvittaessa sen toiselle puolelle. Vaiheessa 5 siirrytään uuteen pisteeseen, mikäli se on sallittu. Jos piste ei ole sallittu vaiheen 4 jälkeen, niin tällöin muutetaan kelpoisuusgradientin merkin suuntaisesti vain sitä komponenttia, joka ei aiheuta pisteen siirtymistä ei-sallitulle alueelle (vaiheet 6 ja 7). Jälleen askelpituutta lyhennetään tarvittaessa.



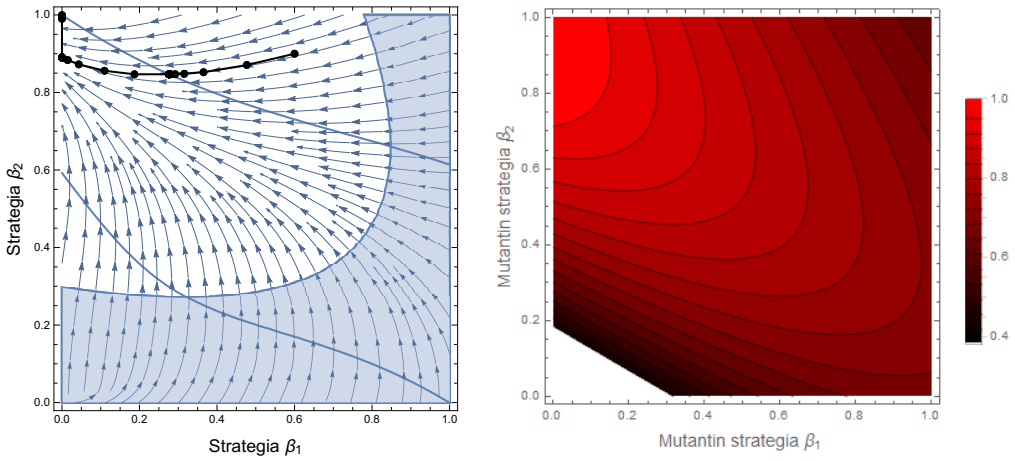
(a) **(b)**

Kuva 1: (a) Kelpoisuusgradienttimenetelmän kulku lähtien pisteestä $(0.6, 0.2)$. Algoritmi pysähtyy isokliinien leikkauspisteeseen, joka on singulaarinen strategia. Kuvaan on piirretty isokliinit ja mallin dynamiikkaa kuvaavaa virtauskenttäkaavio. Elinkelvoton alue on merkitty tummalla pohjalla. (b) Mutantin kelpoisuusmaasto, kun residentti on singulaarisessa strategiassa. Kuvasta nähdään, että singulaarinen strategia on voittamaton. Mallin parametrit: $\delta = 1$, $\sigma = 0.8$, $\alpha = 1$, $K_1 = 10$, $K_2 = 20$. Kelpoisuusgradienttimenetelmän parametrit: $\varepsilon = 0.0001$, $\Delta = 0.5$, $m = 26$

3.1.1 Esimerkkejä

Algoritmi toteutettiin käyttäen Mathematica-ohjelmistoa, jossa käytettiin ohjelman omaa *Sign*-funktiota algoritmin vaiheissa 3, 6 ja 7. Eri lähtöpisteillä ja syötetyillä askelpituuksilla algoritmi tuotti aina oikean vastauksen, mutta siihen vaadittujen iteraatioiden määrä luonnollisesti vaihteli. Esimerkiksi hyvin pienillä askelpituuksilla algoritmin toiminta on tehotonta. Seuraavaksi esitettävissä esimerkeissä (kuten myöhemmissäkin esimerkeissä) syötettyjä parametrien arvoja ei ole valittu sellaisiksi, että algoritmi toimisi kaikkein tehokkaimmin.

Kuvissa 1a ja 2a on esitetty algoritmin toimintaa aikaisemmin tarkastellun resurssi-kuluttajamallin tapauksessa. Kuvan 1a tilanteessa algoritmi päätyy singulaariseen strategiaan $\beta_1^* \approx 0.387464, \beta_2^* \approx 0.612507$ 11 iteraatiolla. Todellinen arvo on $\beta_1^* \approx 0.387423, \beta_2^* \approx 0.612577$. Kyseinen singulaarinen stra-



(a) **(b)**

Kuva 2: (a) Kelpoisuusgradienttimenetelmän kulku lähtien pisteestä $(0.6, 0.9)$. Algoritmi pysähtyy kulmapisteen $(0, 1)$ läheisyyteen. Kuvaan on piirretty isokliinit ja mallin dynamiikkaa kuvaavaa virtauskenttäkaavio. Elinkelvoton alue on merkitty tummalla pohjalla. (b) Mutantin kelpoisuusmaasto, kun residentti on pisteessä $(0, 1)$. Kuvasta nähdään, että residentin ollessa tässä pisteessä yksikään mutantti ei voi tehdä invaasiota. Mallin parametrit: $\delta = 0.1,\ \sigma = 0.95,\ \alpha = 1,\ K_1 = 2,\ K_2 = 5$. Kelpoisuusgradienttimenetelmän parametrit: $\varepsilon = 0.001,\ \Delta = 0.2,\ m = 26$

tegia on voittamaton (kuva 1b). Lisäksi Jacobin matriisi on singulaarisessa strategiassa negatiividefiniitti, joten se on vahvasti konvergoiva. Jacobin matriisi ei kuitenkaan ole symmetrinen, joten se ei ole absoluuttisesti konvergoiva. Tällöin on olemassa jokin sellainen positiividefiniitti matriisi C, että matriisilla CJ on positiivisen reaaliosan omaava ominaisarvo. Siispä joissakin (vaikkakin melko epätodennäköisissä tilanteissa) populaatio saattaa ajautua pois singulaarisen strategian β^* lähistöltä.

Kuvan 2a tapauksessa sisäsingulariteettia ei ole, vaan populaatio ajautuu strategia-avaruuden kulmapisteeseen $(0, 1)$ 36 iteraatiolla. Isokliinin $D_2(\beta) = 0$ läheisyydessä pisteet pakkautuvat tiiviisti isokliinin yläpuolelle, kunnes lopulta siirtyvät sen toiselle puolelle. Reunaa $\beta_1 = 0$ lähestyttäessä algoritmi ottaa jälleen pienempiä askelia, kunnes tapahtuu hyppy kulmapisteen läheisyyteen. Algoritmi pysähtyy pisteeseen $\beta_1^* \approx 4.67795 \times 10^{-10}, \beta_2^* \approx 0.999636$. Populaation ajauduttua kulmapisteeseen yksikään mutantti ei voi tehdä in-

vaasiota (kuva 2b).

3.2 Yleinen kelpoisuusgradienttimenetelmä

Aikaisemmin esitetty kelpoisuusgradienttimenetelmä sopi hyvin johdettuun resurssi-kuluttajamalliin, jossa parametrit oli valittu sellaisiksi, että singulaarinen strategia oli kelpoisuusmaksimi ja strategia-avaruudeksi riitti tarkastella aluetta $[0,1]\times[0,1]$. Jalostetaan seuraavaksi menetelmä toimimaan yleisessä tapauksessa, jossa singulaarinen strategia voi olla myös kelpoisuusminimi tai satulapiste ja strategia-avaruutena on $\mathbb{R}_+ \times \mathbb{R}_+$. Koska strategia-arvot voivat nyt kasvaa mielivaltaisen suureksi, on syytä asettaa niille jokin yläraja ylivuodon välttämiseksi. Menetelmän ideana on lähteä ensin kelpoisuusgradienttien suuntaisesti etsimään singulaarista strategiaa. Jos singulaarista strategiaa ei tällä tavalla löydy, niin algoritmi pysähtyy ennalta määrätyn iteraatiomäärän jälkeen. Tämän jälkeen algoritmi aloittaa jälleen annetusta alkupisteestä ja lähtee etenemään gradienttivektorin vastakkaiseen suuntaan. Jos algoritmi saavuttaa edelleen asetetun iteraatiomäärän pysähtymättä aiemmin, singulaarista strategiaa ei joko ole olemassa tai algoritmi ei ole vielä ehtinyt saavuttaa sitä annetulla tarkkuudella, jolloin iteraatiomäärää on kasvatettava. Ohjelman antamasta tulosteesta voi yrittää tehdä päätelmiä siitä, kumpi näistä tapauksista saattaisi olla kyseessä. Yleisen kelpoisuusgradienttimenetelmän vaiheet voidaan esittää seuraavasti:

1. Aseta lopetusta varten $\varepsilon > 0$. Olkoon $\Delta \geq \varepsilon$ askelpituus alussa. Valitse aloituspiste x^1. Jos populaatio ei ole elinkelpoinen, niin lopeta. Olkoon suurin sallittu iteraatiomäärä N, suurin iteraatiomäärä askeleen puolituksessa m sekä strategiakomponenttien yläraja B. Aseta suuntakertoimeksi $\xi = 1$ (edetään kelpoisuusgradientin suuntaisesti), kierroslaskuriksi $k = 1$ ja siirry vaiheeseen 2.

2. Jos $|D_i(x^k)| < \varepsilon$, $i = 1, 2$ tai jos $k = 2N$, niin lopeta ja x^k on ratkaisu. Muutoin aseta $j = 1$ ja $\rho_1 = \rho_2 = \phi = \Delta$ ja siirry vaiheeseen 3.

3. Jos $D_i(x^k)D_i(x^k + \xi\phi D(x^k)) > 0$ ja $x_i^k + \xi\phi D_i(x^k) > 0$, $i = 1, 2$, niin siirry vaiheeseen 5. Muussa tapauksessa siirry vaiheeseen 4.

4. Jos $j < m$, niin aseta $\phi \leftarrow \frac{1}{2}\phi$ ja $j \leftarrow j + 1$ ja siirry vaiheeseen 3.
   Jos $j = m$, niin siirry vaiheeseen 5.

5. Jos $x_i^k + \xi\phi D_i(x^k) > 0$, $i = 1, 2$, niin aseta $x^{k+1} = x^k + \xi\phi D(x^k)$. Aseta lisäksi $k \leftarrow k + 1$ ja siirry vaiheeseen 8.
   Jos $x_1^k + \xi\phi D_1(x^k) < 0$, niin siirry vaiheeseen 6.
   Jos $x_2^k + \xi\phi D_2(x^k) < 0$, niin siirry vaiheeseen 7.

6. Jos $x_2^k + \xi\rho_2 \cdot \mathrm{sgn}(D_2(x^k)) > 0$, niin aseta $x_2^{k+1} = x_2^k + \xi\rho_2 \cdot \mathrm{sgn}(D_2(x^k))$ ja $x_1^{k+1} = x_1^k$. Aseta lisäksi $k \leftarrow k + 1$ ja siirry vaiheeseen 8.
   Muulloin aseta $\rho_2 = \frac{1}{2}\rho_2$ ja toista vaihe 6.

7. Jos $x_1^k + \xi\rho_1 \cdot \mathrm{sgn}(D_1(x^k)) > 0$, niin aseta $x_1^{k+1} = x_1^k + \xi\rho_1 \cdot \mathrm{sgn}(D_1(x^k))$ ja $x_2^{k+1} = x_2^k$. Aseta lisäksi $k \leftarrow k + 1$ ja siirry vaiheeseen 8.
   Muulloin aseta $\rho_1 = \frac{1}{2}\rho_1$ ja toista vaihe 7.

8. Jos $k \neq N$ ja $x_i < B$, $i = 1, 2$, niin siirry vaiheeseen 2. Muussa tapauksessa aseta $\xi = -1$ (edetään kelpoisuusgradientin vastaisesti) ja $x^k = x^1$ ja siirry vaiheeseen 2.

Vaiheessa 2 ei ole strategia-arvoja koskevia lopetuskriteerejä toisin kuin aiemmin esitellyssä kelpoisuusgradienttimenetelmässä, sillä muuten algoritmin suoritus voi pysähtyä ennenaikaisesti.

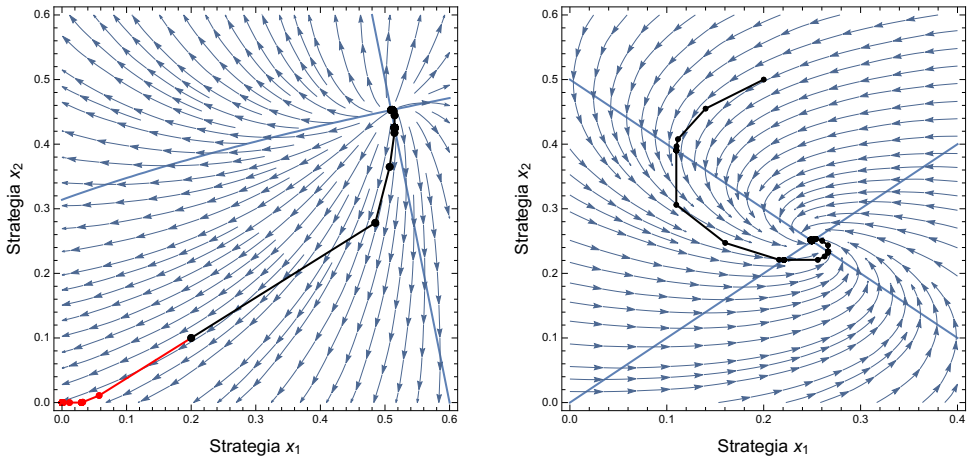
3.2.1 Esimerkkejä

Algoritmin toiminnan havainnollistamiseksi on tarkasteltu kelpoisuusfunktioita, joilla on eräitä toivottuja ominaisuuksia singulaaristen strategioiden suhteen. Kelpoisuusfunktioiden johtaminen tai tulkinta ei ole nyt siis mielenkiinnon kohteena. Tässä käytetyt kelpoisuusfunktiot ovat

$$r = (x_1^m - x_2)^2 - 3x_1^m - x_2 + 3(x_1 + x_1^m)(x_1 + x_2) + (x_1 - x_2^m)^2 - x_2^m + 3(x_2^m)^4, \quad (12)$$

$$r = x_1^2 + x_1^m - (x_1^m)^2 - 2x_1^m x_2 + (x_2 - x_2^m)x_2^m + x_1(x_2 + x_2^m - 1). \quad (13)$$

Kelpoisuusfunktion (12) tapauksessa singulaarinen strategia ($x_1^* \approx 0.51, x_2^* \approx 0.45$) on kelpoisuusminimi. Nyt kelpoisuusgradientit suuntaavaat siitä pois-

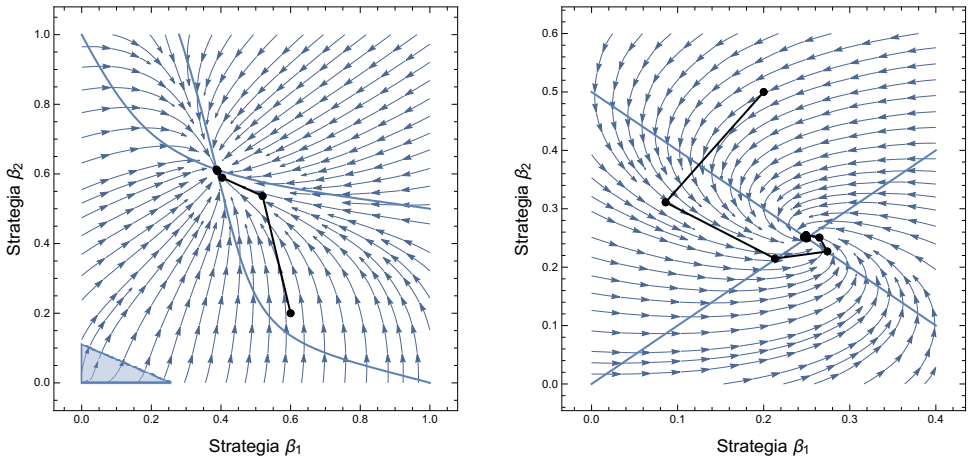


(a) **(b)**

Kuva 3: (a) Yleisen kelpoisuusgradienttimenetelmän kulun havainnollistus, kun kelpoisuusfunktio on muotoa (12) lähtien pisteestä $(0.2, 0.1)$. Algoritmi etenee ensin kelpoisuusgradienttien suuntaisesti (punaiset pisteet), kunnes se palaa takaisin alkupisteeseen ja etenee vastakkaiseen suuntaan (mustat pisteet) löytäen lopulta isokliinien leikkauspisteen. Parametrit: $\varepsilon = 0.0001$, $\Delta = 0.3$, $N = 150$, $B = 100$, $m = 25$. (b) Kelpoisuusfunktio on muotoa (13) ja alkupiste $(0.2, 0.5)$. Algoritmi konvergoi kelpoisuusgradienttien suuntaisesti kohti isokliinien leikkauspistettä kierteisesti (mustat pisteet). Parametrit: $\varepsilon = 0.0001$, $\Delta = 0.3$, $N = 50$, $B = 50$, $m = 25$.

päin, mikä pidentää algoritmin suoritusaikaa singulaarista strategiaa haettaessa. Kuvassa 3a on esitetty algoritmin toiminta lähdettäessä liikkelle pisteestä $(x_1^1, x_2^1) = (0.2, 0.1)$. Aluksi algoritmi etenee kelpoisuusgradienttien suuntaan kohti origoa (punaiset pisteet), kunnes asetetun iteraatiorajan 50 jälkeen eteneminen lopetetaan. Tämän jälkeen algoritmi palaa aloituspisteeseen ja etenee kelpoisuusgradienttien vastakkaiseen suuntaan (mustat pisteet). Singulaarinen strategia saavutetaan neljän desimaalin tarkkuudella 39 iteraatiolla.

Kun kelpoisuusfunktio on muotoa (13), kelpoisuusgradientit ohjaavat singulaariseen strategiaan $(x_1^* \approx 0.25, x_2^* \approx 0.25)$ kierteisesti. Algoritmin suorituskyvyn kannalta tämä on ongelmallista, sillä sen on ylitettävä isokliinejä useita kertoja päästäkseen singulaariseen strategiaan. Tällöin algoritmin vaatima



(a) **(b)**

Kuva 4: (a) Kelpoisuusgradienttimenetelmän kulku resurssi-kuluttajamallin tilanteessa käytettäessä askelpituusehdossa pistetuloa ja aloituspisteenä $(0.6, 0.2)$. Vertaa kuvaan 1a, jossa on käytetty samoja parametreja, mutta ei pistetuloa. (b) Yleisen kelpoisuusgradienttimenetelmän kulku pistetulolla, kun kelpoisuusfunktiona on (13) ja aloituspisteenä $(0.2, 0.5)$. Vertaa kuvaan 3b, jossa on käytetty samoja parametreja, mutta ei pistetuloa askelpituusehdossa.

iteraatiomäärä kasvaa huomattavasti. Käytännön kannalta tällä ei ole suurta merkitystä tässä mallissa, sillä yhden iteraation suorittamiseen kuluva aika on vähäinen. Kuvassa 3b on esimerkkitilanne algoritmin etenemisestä, kun alkupisteeksi on asetettu $(x_1^1, x_2^1) = (0.2, 0.5)$. Kuvasta nähdään, että algoritmin eteneminen hidastuu aina saavuttaessa isokliinin läheisyyteen. Singulaarinen strategia saavutetaan neljän desimaalin tarkkuudella 125 iteraatiolla. Tämän CPU-aika on noin 0.41 sekuntia.

3.3 Paranneltu kelpoisuusgradienttimenetelmä

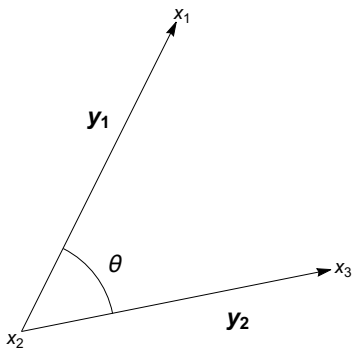
Kuten aiemmin huomattiin, kelpoisuusgradienttimenetelmän heikkoutena oli isokliinin läheisyyteen ajautuminen ja siinä paikallaan polkeminen. Erityisesti, kun isokliinien ylittämisiä tulee paljon (kuten kuvan 3b tapauksessa), algoritmista tulee tehoton. Seuraavaksi pyritään pääsemään eroon tästä heikkoudesta. Aikaisemmin askelpituutta lyhennettiin, jos uudessa pisteessä kel-

poisuusgradientti vaihtoi merkkiään jonkin strategiakomponentin suhteen. Lähdetäänkin nyt askelpituusehtoa varten tutkimaan, muuttuuko pisteiden kautta laskettujen vektoreiden välinen suunta olennaisesti. Sitä varten käytetään pistetuloa. Yleisesti vektoreiden $y_1 \in \mathbb{R}^n$ ja $y_2 \in \mathbb{R}^n$ pistetulo $y_1 \cdot y_2$ on

$$y_1 \cdot y_2 = \|y_1\|\|y_2\| \cos\theta = y_1^T y_2, \tag{14}$$

missä θ on vektoreiden välinen kulma (katso kuva 5). Kaavasta (14) nähdään, että kulman θ suuruuden perusteella pistetulon arvot voidaan luokitella seuraavasti:

- Kun $0 \leq \theta < \pi/2$, niin vektoreiden y_1 ja y_2 pistetulo on positiivinen.
- Kun $\theta = \pi/2$, niin vektoreiden y_1 ja y_2 pistetulo on nolla. Tällöin ne ovat toisiaan vastaan kohtisuorassa.
- Kun $\pi/2 < \theta \leq \pi$, niin vektoreiden y_1 ja y_2 pistetulo on negatiivinen.



Kuva 5: Vektorit y_1 ja y_2 sekä niiden välinen kulma θ.

Oletetaan, että alkuaskelpituudeksi on asetettu $\Delta = \Delta_0$ ja lähtöpisteenä on x_1. Edetään ensin askelpituudella painotettuna kelpoisuusgradientin suuntaisesti pisteeseen x_2. Siitä edetään jälleen askelpituudella painotettuna kelpoisuusgradientin suuntaisesti pisteeseen x_3 (katso kuva 5). Jos nyt vektoreiden $x_1 - x_2$ ja $x_3 - x_2$ pistetulo on positiviinen, niin vektoreiden suunta poikkeaa olennaisesti toisistaan ja askelpituutta lyhennetään puoleen eli $\Delta = \frac{1}{2}\Delta_0$. Lähtien taas pisteestä x_1 toistetaan sama kuin edellä niin kauan, kunnes

$(x_1 - x_2) \cdot (x_3 - x_2) < 0$. Asetetaan $x_1 = x_3$ ja $\Delta = \Delta_0$ ja toistetaan sama kuin edellä, jos lopetusehto ei toteudu.
Kuvassa 4 on esitetty parannellun kelpoisuusgradienttimenetelmän toimintaa aiemmin esiteltyjen esimerkkien tapauksissa. Kuvassa 4a ratkaistaan resurssikuluttajamallin singulaarista strategiaa samoilla parametreilla kuten kuvassa 1a, mutta nyt iteraatioiden määrä on likimain puolet aikaisemmasta; ratkaisuun päästään samalla tarkkuudella vain 6 iteraatiolla. Kuvassa 4b on ratkaistu singulaarista strategiaa, kun kelpoisuusfunktio on muotoa (13). Algoritmi päätyy nyt ratkaisuun selvästi nopeammin kuin kuvan 3b tapauksessa, jossa algoritmin eteneminen hidastui isokliinejä ylitettäessä. Ratkaisuun päästään 12 iteraatiolla CPU-aikana noin 0.02 sekuntia, mikä on huomattava parannus edelliseen.

4 Singulaarisen strategian ratkaiseminen optimointitehtävänä

Kuten aiemmin todettiin, singulaarisessa strategiassa kelpoisuusgradientti on nolla. Olkoon nyt kaavan (6) merkinnöin $f(x) = D_1(x)^2 + D_2(x)^2 + ... + D_n(x)^2$, jota kutsutaan kohdefunktioksi. Oletetaan jatkossa, että funktio f on differentioituva. Tällöin sen gradientti $\nabla f(x)$ koostuu riveistä

$$\nabla f(x)_i = \frac{\partial f(x)}{\partial x_i} = 2\left[D_1(x)\frac{\partial D_1(x)}{\partial x_i} + ... + D_n(x)\frac{\partial D_n(x)}{\partial x_i}\right],\ i = 1, ..., n.$$

Tästä nähdään, että funktion f gradientin laskeminen edellyttää kelpoisuusgradientin derivaattojen tuntemista. Singulaarisen strategian etsiminen voidaan nyt muuntaa optimointitehtäväksi

$$\begin{aligned} \min_x \quad & f(x) \\ \text{s. t.} \quad & g_i(x) \leq 0,\ i = 1, \ldots, h \\ & x_i \geq 0,\ i = 1, \ldots, n, \end{aligned} \tag{15}$$

missä funktiot $g_i(x)$ ovat mallikohtaisia rajoitefunktioita. Ratkaistavana on siis epälineaarinen moniulotteinen rajoitteinen optimointitehtävä. Tähän

muotoon asetellun tehtävän etuna on, että siihen on kehitetty lukuisia ratkaisumenetelmiä, joista tarkastellaan erityisesti derivaattoja hyödyntäviä menetelmiä. Menetelmiä sovelletaan aiemmin esitellyn resurssi-kuluttajamallin singulaaristen strategioiden ratkaisemiseen, missä mallin rajoituksina ovat

$$\begin{cases} \beta_1 \geq 0 \\ \beta_2 \geq 0 \\ \lambda \geq \frac{1}{K_1\beta_1 + K_2\beta_2}. \end{cases}$$

Näistä kaksi ensimmäistä rajoitusta ovat lineaarisia ja kolmas epälineaarinen.

4.1 Ulkopuolinen sakkofunktiomenetelmä

Ulkopuolisen sakkofunktiomenetelmän (USM) ideana on muuntaa rajoitteinen optimointitehtävä rajoitteettomaksi siirtämällä rajoitteet osaksi kohdefunktiota sakkokertoimella kerrotulla sakkofunktiolla. Sakkofunktioiden tarkoituksena on tuottaa kohdefunktioon kustannus ei-sallitussa pisteessä ja sakkokerrointa kasvattamalla pakotetaan ratkaisu lähestymään sallittua aluetta. Esitetään nyt tehtävän (15) epäyhtälörajoitukset yksinkertaisesti muodossa

$$\begin{aligned} \min_x \quad & f(x) \\ \text{s. t.} \quad & g_i(x) \leq 0,\ i = 1, \ldots, m, \end{aligned} \tag{16}$$

missä $m = n + h$. Epäyhtälörajoitteiselle tehtävälle sakkofunktioksi sopii $\alpha(x) = \sum_{i=1}^{m} \psi(g_i(x))$, missä ψ on jatkuva ja $\psi(x) > 0$, kun $x > 0$ ja nolla muulloin. Tyypillisesti sakkofunktio on muotoa $\alpha(x) = \sum_{i=1}^{m} \max\{0, g_i(x)\}^p$. Jatkossa tullaankin käyttämään resurssi-kuluttajamallille derivoituvaa sakkofunktiota

$$\alpha(\beta) = \max\{0, -\beta_1\}^2 + \max\{0, -\beta_2\}^2 + \max\{0, (K_1\beta_1 + K_2\beta_2)^{-1} - \lambda\}^2.$$

Sakkofunktiomenetelmä pyrkii ratkaisemaan tehtävän

$$\begin{aligned} \sup_\mu \quad & \theta(\mu) \\ \text{s. t.} \quad & \mu \geq 0, \end{aligned}$$

missä $\theta(\mu) = \inf\{f(x) + \mu\alpha(x) \mid x \geq 0\}$. Voidaan osoittaa, että kun funktiot f, g_i ja α ovat jatkuvia ja jokaista kerrointa μ kohti on olemassa funktion $\theta(\mu)$ minimoiva piste $x_\mu \geq 0$, joka kuuluu joukon $\mathbb{R}^n$ kompaktiin osajoukkoon, niin $\inf\{f(x) \mid x \geq 0,\ g_i(x) \leq 0\ \forall i\} = \sup_{\mu \geq 0} \theta(\mu) = \lim_{\mu \to \infty} \theta(\mu)$. Tällöin päästään siis mielivaltaisen lähelle tehtävän (16) optimia kasvattamalla riittävästi sakkokerrointa μ. Lisäksi, jos $\alpha(x_\mu) = 0$ jollakin kertoimella μ, niin x_μ on tehtävän sallittu ratkaisu. Suuremmilla sakkokertoimen arvoilla menetelmä päätyy yleensä nopeammin optimiin, mutta toisaalta liian suurilla sakkokertoimilla menetelmä voi ajautua laskennallisiin vaikeuksiin. Ulkopuolinen sakkofunktiomenetelmä etenee seuraavasti:

1. Aseta lopetusta varten $\varepsilon > 0$, aloituspiste x_1, sakkoparametri μ_1 ja kasvatuskerroin $\beta > 1$. Aseta $k = 1$ ja siirry vaiheeseen 2.

2. Käyttäen pistettä x_k aloituspisteenä ratkaise tehtävä

$$\begin{aligned} \min_x \quad & f(x) + \mu_k \alpha(x) \\ \text{s. t.} \quad & x \in \mathbb{R}^n. \end{aligned}$$

   Olkoon x_{k+1} kyseisen tehtävän ratkaisu. Siirry vaiheeseen 3.

3. Jos $\mu_k \alpha(x_{k+1}) < \varepsilon$, lopeta ja piste x_{k+1} on ratkaisu. Muuten aseta $\mu_{k+1} = \beta\mu_k$ ja $k \leftarrow k + 1$ ja toista vaihe 2.

Menetelmän etuna on se, ettei pisteiden x_i tarvitse olla sallittuja. Nyt on kuitenkin syytä pitää mielessä, ettei negatiivisilla strategian arvoilla ole mitään mielekästä tulkintaa; eihän mallin puitteissa kelpoisuusgradienttien laskeminen näissä pisteissä olisi edes sallittua! Ajatuksena onkin hieman naiivisti laajentaa menetelmää varten kohdefunktion määrittelyjoukkoa koskemaan myös negatiivisia strategian arvoja ja sallia kohdefunktion arvojen laskeminen näissä pisteissä. Tämä pitää muistaa tuloksia tulkittaessa.

4.2 Viivahaku

Moniulotteisen epälineaarisen rajoitteettoman optimointitehtävän osana käytetään hyvin usein yksiulotteista viivahakua. Siinä edetään pisteestä x_k jol-

lakin tavalla määrätyllä hakusuunnalla d_k ja askelpituudella λ_k uuteen iteraatiopisteeseen $x_{k+1} = x_k + \lambda_k d_k$. Askelpituudeksi valitaan se λ_k, jolla piste x_{k+1} on sallittu ja lauseke $f(x_k + \lambda d_k)$ minimoituu. Hakusuunta määrätään yleensä parantavaksi, jolloin tarkastellaan vain ei-negatiivisia askelpituuksia. Usein ei ole kuitenkaan tarpeen löytää minimiä tai se ei ole edes mahdollista, vaan riittää etsiä sellainen askelpituus, jolla funktion arvo pienenee riittävästi. Tällöin suoritetaan *epätarkka viivahaku*. Viivahaun tarkkuudesta tinkiminen voi joissakin tilanteissa nopeuttaa algoritmien suoritusta. Sopivan askelpituuden määräämiseksi sen on toteutettava tietyt ehdot.

Oletetaan jatkossa, että funktio f on differentioituva. Eräs näistä ehdoista hyväksyttävälle askelpituudelle on *riittävän pienentymisen ehto* (*Armijon ehto*)

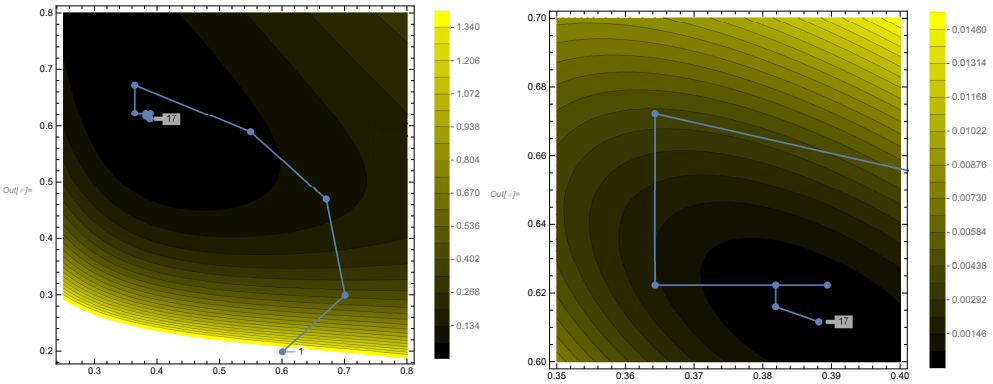
$$f(x + \lambda d) \leq \mu \lambda \nabla f(x)^T d, \tag{17}$$

missä $0 < \mu < 1$. Lisäksi voidaan konvergenssin takaamiseksi käyttää vielä *kaarevuusehtoa*

$$|\nabla f(x + \lambda d)^T d| \leq \eta |\nabla f(x)^T d|,$$

missä $0 < \mu < \eta < 1$.

Takautuvassa viivahaussa aloitetaan asetetusta askelpituudesta (yleensä $\lambda_1 = 1$) pienentäen sitä, kunnes riittävän pienentymisen ehto (17) toteutuu. Tällöin ei ole takeita kaarevuusehdon täyttymisestä eikä näin ollen konvergenssista. Menetelmä on kuitenkin varsin tehokas etenkin, jos funktion gradienttien laskeminen on laskennallisesti työlästä ja tästä syystä sitä tullaan käyttämään jatkossa. Jokaisella askeleella suoritetaan ensin neliöllinen interpolaatio ja myöhemmin tarvittaessa kuutiollisia interpolaatioita. Kussakin interpolaatiossa askelpituudeksi λ_k valitaan tämän minimi $\tilde{\lambda}$, kunhan $\tilde{\lambda} \in [c_1\lambda_{k-1}, c_2\lambda_{k-1}]$, missä $0 < c_1 \leq c_2 < 1$. Jos $\tilde{\lambda} < c_1\lambda_{k-1}$, niin uudeksi askelpituudeksi asetetaan $\lambda_k = c_1\lambda_{k-1}$. Vastaavasti, jos $\tilde{\lambda} > c_2\lambda_{k-1}$, niin uudeksi askelpituudeksi asetetaan $\lambda_k = c_2\lambda_{k-1}$. Tyypillisesti käytetään arvoja $c_1 = 0.1$ ja $c_2 = 0.5$ [3]. Neliöllisessä interpolaatiossa minimoidaan pisteisiin $f(x)$, $\nabla f(x)$ ja $f(x + \lambda_k d)$ sovitettua kvadraattista funktiota. Kuutiollisessa interpolaatiossa minimoidaan pisteisiin $f(x)$, $\nabla f(x)$, $f(x + \lambda_k d)$ ja $f(x + \lambda_{k-1} d)$ sovitettua kuutiollista funktiota.



Kuva 6: (a) Ulkopuolisen sakkofunktiomenetelmän toiminta resurssi-kuluttajamallin singulaarisen strategian ratkaisemiseksi, kun käytetään diskreettiä Hooke-Jeeves-algoritmia. Aloituspiste $(0.6, 0.2)$ ja lopetuspiste on kuvassa numeroitu. Kuvassa on esitetty myös kohdefunktion tasa-arvokäyrät. Menetelmä päätyy optimiin yhdellä ulkopuolisen sakkofunktiomenetelmän iteraatiolla, joten kuvassa esitetyt pisteet ovat diskreetin Hooke-Jeeves-menetelmän iteraatiopisteet. (b) Algoritmin generoimat pisteet lähellä optimia. Sakkofunktiomenetelmän parametrit: $\varepsilon = 0.01$, $\mu_1 = 1$, $\beta = 2$. Hooke-Jeeves-menetelmän parametrit: $\varepsilon = 0.001$, $\alpha = 0.7$, $\Delta = 0.1$. Mallin parametrit samat kuin kuvassa 1a.

Tässä tutkielmassa takautuva viivahaku on toteutettu käyttäen Mathematican FindMinimum-rutiinia Backtracking-optiolla. Siinä käytetään oletusarvoja $\mu = 1 \times 10^{-4}$, $c_1 = 0.1$ ja $c_2 = 0.5$ [12].

4.3 Hooke-Jeeves-menetelmä

Tarkastellaan ensimmäisenä optimointimenetelmänä Hooke-Jeeves-menetelmää, joka on *suora optimointimenetelmä*. Toisin sanoen tässä menetelmässä ei hyödynnetä derivaattainformaatiota, vaan ainoastaan funktion arvojen laskuja. Yleisesti ottaen suorahakumenetelmien käyttäminen ei ole kannattavaa, mikäli tehokkaampia derivaattapohjaisia menetelmiä on mahdollista käyttää. Hooke-Jeeves-menetelmä on valittu mukaan, sillä se on vertailukelpoisempi kelpoisuusgradienttimenetelmän kanssa kuin jatkossa tarkasteltavat derivaattoja käyttävät menetelmät. Erityisesti tarkastel-

laan sen diskreettiä versiota, jossa algoritmi säätelee itse askelpituutensa aivan kuten kelpoisuusgradienttimenetelmässä. Tällöin askelpituuden määräämiseksi ei tarvita viivahakua, joka voisi muodostua laskennalliseksi pullonkaulaksi.
Hooke-Jeeves-menetelmässä lähdetään ensin minimoimaan kohdefunktiota koordinaattiakselien suunnissa. Sen jälkeen suoritetaan korjausaskel, joka kulkee kahden edellisen iteraatiopisteen suuntaisesti. Tätä jatketaan niin kauan, kunnes askelpituus alittaa asetetun arvon. Diskreetti Hooke-Jeeves-algoritmi voidaan esittää seuraavasti:

1. Aseta lopetusta varten $\varepsilon > 0$. Olkoot $d_1, ..., d_n$ koordinaattiakselien suunnat, $\Delta \geq \varepsilon$ askelpituus alussa ja $\alpha > 0$ kiihdytyskerroin. Valitse aloituspiste x_1. Aseta $y_1 = x_1$ ja $j = k = 1$. Siirry vaiheeseen 2.

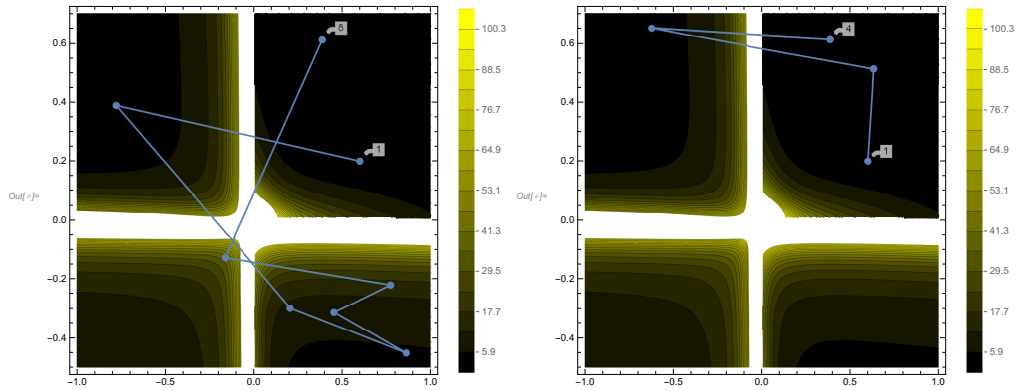
2. Jos $f(y_j + \Delta d_j) < f(y_j)$, niin yritys on *onnistunut*; aseta $y_{j+1} = y_j + \Delta d_j$ ja siirry vaiheeseen 3. Jos $f(y_j + \Delta d_j) \geq f(y_j)$, niin yritys on *epäonnistunut*. Tällöin, jos $f(y_j - \Delta d_j) < f(y_j)$, niin aseta $y_{j+1} = y_j - \Delta d_j$ ja siirry vaiheeseen 3. Muuten aseta $y_{j+1} = y_j$ ja siirry vaiheeseen 3.

3. Jos $j < n$, niin aseta $j \leftarrow j + 1$ ja siirry vaiheeseen 2. Jos $j = n$ ja $f(y_{n+1}) < f(x_k)$, niin siirry vaiheeseen 4. Muussa tapauksessa siirry vaiheeseen 5.

4. Aseta $x_{k+1} = y_{n+1}$ ja $y_1 = x_{k+1} + \alpha(x_{k+1} - x_k)$. Aseta $k \leftarrow k + 1$ ja $j = 1$ ja siirry vaiheeseen 2.

5. Jos $\Delta \leq \varepsilon$, niin lopeta ja x_k on ratkaisu. Muussa tapauksessa aseta $\Delta \leftarrow 0.5\Delta$, $y_1 = x_k$, $x_{k+1} = x_k$, $k \leftarrow k+1$ ja $j = 1$ ja siirry vaiheeseen 2.

Algoritmin suorituksen aikana on tyypillistä laskea pisteitä, jotka hylätään, jos kohdefunktion arvo ei pienene. Vaiheissa 2 ja 3 suoritetaan minimointi koordinaattiakselien suunnissa. Vaiheessa 4 muodostetaan korjausaskel, joka voidaan myöhemmin hylätä, jos sen kautta ei saada kohdefunktion arvoa pienennettyä.



Kuva 7: (a) Ulkopuolisen sakkofunktiomenetelmän toiminta resurssi-kuluttajamallin singulaarisen strategian ratkaisemiseksi, kun käytetään nopeimman laskeutumisen menetelmää ja takautuvaa viivahakua. Aloituspiste $(0.6, 0.2)$ ja lopetuspiste on kuvassa numeroitu. Kuvassa on esitetty myös kohdefunktion tasa-arvokäyrät. Maksimaaliseksi NLM-iteraatioiden määräksi kullakin iteraatiolla asetettiin 100. Sakkofunktiomenetelmän parametrit: $\varepsilon = 0.1,\ \mu_1 = 1,\ \beta = 4$. (b) Modifioitu sakkofunktiomenetelmä, jossa nopeimman laskeutumisen menetelmässä kasvatetaan sakkoparamteria aina seuraavalla iteraatiolla. Algoritmi päätyy optimiin selvästi nopeammin kuin a-kohdassa. Parametrit: $\varepsilon = 0.01,\ \mu_1 = 1,\ \beta = 5$. Mallin parametrit samat kuin kuvassa 1a.

Kuvissa 6a ja 6b on esitetty diskreetin Hooke-Jeeves-menetelmän kulku osana sakkofunktiomenetelmää aiemmin esitetyn resurssi-kuluttajamallin singulaarisen strategian ratkaisemiseksi. Algoritmi päätyy singulaariseen strategiaan $\beta_1^* \approx 0.38805$ $\beta_2^* \approx 0.611675$ sakkokerrointa kasvattamatta 16 Hooke-Jeeves-menetelmän iteraatiolla. Tämä osuu todelliseen arvoon kahden desimaalin tarkkuudella. Algoritmi on hieman herkkä syötettävien arvojen suhteen. Erityisesti liian suurilla alkuaskelpituuden arvoilla algoritmi ei konvergoi optimiin.

4.4 Nopeimman laskeutumisen menetelmä

Moniulotteisista rajoitteettomista gradienttipohjaisista optimointimenetelmistä nopeimman laskeutumisen menetelmä eli NLM on idealtaan yksinker-

taisimmasta päästä. Ajatuksena on edetä pisteestä x siihen suuntaan, jossa funktion arvo pienenee eniten. Tässä suunnassa suoritetaan sitten viivahaku. Kun funktio f on differentioituva pisteessä x, niin suuntaderivaatta suuntaan d on

$$f'(x;d) = \lim_{\lambda \to 0_+} \frac{f(x+\lambda d) - f(x)}{\lambda} = \nabla f(x)^T d.$$

Halutaan nyt etsiä sellainen suunta d, jolla suuntaderivaatta minimoituu ja lisäksi $\|d\| = 1$. Cauchy-Schwarzin epäyhtälön nojalla

$$-\|\nabla f(x)\|\|d\| \leq \nabla f(x)^T d \leq \|\nabla f(x)\|\|d\|$$

eli haluttu suunta on $d^* = -\frac{\nabla f(x)}{\|\nabla f(x)\|}$. Nopeimman laskeutumisen menetelmä kootusti on:

1. Aseta lopetusta varten $\varepsilon > 0$. Valitse aloituspiste x_1 ja aseta $k = 1$. Siirry vaiheeseen 2.

2. Jos $\|\nabla f(x)\| < \varepsilon$, lopeta. Piste x_k on tällöin ratkaisu. Muuten aseta suunnaksi $d_k = -\frac{\nabla f(x_k)}{\|\nabla f(x_k)\|}$ ja suorita viivahaku löytääksesi funktion $f(x_k + \lambda d_k)$ minimi, $\lambda \geq 0$. Aseta $x_{k+1} = x_k + \lambda_k d_k$ ja $k \leftarrow k+1$ ja toista vaihe 2.

Vaikka nopeimman laskeutumisen menetelmä on hyvin käyttökelpoinen differentioituville funktioille, se ei silti pärjää tehokkuudessaan kehittyneemmille menetelmille. Menetelmän osana voidaan käyttää myös epätarkkaa viivahakua, mutta tällöin sen konvergenssi saattaa huonontua entisestään.
Kuvassa 7a on havainnollistettu ulkopuolisen sakkofunktiomenetelmän toimintaa, kun sen osana on käytetty nopeimman laskeutumisen menetelmää takautuvalla viivahaulla. Sakkofunktiomenetelmälle ominaiseen tapaan liikutaan ei-sallitulla alueella, kunnes päädytään optimiin. Tässä tapauksessa saatu optimi $(\beta_1^*, \beta_2^*) \approx (0.387559, 0.612707)$ on oikea kahden desimaalin tarkkuudella ja se saadaan seitsemällä iteraatiolla. Sakkoparametria on pitänyt siis kasvattaa kuusi kertaa. Tarkastellun resurssi-kuluttajamallin tapauksessa algoritmi ei toimi ideaalisti, sillä se on hyvin herkkä asetettavien parametrien suhteen ja laskenta-aika on pitkä (mitattu CPU-aika yli 10 sekuntia). Lisäksi nopeimman laskeutumisen menetelmän konvergenssi epätarkalla viivahaulla on niin huono, että on syytä asettaa sille maksimaalinen

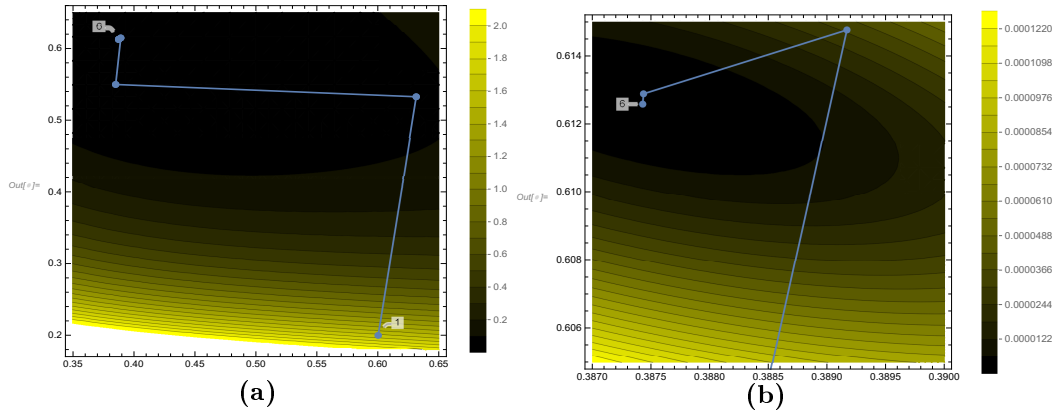
NLM-iteraatiomäärä (vaiheeseen 2) algoritmin toiminnan takaamiseksi. Tässä esimerkissä rajaksi on asetettu 100 NLM-iteraatiota. NLM-iteraatioiden määrä kullakin iteraatiolla oli {100, 100, 100, 12, 24, 100, 9}.
Kuvassa 7b on käytetty tätä tutkielmaa varten modifioitua versiota edellisestä, jossa sakkoparametria kasvatetaan jokaisen NLM-iteraation jälkeen. Modifioitu menetelmä voidaan nähdä ulkopuolisena sakkofunktiomenetelmänä, jossa kunkin iteraation aikana lasketaan vain yksi NLM-iteraatio. Nyt ulkopuolisen sakkofunktiomenetelmän lopetuskriteeri on otettu osaksi nopeimman laskeutumisen menetelmän lopetuskriteeriä (algoritmin vaihe 2). Aikaisemmin olleita konvergenssivaikeuksia ei ole samassa määrin ja algoritmi päätyy optimiin $(\beta_1^*, \beta_2^*) \approx (0.387025, 0.613600)$ kahden desimaalin tarkkuudella huomattavan paljon nopeammin (CPU-aika noin 0.1 sekuntia) kolmella iteraatiolla. Modifioitu versio ei ole peräisin kirjallisuudesta, joten sen toiminnasta yleisesti ei ole takeita. Tässä tapauksesssa se kuitenkin toimii hyvin.

4.5 Kvasi-Newton-menetelmät ja Broydenin perhe

Olkoon funktio $f : \mathbb{R}^n \to \mathbb{R}$ differentioituva pisteessä x ja $\nabla f(x)^T d < 0$ jollekin suunnalle d. Voidaan nyt kirjoittaa

$$\frac{f(x + \lambda d) - f(x)}{\lambda} = \nabla f(x)^T d + \|d\| \alpha(x; \lambda d),$$

missä $\alpha(x; \lambda d) \to 0$, kun $\lambda \to 0_+$. Tällöin on olemassa sellainen $\delta > 0$, että $f(x + \lambda d) < f(x)$ kaikilla $\lambda \in (0, \delta)$ eli vektori d on funktion f *parantava suunta* pisteessä x. Uutta hakusuuntaa määritettäessä käytetään symmetrisiä positiivisesti definiittejä matriiseja. Jos hakusuunta on $d = -B\nabla f(x)$, missä matriisi B on symmetrinen ja positiividefiniitti, niin $d^T \nabla f(x) = -\nabla f(x)^T B \nabla f(x) < 0$ eli suunta d on laskeva. Nopeimman laskeutumisen menetelmässä matriisiksi B voidaan ajatella identiteettimatriisi I ja Newtonin menetelmässä Hessen matriisin käänteismatriisi $H^{-1}(x)$. Eräs Newtonin menetelmän ongelmista on, ettei matriisi $H^{-1}(x)$ ole aina positiividefiniitti, jolloin joudutaan konvergenssivaikeuksiin.
Kvasi-Newton-menetelmät pyrkivät korjaamaan Newtonin menetelmän ongelmia estimoimalla Hessen matriisin käänteismatriisia symmetrisellä positiivisesti definiitillä matriisilla B_k, jota päivitetään joka iteraatiolla kaavalla



Kuva 8: (a) Ulkopuolisen sakkofunktiomenetelmän toiminta resurssi-kuluttajamallin singulaarisen strategian ratkaisemiseksi, kun käytetään DFP-menetelmää ja takautuvaa viivahakua. Aloituspiste $(0.6, 0.2)$ ja lopetuspiste on kuvassa numeroitu. Kuvassa on esitetty myös kohdefunktion tasa-arvokäyrät. Optimi saavutetaan yhdellä USM-iteraatiolla, joten pisteet ovat DFP-menetelmän generoimia pisteitä iteraation aikana. (b) DFP-menetelmän kulku lähempänä optimia. Sakkofunktiomenetelmän parametrit: $\varepsilon = 0.0001$, $\mu_1 = 1$, $\beta = 4$. DFP-menetelmän parametrit: $\varepsilon = 0.0001$. Mallin parametrit ovat samat kuin kuvassa 1a.

$B_{k+1} = B_k + C_k$, missä C_k on korjausmatriisi. Valitaan sellainen matriisi C_k, että kvasi-Newton-ehto

$$B_{k+1}(\nabla f(y_{k+1}) - \nabla f(y_k)) = y_{k+1} - y_k$$

on voimassa. Käytetään nyt selvyyden vuoksi merkintöjä $p_k = y_{k+1} - y_k$ ja $q_k = \nabla f(y_{k+1}) - \nabla f(y_k)$. Kvasi-Newton-ehto voidaan nyt esittää muodossa

$$C_k q_k = p_k - B_k q_k. \tag{18}$$

Niin kutsutun Broydenin perheen muodostavat matriisit

$$C_k^B = (1 - \phi) C_k^{DFP} + \phi C_k^{BFGS}, \tag{19}$$

missä

$$C_k^{DFP} = \frac{p_k p_k^T}{p_k^T q_k} - \frac{B_k q_k q_k^T B_k}{q_k^T B_k q_k} \tag{20}$$

ja

$$C_k^{BFGS} = \frac{p_k p_k^T}{p_k^T q_k}\left(1 + \frac{q_k^T B_k q_k}{p_k^T q_k}\right) - \frac{B_k q_k p_k^T + p_k q_k^T B_k}{p_k^T q_k} \tag{21}$$

ja $\phi \in \mathbb{R}$, toteuttavat ehdon (18). Erityisesti, kun valitaan yhtälössä (19) $\phi = 0$, menetelmää kutsutaan Davidon-Fletcher-Powell-menetelmäksi tai DFP-menetelmäksi. Kun $\phi = 1$, menetelmää kutsutaan Broyden-Fletcher-Goldfarb-Shanno-menetelmäksi tai BFGS-menetelmäksi. Parametrin ϕ ei tarvitsisi olla edes vakio jokaisella iteraatiolla, mutta tietyillä arvoilla matriisi B_{k+1} ei ole enää positiividefiniitti eikä menetelmä toimi odotetusti. Etenkin epätarkkaa viivahakua käytettäessä parametrin ϕ valinnalla on merkitystä algoritmin toiminnan kannalta. DFP-/BFGS-menetelmä voidaan esittää seuraavasti:

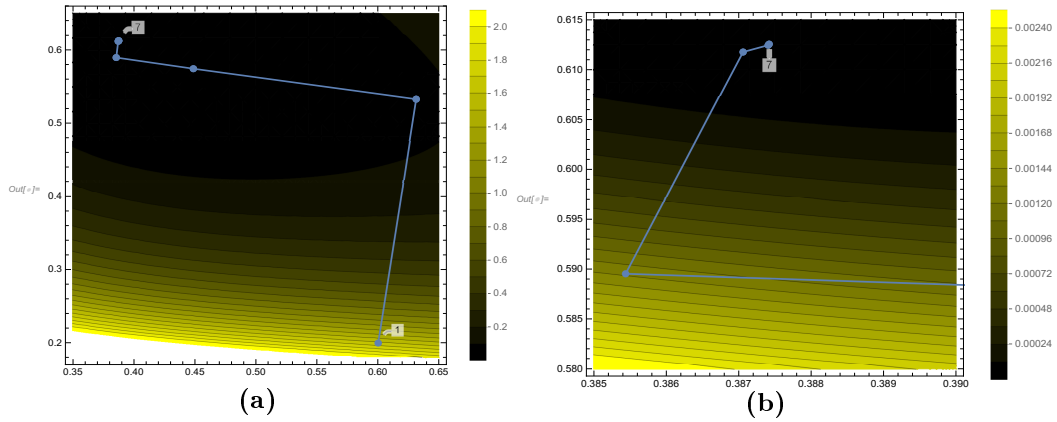
1. Aseta lopetusta varten $\varepsilon > 0$. Valitse aloituspiste x_1. Olkoon B_1 symmetrinen positiividefiniitti matriisi. Aseta $y_1 = x_1$ ja $i = k = 1$ ja siirry vaiheeseen 2.

2. Jos $\|\nabla f(y_k)\| < \varepsilon$, lopeta. Muuten laske $d_k = -B_k \nabla f(y_k)$ ja määrää λ_k, joka minimoi funktion $f(y_k + \lambda d_k)$, kun $\lambda > 0$. Olkoon uusi piste $y_{k+1} = y_k + \lambda_k d_k$. Jos $k < n$, siirry vaiheeseen 3. Jos $k = n$, aseta $y_1 = x_{i+1} = y_{n+1}$ ja $i \leftarrow i + 1$ ja $k = 1$. Toista vaihe 2.

3. Laske uusi matriisi B_{k+1} kaavasta $B_{k+1} = B_k + C_k^{DFP}$ tai $B_{k+1} = B_k + C_k^{BFGS}$, missä matriisit C_k^{DFP} ja C_k^{BFGS} ovat määritelty kaavoissa (20) ja (21). Aseta $k \leftarrow k + 1$ ja siirry vaiheeseen 2.

Vaikka molemmat menetelmät ovat hyvin tunnettuja, BFGS-menetelmä on DFP-menetelmää suorituskykyisempi ja luotettavampi ja nykyisin käytetympi.

Kuvassa 8 on esitetty DFP-menetelmän käyttöä resurssi-kuluttajamallin singulaarisen strategian ratkaisemiseksi ja kuvassa 9 vastaavasti BFGS-menetelmää käyttäen, kun ne ovat upotettu osaksi ulkopuolista sakkofunktiomenetelmää. Molemmissa tapauksissa optimi löytyy ensimmäisellä ulkopuolisen sakkofunktiomenetelmän iteraatiolla, joten kuvissa esitetyt pisteet ovat yhden iteraation aikana laskettuja pisteitä ja havainnollistavat eritoten sitä,



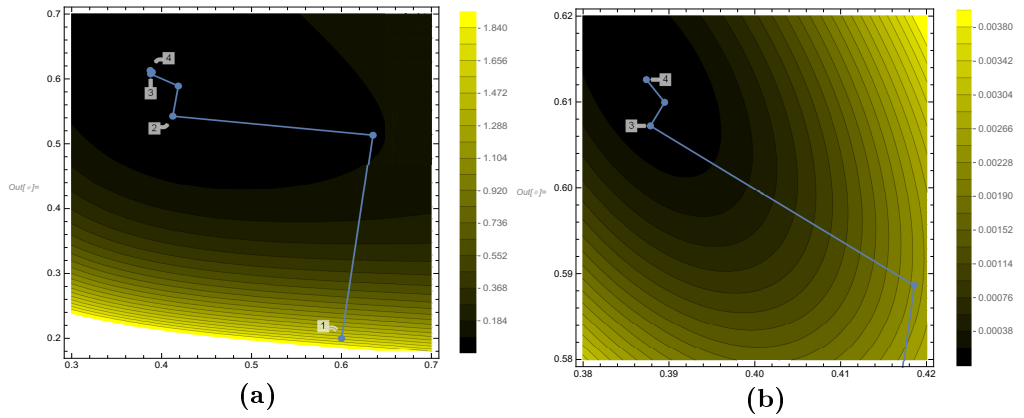
Kuva 9: (a) Ulkopuolisen sakkofunktiomenetelmän toiminta resurssikuluttajamallin singulaarisen strategian ratkaisemiseksi, kun käytetään BFGS-menetelmää ja takautuvaa viivahakua. Aloituspiste $(0.6, 0.2)$ ja lopetuspiste on kuvassa numeroitu. Kuvassa on esitetty myös kohdefunktion tasa-arvokäyrät. Optimi saavutetaan yhdellä USM-iteraatiolla, joten pisteet ovat BFGS-menetelmän generoimia pisteitä iteraation aikana. (b) BFGS-menetelmän generoimat viimeiset pisteet optimin lähellä. Sakkofunktiomenetelmän parametrit: $\varepsilon = 0.0001$, $\mu_1 = 1$, $\beta = 4$. DFP-menetelmän parametrit: $\varepsilon = 0.0001$. Mallin parametrit ovat samat kuin kuvassa 1a.

miten DFP- ja BFGS-menetelmät eroavat optimia lähestyttäessä. Kummassakin menetelmässä on valittu matriisiksi B_1 identiteettimatriisi ja käytetty takautuvaa viivahakua. Tässä esimerkissä DFP- ja BFGS-menetelmät toimivat yhtä hyvin ja löytävät optimin neljän desimaalin tarkkuudella likimain samassa ajassa (CPU-aika noin 0.2 sekuntia).

4.6 Konjugaattigradienttimenetelmät

Konjugaatti- eli liittogradienttimenetelmät pyrkivät tehostamaan nopeimman laskeutumisen menetelmää lisäämällä hakusuuntaan korjaustermin, joka huomioi edellisen hakusuunnan:

$$d_{k+1} = -\nabla f(y_{k+1}) + \alpha_k d_k. \tag{22}$$



Kuva 10: (a) Ulkopuolisen sakkofunktiomenetelmän toiminta resurssi-kuluttajamallin singulaarisen strategian ratkaisemiseksi, kun käytetään Hestenes-Stiefel-menetelmää ja takautuvaa viivahakua. Aloituspiste $(0.6, 0.2)$ ja iteraatiopisteet on kuvassa numeroitu. Kuvassa on esitetty myös kohdefunktion tasa-arvokäyrät. Optimi saavutetaan yhdellä USM-iteraatiolla, joten kuvan pisteet ovat Hestenes-Stiefel-menetelmän generoimia pisteitä iteraation aikana. (b) Hestenes-Stiefel-menetelmän generoimat viimeiset pisteet optimin lähellä. Ulkopuolisen sakkofunktiomenetelmän parametrit: $\varepsilon = 0.001$, $\mu_1 = 1$, $\beta = 2$. Hestenes-Stiefel-menetelmän parametrit: $\varepsilon = 0.001$. Mallin parametrit ovat samat kuin kuvassa 1a.

Termi α_k valitaan sellaiseksi, että hakusuunnat ovat konjugaatteja, kun kohdefunktio on kvadraattinen. Yleisesti vektoreita $d_1, ..., d_k$ sanotaan *konjugaateiksi*, jos ne ovat lineaarisesti riippumattomia ja

$$d_i^T H d_j = 0, \quad \forall\, i \neq j, \tag{23}$$

missä H on jokin symmetrinen $n \times n$-matriisi. Kun $\alpha_k \geq 0$, niin yhtälö (22) voidaan esittää muodossa

$$d_{k+1} = \tau^{-1}[-\tau \nabla f(y_{k+1}) + (1 - \tau) d_k],$$

missä $\tau = (1 + \alpha_k)^{-1}$. Täten uusi suunta voidaan nähdä nopeimmin laskeutuvan suunnan ja edellisen suunnan konveksina kombinaationa.

Tarkastellaan kvadraattista funktiota

$$f(x) = \frac{1}{2} x^T H x + c^T x,$$

missä matriisi H on symmetrinen ja positiividefiniitti. Tämän gradientti on

$$\nabla f(x) = Hx + c.$$

Käytetään jälleen merkintöjä $p_k = y_{k+1} - y_k = \lambda_k d_k$ ja $q_k = \nabla f(y_{k+1}) - \nabla f(y_k)$. Ehdosta (23) saadaan

$$d_{k+1}^T H p_k = d_{k+1}^T H(y_{k+1} - y_k) = d_{k+1}^T (Hy_{k+1} + c - (Hy_k + c)) = d_{k+1}^T q_k = 0.$$

Kun nyt kerrotaan yhtälö (22) termillä q_k ja ratkaistaan α_k, niin saadaan

$$\alpha_k = \frac{\nabla f(y_{k+1})^T q_k}{d_k^T q_k} = \frac{\lambda_k \nabla f(y_{k+1})^T q_k}{p_k^T q_k}. \tag{24}$$

Tätä kerrointa käytettäessä menetelmää kutsutaan *Hestenes-Stiefel-menetelmäksi*. Menetelmää voidaan käyttää myös tilanteissa, joissa funktio ei ole kvadraattinen, kunhan se käyttäytyy lokaalisti kvadraattisen funktion tavoin [1]. Voidaan johtaa myös muita α_k-kertoimia käytettäessä tarkkaa viivahakua, mutta ne ohitetaan tässä. Kerroin (24) on valittu konjugaattigradienttimenetelmän toteutuksessa, sillä käytetään epätarkkaa viivahakua. Konjugaattigradienttimenetelmä Hestenes-Stiefel-muodossa etenee seuraavasti:

1. Aseta lopetusta varten $\varepsilon > 0$ ja valitse aloituspiste x_1. Aseta $y_1 = x_1$, $d_1 = -\nabla f(y_1)$ ja $i = k = 1$ ja siirry vaiheeseen 2.

2. Jos $\|\nabla f(y_k)\| < \varepsilon$, lopeta. Muussa tapauksessa olkoon λ_k sellainen, että se on saatu funktion $f(y_k + \lambda d_k),\ \lambda > 0$, minimoinnissa käytettäessä epätarkkaa viivahakua. Aseta $y_{k+1} = y_k + \lambda_k d_k$. Jos $k < n$, siirry vaiheeseen 3. Muuten siirry vaiheeseen 4.

3. Aseta uudeksi suunnaksi $d_{k+1} = -\nabla f(y_{k+1}) + \alpha_k d_k$, missä kerroin α_k on kaavan (24) mukainen. Aseta $k \leftarrow k + 1$ ja siirry vaiheeseen 2.

4. Aseta $y_1 = x_{i+1} = y_{n+1}$ ja $d_1 = -\nabla f(y_1)$. Aseta $k = 1$ ja $i \leftarrow i + 1$ ja siirry vaiheeseen 2.

Konjugaattigradienttimenetelmät eivät yleisesti ole yhtä tehokkaita kuin kvasi-Newton-menetelmät, mutta niiden muistitarve on pienempi. Siksi ne

ovat käyttökelpoisempia suurten tehtävien ratkaisemisessa, kun $n > 100$ [1]. Kuvissa 10a ja 10b on havainnollistettu Hestenes-Stiefel-menetelmän toimintaa osana ulkopuolista sakkofunktiomenetelmää. Jälleen optimi löytyy yhdellä sakkofunktiomenetelmän iteraatiolla, joten kuvassa esitetyt pisteet ovat Hestenes-Stiefel-menetelmän generoimia pisteitä yhden iteraation aikana. Tässä tapauksessa algoritmi päätyy optimiin neljän desimaalin tarkkuudella CPU-aikana noin 0.4 sekuntia. Tämä on likimain kaksinkertainen aiemmin esiteltyjen kvasi-Newton-menetelmien käyttämään aikaan.

4.7 Sallittujen suuntien menetelmät

Sallittujen suuntien menetelmissä siirrytään sallitusta pisteestä kohdefunktion kannalta parempaan sallittuun pisteeseen ottamalla rajoitteet huomioon jo hakusuuntaa määrättäessä. Suuntaa d sanotaan *parantavaksi sallituksi suunnaksi* sallitussa pisteessä x, jos on olemassa sellainen $\delta > 0$, että $f(x + \lambda d) < f(x)$ ja piste $x + \lambda d$ on sallittu kaikilla $\lambda \in (0, \delta)$. Sallittujen suuntien menetelmät soveltuvat erityisesti tehtäville, joissa ovat lineaariset rajoitteet. Epälineaarisilla rajoitteilla on kiinnitettävä enemmän huomiota siihen, ettei ajauduta sallitun alueen ulkopuolelle. Koska yleisesti ei voida olettaa, että mallien rajoitteet ovat lineaarisia, niin esitellään Zoutendijkin menetelmä vain epälineaaristen rajoitteiden tapauksessa. Ennen varsinaista algoritmin läpikäyntiä on syytä esitellä sitä varten kaksi teoreettista tulosta.

4.7.1 Gordanin lemma

Esitellään ensin konveksin analyysin keskeinen tulos, joka tunnetaan nimellä Gordanin lemma. Se on tärkeä johdettaessa myöhemmin epälineaarisen optimoinnin optimaalisuusehtoja.
Olkoon A $m \times n$-matriisi. Gordanin lemman nojalla tarkalleen toinen systeemeistä

$$\text{Systeemi 1:} \quad Ax < 0,\ x \in \mathbb{R}^n$$
$$\text{Systeemi 2:} \quad A^T p = 0,\ p \geq 0,\ p \in \mathbb{R}^m,\ p \neq 0$$

on ratkeava (tulkitaan komponenteittain).
Gordanin lemman todistus etenee seuraavasti: Oletetaan ensin, että systee-

millä 1 on ratkaisu. Tällöin on olemassa jokin vektori $x \in \mathbb{R}^n$, jolle $Ax < 0$. Tehdään vastaoletus, että myös systeemillä 2 on ratkaisu eli on olemassa jokin nollasta poikkeava ei-negatiivinen vektori $p \in \mathbb{R}^m$, jolle $A^T p = 0$ ja täten $x^T A^T p = 0$. Tästä saadaan ristiriita, sillä systeemin 1 ratkeavuuden nojalla pitäisi olla $p^T Ax = x^T A^T p < 0$. Siis systeemi 2 ei ole ratkeava.
Oletetaan sitten, ettei systeemillä 1 ole ratkaisua. Tarkastellaan joukkoja $S_1 = \{z \mid z = Ax,\ x \in \mathbb{R}^n\}$ ja $S_2 = \{z \mid z < 0\}$. Koska joukot S_1 ja S_2 ovat epätyhjiä konvekseja joukkoja ja $S_1 \cap S_2 = \emptyset$, niin on olemassa joukot erottava hypertaso [1]. Toisin sanoen on olemassa sellainen nollasta poikkeava vektori $p \in \mathbb{R}^m$, että $p^T z \leq p^T Ax$ jokaisella $x \in \mathbb{R}^n,\ z \in \mathrm{cl}(\mathrm{S}_2)$, missä $\mathrm{cl}(\mathrm{S}_2)$ on joukon S_2 sulkeuma. Koska jokainen vektorin z komponentti voidaan asettaa mielivaltaisen pieneksi, niin pitää olla $p \geq 0$. Asettamalla $z = 0$ saadaan $p^T Ax \geq 0$ jokaisella $x \in \mathbb{R}^n$. Erityisesti valitsemalla $x = -A^T p$ saadaan $0 \leq p^T Ax = -\|A^T p\|^2$. Siispä $A^T p = 0$ ja systeemillä 2 on ratkaisu.

4.7.2 Fritz-John-ehdot

Johdatellaan seuraavaksi Fritz-John-ehdot, jotka ovat yksi välttämättömistä optimaalisuusehdoista. Tarkastellaan minimointitehtävää (16). Jos $\hat{x}$ on tämän tehtävän lokaali ratkaisu, niin ei ole sellaista vektoria $d \in \mathbb{R}^n$, jolle $\nabla f(\hat{x})^T d < 0$ ja $\nabla g_i(\hat{x})^T d < 0$ kaikilla $i \in I$, missä $I = \{i \mid g_i(\hat{x}) = 0\}$ on aktiivisten indeksien joukko pisteessä $\hat{x}$ (Tämä nähdään johdettaessa Zoutendijkin menetelmää). Olkoon matriisin A vaakariveinä vektorit $\nabla f(\hat{x})^T$ ja $\nabla g_i(\hat{x})^T$ kaikilla $i \in I$. Tällöin ei siis ole sellaista vektoria d, että systeemi $Ad < 0$ on ratkeava. Gordanin lemman nojalla on olemassa vektori $p \geq 0,\ p \neq 0$, jolle $A^T p = 0$. Merkitään vielä vektorin p komponentteja symboleilla u_0 ja u_i, missä $i \in I$, jolloin voidaan nyt muotoilla Fritz-John-ehdot seuraavasti:
Olkoon $\hat{x}$ tehtävän (16) sallittu piste ja merkitään aktiivisten indeksien joukkoa $I = \{i \mid g_i(\hat{x}) = 0\}$ kyseisessä pisteessä. Oletetaan, että funktiot f ja g_i kaikilla $i \in I$ ovat differentioituvia pisteessä $\hat{x}$. Oletetaan lisäksi, että funktiot g_i kaikilla $i \notin I$ ovat jatkuvia pisteessä $\hat{x}$. Jos $\hat{x}$ on tehtävän (16) lokaali

ratkaisu, niin on olemassa sellaiset vakiot u_0 ja u_i kaikilla $i \in I$, että

$$u_0 \nabla f(\hat{x}) + \sum_{i \in I} u_i \nabla g_i(\hat{x}) = 0$$
$$u_0, u_i \geq 0, \ i \in I$$
$$(u_0, u_l) \neq (0, 0),$$

missä u_l on vektori, joka koostuu komponenteista u_i, $i \in I$.
Pistettä $\hat{x}$ kutsutaan *Fritz-John-pisteeksi*. On huomionarvoista, että välttämättömät Fritz-John-ehdot eivät takaa, että Fritz-John-piste olisi optimi. Ehdoista saadaan riittävät, jos kohde- ja rajoitefunktioista tehdään lisäoletuksia, mutta niitä ei käsitellä tässä.

4.7.3 Zoutendijkin menetelmä

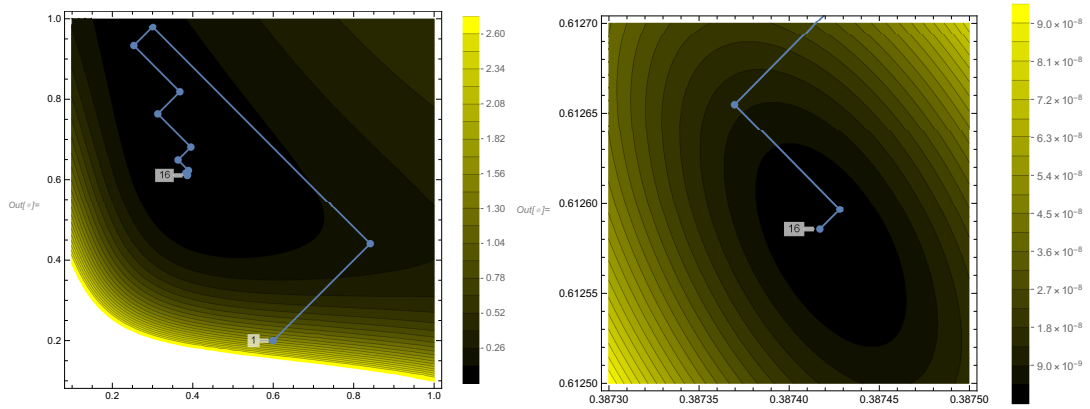
Olkoon minimoitavana tehtävä (16), missä rajoitteet g_i voivat olla epälineaarisia. Oletetaan, että $\nabla f(x)^T d < 0$ ja $\nabla g_i(x)^T d < 0$, $i \in I$, jollekin vektorille d. Kun $i \notin I$, niin $g_i(x) < 0$ ja funktion g_i jatkuvuuden nojalla $g_i(x + \lambda d) \leq 0$ riittävän pienelle askelpituudelle $\lambda > 0$. Funktion g_i,$i \in I$, differentioituvuuden nojalla

$$g_i(x + \lambda d) = g_i(x) + \lambda \nabla g_i(x)^T d + \lambda \|d\| \alpha(x; \lambda d),$$

missä $\alpha(x; \lambda d) \to 0$, kun $\lambda \to 0$. Koska $\nabla g_i(x)^T d < 0$, niin $g_i(x + \lambda d) < g_i(x) = 0$, kun $\lambda > 0$ on riittävän pieni. Täten piste $x + \lambda d$ on sallittu, kun askelpituus $\lambda > 0$ valitaan tarpeeksi pieneksi. Vastaavasti todetaan, että $f(x + \lambda d) < f(x)$, kun $\lambda > 0$ on riittävän pieni. Riittävä ehto parantavalle sallitulle suunnalle d on siis, että sen on toteutettava epäyhtälöt

$$\begin{aligned} &\nabla f(x)^T d < 0 \\ &\nabla g_i(x)^T d < 0, i \in I. \end{aligned} \tag{25}$$

Suunnan d määräämiseksi minimoidaan arvojen $\nabla f(x)^T d$ ja $\nabla g_i(x)^T d$ maksimia, kun $i \in I$. Asetetaan lisäksi normeerausrajoitukset suunnalle d, jotta vältyttäisiin rajoittamattomalta ratkaisulta. On siis ratkaistavana lineaari-



(a) **(b)**

Kuva 11: (a) Zoutendijkin menetelmän kulku resurssi-kuluttajamallin singulaarista strategiaa ratkaistaessa. Menetelmän tarkempi kuvaus tekstissä. Aloituspiste $(0.6, 0.2)$ ja lopetuspiste on kuvassa numeroitu. Kuvassa on esitetty myös kohdefunktion tasa-arvokäyrät. (b) Zoutendijkin menetelmän generoimat viimeiset pisteet optimin lähellä. Parametrit: $\varepsilon = 0.0001$. Mallin parametrit samat kuin kuvassa 1a.

nen optimointitehtävä

$$\begin{aligned} \min \quad & z \\ \text{s. t.} \quad & \nabla f(x)^T d - z < 0 \\ & \nabla g_i(x)^T d - z < 0, \ i \in I \\ & -1 \leq d_j \leq 1, \ j = 1, ..., n. \end{aligned} \tag{26}$$

Jos sen ratkaisuna on $(\hat{z}, \hat{d})$, jossa $\hat{z} < 0$, niin suunta $\hat{d}$ on tällöin parantava sallittu suunta. Toisaalta $\hat{z} = 0$, jos ja vain jos systeemille (25) ei löydy ratkaisua. Tämä taas on Gordanin lemman nojalla ekvivalenttia sille, että on olemassa sellaiset kertoimet u_0 ja u_i jokaiselle $i \in I$, että

$$u_0 \nabla f(x) + \sum_{i \in I} u_i \nabla g_i(x) = 0,$$

missä kertoimista $u_0, u_i \geq 0$ vähintään yhden on poikettava nollasta. Nämä ovat juuri Fritz-Johnin optimaalisuusehdot. Toisin sanoen tehtävän (26) ratkaisussa $z = 0$, jos ja vain jos x on Fritz-John-piste.
Esitetään nyt Zoutendijkin menetelmä kootusti.

1. Aseta lopetusta varten $\varepsilon > 0$. Valitse aloituspiste x_1, jolle $g_i(x_1) \leq 0,\ i = 1, ..., m$. Aseta $k = 1$.

2. Ratkaise tehtävä (26) pisteessä x_k. Olkoon ratkaisu (z_k, d_k). Jos $z_k \geq -\varepsilon$, niin lopeta. Tällöin x_k on Fritz-John-piste.

3. Olkoon λ_k ratkaisu yksiulotteiselle optimointitehtävälle

$$\begin{aligned} \min \quad & f(x_k + \lambda d_k) \\ \text{s. t.} \quad & 0 \leq \lambda \leq \lambda_{max}, \end{aligned}$$

   missä $\lambda_{max} = \sup\{\lambda \mid g_i(x_k + \lambda d_k) \leq 0, i = 1, ..., m\}$. Olkoon $x_{k+1} = x_k + \lambda_k d_k$. Aseta $k \leftarrow k + 1$ ja siirry vaiheeseen 2.

Kuvassa 11 on esitetty sovellutus Zoutendijkin menetelmästä resurssikuluttajamallin singulaarisen strategian ratkaisemiseksi. Menetelmään on otettu vain rajoitusfunktio

$$g(\beta_1, \beta_2) = \frac{1}{K_1\beta_1 + K_2\beta_2} - \lambda(\beta_1, \beta_2).$$

Ei-negatiivisuusrajoitusten toteutuminen voidaan tarkistaa helposti kuvasta. Suunnanhakutehtävässä on käytetty Simplex-menetelmää. Aikaisemmin toteutettua takautuvaa viivahakua FindMinimum-rutiinilla ei käytetä, sillä Backtracking-optio soveltuu vain rajoittamattomille viivahakutehtäville. Askelpituuden ratkaisemiseksi on algoritmin vaiheessa 3 käytetty Mathematican Minimize-rutiinia, jossa maksimiaskelpituutena on $\lambda_{max} = 1$ mikäli

$$g(\beta_1 + \lambda_{max} d_1, \beta_2 + \lambda_{max} d_2) \leq 0. \tag{27}$$

Muussa tapauksessa maksimiaskelpituutta puolitetaan niin kauan, että epäyhtälö (27) on voimassa. Menetelmän generoimat peräkkäiset hakusuunnat ovat kohtisuorassa toisiaan vastaan, mikä ei ole konvergenssin kannalta hyvä asia. Tässä algoritmi päätyy aloituspisteestä $(0.6, 0.2)$ optimiin neljän desimaalin tarkkuudella 15 iteraatiolla. Menetelmä ei kuitenkaan yllä tehokkaimpien joukkoon (CPU-aika noin 5 sekuntia), mutta suoriutuu kuitenkin selvästi nopeammin kuin ulkopuolinen sakkofunktiomenetelmä käytettäessä nopeimman laskeutumisen menetelmää takautuvalla viivahaulla (katso liite A).

4.8 Toistettu kvadraattinen optimointi (SQP)

Kuten aiemmin havaittiin, Zoutendijkin menetelmän konvergenssi ei ole ihanteellinen, sillä menetelmässä käytetään ensimmäisen asteen approksimaatiota. Sen sijaan toistetussa kvadraattisessa optimoinnissa (tai peräkkäisissä kvadraattisissa approksimaatioissa) eli SQP-menetelmässä käytetään toisen asteen approksimaatiota. Nimensä mukaisesti SQP-menetelmien ideana on muuntaa epälineaarinen optimointitehtävä jonoksi helpommin ratkaistavia kvadraattisia optimointitehtäviä, joissa on kvadraattinen kohdefunktio ja lineaariset rajoitukset. Menetelmässä olennaisena osana ovat KKT-ehdot, joten esitellään ne ensin.

4.8.1 Karush-Kuhn-Tuckerin ehdot

Karush-Kuhn-Tuckerin ehdot eli KKT-ehdot ovat toinen tunnettu esitys välttämättömille optimaalisuusehdoille. Erona aikaisempiin Fritz-John-ehtoihin on, että nyt vaaditaan lisäksi aktiivisille rajoitteille gradienttivektoreiden lineaarista riippumattomuutta tarkastellussa pisteessä. KKT-ehdot voidaan johtaa suoraan Fritz-John-ehdoista.
Oletetaan siis, että Fritz-John-ehdot ja riippumattomuusoletus ovat voimassa. Tällöin on olemassa vakiot u_0 ja $\hat{u}_i$ kaikilla $i \in I$ joille pätee

$$\begin{aligned} u_0 \nabla f(\hat{x}) + \sum_{i \in I} \hat{u}_i \nabla g_i(\hat{x}) &= 0 \\ u_0, \hat{u}_i &\geq 0, \ i \in I \\ (u_0, \hat{u}_l) &\neq (0, 0), \end{aligned}$$

missä $\hat{u}_l$ on vektori, joka koostuu komponenteista $\hat{u}_i, \ i \in I$. Koska gradientit $\nabla g_i(\hat{x}), \ i \in I$ ovat lineaarisesti riippumattomia, niin $u_0 > 0$. Tällöin voidaan jakaa yhtälössä kertoimet luvulla u_0 ja merkitä $u_i := \frac{\hat{u}_i}{u_0}$. Esitetään nyt nämä ehdot kokonaisuudessaan.
Olkoon $\hat{x}$ tehtävän (16) sallittu piste ja $I = \{i \mid g_i(\hat{x}) = 0\}$ aktiivisten indeksien joukko. Oletetaan, että funktiot f ja g_i kaikilla $i \in I$ ovat differentioituvia pisteessä $\hat{x}$ ja että funktiot g_i ovat jatkuvia pisteessä $\hat{x}$ kaikilla $i \notin I$. Oletetaan lisäksi, että vektorit $\nabla g_i(\hat{x})$ ovat lineaarisesti riippumattomia kai-

killa $i \in I$. Jos $\hat{x}$ on tehtävän (16) lokaali ratkaisu, niin on olemassa sellaiset vakiot u_i kaikilla $i \in I$, että

$$\nabla f(\hat{x}) + \sum_{i \in I} u_i \nabla g_i(\hat{x}) = 0$$
$$u_i \geq 0,\ i \in I.$$

Jos näiden oletusten lisäksi, funktiot g_i kaikilla $i \notin I$ ovat myös differentioituvia pisteessä $\hat{x}$, niin KKT-ehdot voidaan esittää muodossa

$$\nabla f(\hat{x}) + \sum_{i=1}^{m} u_i \nabla g_i(\hat{x}) = 0$$
$$u_i g_i(\hat{x}) = 0,\ i = 1, ..., m$$
$$u_i \geq 0,\ i = 1, ..., m$$

valitsemalla $u_i = 0$ kaikilla $i \notin I$.
Kertoimia u_i kutsutaan Lagrangen kertoimiksi tai KKT-kertoimiksi ja pistettä $\hat{x}$ KKT-pisteeksi. Jälleen on hyvä huomata, ettei KKT-piste välttämättä ole optimi (voi olla esimerkiksi satulapiste). Myös KKT-ehdoista saadaan riittävät tietyin kohde- ja rajoitefunktio-oletuksin, mutta ne ohitetaan tässä.

4.8.2 Alkeellinen SQP-menetelmä (RSQP)

Olkoon edelleen ratkaistavana epäyhtälörajoitteita sisältävä tehtävä (16). Tehtävän KKT-ehdot vektorimuodossa ovat

$$\begin{aligned} \nabla f(x) + \nabla g(x)^T u &= 0 \\ g(x) &\leq 0 \\ u^T g(x) &= 0 \\ u &\geq 0, \end{aligned} \tag{28}$$

missä $m \times n$-matriisi $\nabla g(x)$ on rajoitteiden Jacobin matriisi pisteessä x. Toisin sanoen matriisi $\nabla g(x)$ koostuu riveistä $\nabla g_i(x)^T$, missä $i = 1, ..., m$. Muodostetaan sitten ehdoista (28) lineaarinen approksimaatio pisteessä (x_k, u_k).

Saadaan

$$\begin{aligned} \nabla L(x_k) + \nabla^2 L(x_k)(x - x_k) + \nabla g(x_k)^T (u - u_k) &= 0 \\ g(x_k) + \nabla g(x_k)(x - x_k) &\leq 0 \\ (u_k + (u - u_k))^T g(x_k) + u_k^T \nabla g(x_k)(x - x_k) &= 0 \\ u &\geq 0, \end{aligned} \tag{29}$$

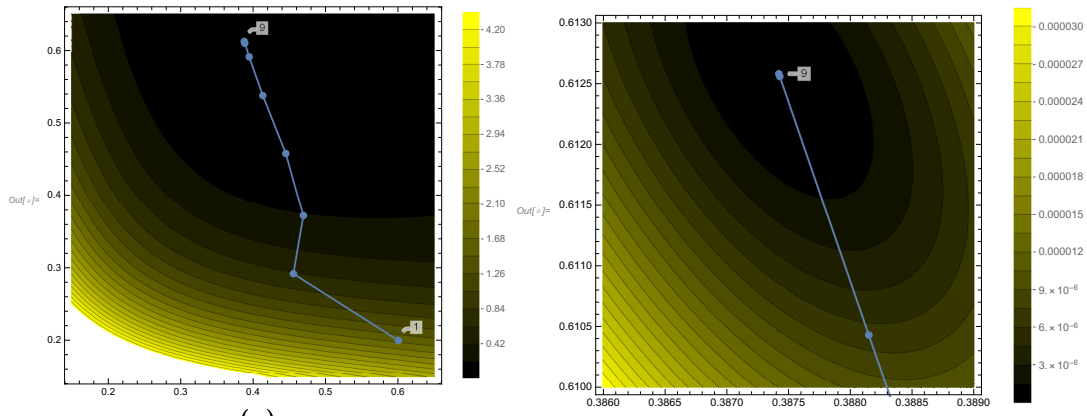
missä $\nabla L(x_k) = \nabla f(x_k) + \nabla g(x_k)^T u_k$ on Lagrangen funktion gradientti muuttujan x suhteen ja $\nabla^2 L(x_k) = \nabla^2 f(x_k) + \nabla^2 g(x_k)^T u_k$ on Lagrangen funktion Hessen matriisi muuttujan x suhteen, kun käytetään merkintää $\nabla^2 g(x_k) = (\nabla^2 g_1(x_k), ..., \nabla^2 g_m(x_k))^T$. Jos tarkastellaan termiä $(u - u_k)^T \nabla g(x_k)(x - x_k)$, niin huomataan, että sen suuruusluokka on hyvin pieni muihin verratuna, kun erotukset $(u - u_k)$ ja $(x - x_k)$ ovat lähellä nollaa. Näin pitäisi käydä, kun ollaan lähellä optimia. Kun lisätään ehtojen (29) viimeiseen yhtälöön kyseinen termi ja merkitään $d = x - x_k$, ehdot tulevat muotoon

$$\begin{aligned} \nabla f(x_k) + \nabla^2 L(x_k) d + \nabla g(x_k)^T u &= 0 \\ g(x_k) + \nabla g(x_k) d &\leq 0 \\ u^T (g(x_k) + \nabla g(x_k) d) &= 0 \\ u &\geq 0. \end{aligned} \tag{30}$$

Yhtälöryhmästä voitaisiin ratkaista $(d, u) = (d_k, u_{k+1})$ olettaen, että ratkaisu olisi olemassa. Sen jälkeen asetettaisiin $x_{k+1} = x_k + d_k$ ja prosessia toistettaisiin, kunnes $d = 0$. Tällöin kyseessä olisi tehtävän (16) KKT-piste. Epäyhtälöistä johtuen yhtälöryhmän ratkaiseminen on kuitenkin haastavaa.
Tarkastellaan sen sijaan kvadraattista tehtävää

$$\begin{aligned} \min_d \quad & f(x_k) + \nabla f(x_k)^T d + \frac{1}{2} d^T \nabla^2 L(x_k) d \\ \text{s. t.} \quad & g(x_k) + \nabla g(x_k) d \leq 0. \end{aligned} \tag{31}$$

Huomataan, että ehdot (30) ovat juurikin kyseisen kvadraattisen tehtävän KKT-ehdot. Tällöin voidaan yhtälöryhmän ratkaisemisen sijaan minimoida kvadraattinen tehtävä (31), jossa ovat alkuperäisen tehtävän (16) rajoitteet linearisoituna pisteessä x_k. Menetelmässä muodostetaan jono $\{(x_k, u_k)\}$, joka lähenee KKT-ratkaisua $(\hat{x}, \hat{u})$. Kullakin iteraatiolla ratkaistaan tehtävän (31)



(a) **(b)**

Kuva 12: (a) RSQP-menetelmän kulku resurssi-kuluttajamallin singulaarista strategiaa ratkaistaessa. Menetelmän tarkempi kuvaus tekstissä. Aloituspiste $(0.6, 0.2)$ ja lopetuspiste on kuvassa numeroitu. Kuvassa on esitetty myös kohdefunktion tasa-arvokäyrät. (b) RSQP-menetelmän generoimat viimeiset pisteet optimin lähellä. Parametrit: $\varepsilon = 0.001$. Mallin parametrit samat kuin kuvassa 1a.

primaali-duaaliratkaisu (d_k, u_{k+1}) ja asetetaan $x_{k+1} = x_k + d_k$.
Alkeellinen SQP-menetelmä (RSQP) vaiheittain on:

1. Aseta lopetusta varten $\varepsilon > 0$. Valitse sallittu primaali-duaaliratkaisu (x_1, u_1). Aseta $k = 1$ ja siirry vaiheeseen 2.

2. Ratkaise tehtävä (31). Olkoon sen ratkaisu (d_k, u_{k+1}). Siirry vaiheeseen 3.

3. Jos $\|d_k\| < \varepsilon$, niin lopeta. Tällöin piste (x_k, u_{k+1}) on tehtävän (16) KKT-ratkaisu. Muussa tapauksessa aseta $x_{k+1} = x_k + d_k$ ja $k \leftarrow k + 1$ ja siirry vaiheeseen 2.

Menetelmässä oletetaan, että tehtävällä (31) on ratkaisu. Aloitettaessa tarpeeksi läheltä optimia menetelmän konvergenssi on kvadraattista. Tällöin on olemassa sellainen luku $M > 0$, että

$$\|x_{k+1} - x^*\| \leq \beta \|x_k - x^*\|^2$$

kaikilla $k \geq M$, missä $\beta < \infty$. Menetelmällä on kuitenkin myös heikkoutensa. Globaalisti konvergoivan menetelmän konvergenssi ei riipu lähtöpisteestä. RSQP-menetelmä ei kuitenkaan ole globaalisti konvergoiva eikä menetelmällä välttämättä päästä optimiin, jos lähtöpiste on liian kaukana siitä. Myös Lagrangen funktion Hessen matriisin laskeminen on laskennallisesti vaativa prosessi eikä saatava matriisi ole välttämättä aina positiividefiniitti. Näihin ongelmiin on kuitenkin kehitetty parannuksia, joita voi tarvittaessa käyttää. Kuvissa 12a ja 12b on havainnollistettu RSQP-menetelmän kulkua resurssi-kuluttajamallin singulaarisen strategian ratkaisemiseksi. Menetelmä suoriutuu tehokkaasti päätyen optimiin kahdeksan desimaalin tarkkuudella kahdeksalla iteraatiolla (CPU-aika noin 0.5 sekuntia), kun aloituspisteenä on $(0.6, 0.2, 0, 0, 0)$. Kvadraattisten tehtävien ratkaisemisessa on käytetty Mathematican QuadraticOptimization-rutiinia. Algoritmi konvergoi kohti optimia selvästi nopeammin kuin Zoutendijkin menetelmässä (katso liite A).

5 Yhteenveto

Tutkielmassa johdettiin ja havainnollistettiin eräitä numeerisia menetelmiä resurssi-kuluttajamallin singulaarisen strategian ratkaisemiseksi. Menetelmille oli yhteistä se, että ne hyödynsivät kelpoisuusfunktion differentioituvuutta. Luvussa 3 esitelty kelpoisuusgradienttimenetelmä hyödynsi mallin kelpoisuusgradientteja. Menetelmän etuna on, että sillä voidaan myös haluttaessa selvittää mihin tilaan populaatio ajautuu mahdollisen singulaarisen strategian lisäksi. Luvussa 4 singulaarisen strategian ratkaiseminen muunnettiin rajoitteiseksi optimointitehtäväksi ja hyödynnettiin muodostettujen kohdeja rajoitefunktioiden differentioituvuutta, erityisesti kelpoisuusgradienttien differentioituvuutta.

Yksi tutkielman tavoitteista oli myös vertailla näiden menetelmien käyttökelpoisuutta mallin singulaarisen strategian ratkaisemiseen. Erityisesti menetelmät, jotka ovat hyvin epästabiileja alkuparametrien suhteen eivät ole hyviä. Tällainen oli esimerkiksi ulkopuolinen sakkofunktiomenetelmä, jossa käytettiin nopeimman laskeutumisen menetelmää epätarkalla viivahaulla rajoitteettoman optimointitehtävän ratkaisemiseen.

Stabiiliuden lisäksi menetelmän tehokkuus on ratkaisevana tekijänä. Sen vuoksi menetelmien yhteydessä esitettiin mitatut CPU-ajat ja tarkkuus, jolla ratkaisu vastasi singulaarista strategiaa. Tietysti menetelmän käyttökelpoisuuteen vaikuttaa myös se, miten helposti sen voi muodostaa ja muokata omia käyttötarkoituksia varten. Esimerkiksi kelpoisuusgradienttimenetelmässä pistetulon käyttö askelpituusehdon toteuttamisessa osottautuu hyvin käyttökelpoiseksi, sillä näin toteutettuna algoritmista tulee tehokkaampi. Saaduista tuloksista voidaan päätellä, että yksinkertaisista menetelmistä ulkopuolinen sakkofunktiomenetelmä DFP- tai BFGS-menetelmällä varustettuna on erittäin käyttökelpoinen tehokkuutensa ansiosta. Jos mallissa on vain lineaarisia rajoiteita, tulee Zoutendijkin menetelmä lineaarisilla rajoitteilla varteenotettavaksi vaihtoehdoksi. Tätä menetelmää ei kuitenkaan käsitelty tutkielmassa.

Jollei menetelmän yksinkertaisuus ole prioriteettinä, niin käytetty RSQP-menetelmä on myös suositeltava, mutta sen alitehtävien ratkaisemiseksi valittava QP-algoritmi on tietysti olennainen RSQP-menetelmän tehokkuuden kannalta. Kvadraattisesta optimoinnista olisi jo itsessään saanut tutkielmaan ihan oman lukunsa, joten käytetyssä RSQP-menetelmässä hyödynnettiin Mathematican omaa kvadraattisen optimointitehtävän ratkaisurutiinia, joka valitsee automaattisesti tilanteeseen parhaiten sopivan menetelmän.

Kirjallisuutta

[1] Bazaara, M. S., Sherali, H. D., Shetty, C. M. (2006): *Nonlinear Programming: Theory and Algorithms*, 3. painos. John Wiley & Sons, New Jersey.

[2] Bonnans, J. F., Gilbert, J. C., Lemaréchal, C., Sagastizábal, C. A. (2006): *Numerical Optimization*, 2. painos. Springer-Verlag, Berlin.

[3] Dennis, J. E., Schnabel, R. B. (1983): *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*, Prentice-Hall, New Jersey.

[4] Geritz, S. A. H., Gyllenberg, M., Jacobs, F. J. A., Parvinen, K. (2002): Invasion dynamics and attractor inheritance. *Journal of Mathematical Biology*. 44, 548-560.

[5] Geritz, S. A. H., Kisdi, É. (2004): On the mechanistic underpinning of discrete-time population models with complex dynamics. *Journal of Theoretical Biology*. 228, 261–269.

[6] Haataja, J. (2004): *Optimointitehtävien ratkaiseminen*, 3. uudistettu painos. CSC-Tieteellinen laskenta Oy, Helsinki.

[7] Leimar, O. (2001): Evolutionary change and Darwinian demons. *Selection*. 2, 65–72.

[8] Leimar, O. (2009): Multidimensional convergence stability. *Evolutionary Ecology Research*. 11, 191–208.

[9] Mäkelä, M. M. (2012): *Konveksi analyysi ja optimointi*, Luentomoniste. Turun yliopisto.

[10] Mäkelä, M. M. (2016): *Optimointialgoritmit*, Luentomoniste. Turun yliopisto.

[11] Parvinen, K. (2016): *Metapopulaatiomallit*, Luentomoniste. Turun yliopisto.

[12] Wolfram Language Documentation. Line Search Methods. Viitattu 17.11.2019. https://reference.wolfram.com/language/tutorial/UnconstrainedOptimizationLineSearchMethods.html

Liite A Data

Tässä liitteessä on taulukoitu esiteltyjen menetelmien generoimat pisteet ja kohdefunktion arvot aiemmin tarkasteltujen kuvien tilanteissa ratkaistaessa resurssi-kuluttajamallin singulaarista strategiaa, kun parametreina ovat $\delta = 1, \sigma = 0.8, \alpha = 1, K_1 = 10, K_2 = 20$. Kussakin taulukossa on viittaus siihen liittyvään kuvaan. Lisäksi esitetään kootusti mitatut CPU-ajat sekä tarkkuudet, joilla ratkaisut vastaavat todellista optimia.

Taulukko 1: Kelpoisuusgradienttimenetelmä (kuva 1).

k	β^k	$f(\beta^k)$
1	$(0.6, 0.2)$	1.62797
2	$(0.577277, 0.51817)$	5.86429×10^{-2}
3	$(0.461832, 0.554683)$	1.25828×10^{-2}
4	$(0.412315, 0.581024)$	2.02207×10^{-3}
5	$(0.397085, 0.597564)$	3.78642×10^{-4}
6	$(0.391665, 0.605643)$	7.69251×10^{-5}
7	$(0.389346, 0.609365)$	1.61649×10^{-5}
8	$(0.388307, 0.611086)$	3.44993×10^{-6}
9	$(0.387832, 0.611884)$	7.41579×10^{-7}
10	$(0.387613, 0.612255)$	1.59935×10^{-7}
11	$(0.387511, 0.612427)$	3.4546×10^{-8}
12	$(0.387464, 0.612507)$	7.46728×10^{-9}

Taulukko 2: USM+Hooke-Jeeves (kuva 6).

k	β^k	$f(\beta^k)$
1	$(0.6, 0.2)$	1.62797
2	$(0.7, 0.3)$	0.445361
3	$(0.67, 0.47)$	0.109753
4	$(0.549, 0.589)$	5.04112×10^{-2}
5	$(0.3643, 0.6723)$	4.19103×10^{-3}
6	$(0.3643, 0.6723)$	4.19103×10^{-3}
7	$(0.3643, 0.6723)$	4.19103×10^{-3}
8	$(0.3643, 0.6723)$	4.19103×10^{-3}
9	$(0.3893, 0.6223)$	2.50043×10^{-4}
10	$(0.3818, 0.6223)$	1.40967×10^{-4}
11	$(0.3818, 0.6223)$	1.40967×10^{-4}
12	$(0.3818, 0.6223)$	1.40967×10^{-4}
13	$(0.3818, 0.61605)$	8.33005×10^{-5}
14	$(0.38805, 0.611675)$	1.40572×10^{-6}
15	$(0.38805, 0.611675)$	1.40572×10^{-6}
16	$(0.38805, 0.611675)$	1.40572×10^{-6}
17	$(0.38805, 0.611675)$	1.40572×10^{-6}

Taulukko 3: USM+NLM (kuva 7a).

k	β^k	$f(\beta^k)$	NLM-iteraatiot
1	$(0.6, 0.2)$	1.62797	100
2	$(-0.777838, 0.388919)$	2.91333	100
3	$(0.207478, -0.299075)$	10.3396	100
4	$(0.86223, -0.451898)$	5.5017	12
5	$(0.455333, -0.313609)$	8.56573	24
6	$(0.773821, -0.22044)$	15.5548	100
7	$(-0.163308, -0.128582)$	39.4825	9
8	$(0.38756, 0.612707)$	1.54298×10^{-7}	

Taulukko 4: Modifioitu USM+NLM (kuva 7b).

k	β^k	$f(\beta^k)$
1	$(0.6, 0.2)$	1.62797
2	$(0.633918, 0.513122)$	8.52707×10^{-2}
3	$(-0.622979, 0.649272)$	3.70567
4	$(0.387025, 0.613600)$	1.47932×10^{-6}

Taulukko 5: USM+DFP-menetelmä (kuva 8).

k	β^k	$f(\beta^k)$
1	$(0.6, 0.2)$	1.62797
2	$(0.631505, 0.532673)$	8.42236×10^{-2}
3	$(0.384967, 0.549807)$	1.01344×10^{-2}
4	$(0.389165, 0.614762)$	3.16086×10^{-5}
5	$(0.387436, 0.612891)$	2.10975×10^{-7}
6	$(0.387429, 0.612588)$	5.98903×10^{-10}

Taulukko 6: USM+BFGS-menetelmä (kuva 9).

k	β^k	$f(\beta^k)$
1	$(0.6, 0.2)$	1.62797
2	$(0.631488, 0.532728)$	8.42184×10^{-2}
3	$(0.448715, 0.574278)$	8.34671×10^{-3}
4	$(0.385445, 0.589523)$	1.29905×10^{-3}
5	$(0.38706, 0.611772)$	2.64712×10^{-6}
6	$(0.387407, 0.61246)$	3.41708×10^{-8}
7	$(0.38742, 0.612567)$	3.31531×10^{-10}

Taulukko 7: USM+Hestenes-Stiefel-menetelmä (kuva 10).

k	β^k	$f(\beta^k)$
1	$(0.6, 0.2)$	1.62797
2	$(0.633918, 0.513122)$	8.52707×10^{-2}
3	$(0.412381, 0.542288)$	8.97996×10^{-3}
4	$(0.418483, 0.588641)$	2.33772×10^{-3}
5	$(0.387882, 0.6072)$	5.24844×10^{-5}
6	$(0.389535, 0.609927)$	1.39424×10^{-5}
7	$(0.387413, 0.612559)$	1.55906×10^{-9}

Taulukko 8: Zoutendijkin menetelmä (kuva 11).

k	β^k	$f(\beta^k)$
1	$(0.6, 0.2)$	1.62797
2	$(0.840415, 0.440415)$	0.190196
3	$(0.3014, 0.979429)$	7.81511×10^{-2}
4	$(0.255009, 0.933037)$	6.05722×10^{-2}
5	$(0.368954, 0.819092)$	4.44778×10^{-2}
6	$(0.314243, 0.764381)$	2.20435×10^{-2}
7	$(0.396153, 0.682471)$	9.9402×10^{-3}
8	$(0.36435, 0.650668)$	2.0858×10^{-3}
9	$(0.390203, 0.624814)$	4.15607×10^{-4}
10	$(0.383636, 0.618248)$	5.31491×10^{-5}
11	$(0.387795, 0.61409)$	6.73105×10^{-6}
12	$(0.386962, 0.613257)$	7.81186×10^{-7}
13	$(0.387467, 0.612752)$	9.05647×10^{-8}
14	$(0.38737, 0.612655)$	1.03834×10^{-8}
15	$(0.387428, 0.612597)$	1.19032×10^{-9}
16	$(0.387417, 0.612586)$	1.3628×10^{-10}

Taulukko 9: RSQP-menetelmä (kuva 12).

k	β^k	$f(\beta^k)$
1	$(0.6, 0.2)$	1.62797
2	$(0.455576, 0.291408)$	0.64405
3	$(0.469391, 0.371224)$	0.218838
4	$(0.445194, 0.45827)$	5.95518×10^{-2}
5	$(0.413526, 0.537809)$	1.03483×10^{-2}
6	$(0.394788, 0.590958)$	7.1925×10^{-4}
7	$(0.388149, 0.610433)$	6.63023×10^{-6}
8	$(0.387431, 0.612554)$	7.35072×10^{-10}
9	$(0.387423, 0.612577)$	9.29297×10^{-18}

Taulukko 10: Menetelmien CPU-ajat sekunteina eri vastaustarkkuuksilla. Menetelmät, jotka eivät konvergoineet kyseisellä tarkkuudella on merkitty taulukkoon viivalla ($-$).

Tarkkuus / Menetelmä	10^{-2}	10^{-4}	10^{-8}	10^{-12}
KGM	0.015625	0.03125	0.046875	0.078125
USM+Hooke-Jeeves	0.140625	0.203125	0.421875	0.625
USM+NLM	10.3438	33.6094	–	–
Modifioitu USM+NLM	0.109375	1.125	–	–
USM+DFP	0.109375	0.1875	0.3125	–
USM+BFGS*	0.109375	0.171875	–	–
USM+Hestenes-Stiefel	0.296875	0.390625	0.515625	–
Zoutendijk**	3.9375	5.03125	–	–
RSQP	0.46875	0.46875	0.5625	–

*Paras tarkkuus, mikä saavutettiin oli luokkaa 10^{-7} ajalla 0.328125.

**Paras tarkkuus, mikä saavutettiin oli luokkaa 10^{-7} ajalla 7.10938.