Globális optimalizálás
Előadás jegyzet

Előadó: G.-Tóth Boglárka
Jegyzet: Tűz Csaba

2009. december 15.

Megjegyzés. A jegyzet tartalma sok esetben G.-Tóth Boglárka könyvének fordítása, így szerzői jogvédelem alatt áll.

1. Optimalizálás

Általában: Lehetséges alternatívák közül válasszuk ki a „legjobbat”.
Alkalmazás-orientált terület → Az alkalmazott matematika része.

Főbb részterületek

- **Nemlineáris programozás**
- **Sztochasztikus programozás**: A változók valószínűségi változók, a cél egy függvény várható értékének optimalizálása.

  Megjegyzés. Előfordulhat, hogy egy nem sztochasztikus problémára sztochasztikus eljárást adunk meg. (Pl. Véletlen keresés)
- **Konvex programozás**: A célfüggvény és a megszorítások is konvexek. Ekkor mindig van pontosan egy optimum.
- **Lineáris programozás**
- **Konkáv programozás**: A megszorítások konvexek, a célfüggvény konkáv
- **Egészértékű programozás**
- **Kombinatorikus optimalizálás**
- **Robusztus programozás**
- **Kényszerek kielégítése**: Nincsen célfüggvény, csak a kényszerfeltételek adottak
- **Többcélú optimalizálás**: A célfüggvény egy vektorfüggvény. A legjobb megoldás kiválasztásában dominancia alapján döntünk, a Pareto-hatékony megoldást keressük. (Evidenciaelmélet).

  Példa (Kocsivétel)**.** Az ár-fogyasztás optimalizálása a cél. A megoldást a Pareto-hatékony halmaz adja.

- **Kvadratikus programozás**: A célfüggvény kvadratikus, a feltételek lineárisak. Ha a célfüggvény konvex, polinom idejű algoritmussal megoldható.
- **Globális optimalizálás**: Több lokális optimum van. A nemlineáris programozással ellentétben itt a globális optimumot keressük.

Általános globális optimalizálási feladat felírva:

$$\min_{x \in X} f(x)$$

$$X: \left.\begin{array}{ll} g_i(x) \leq 0 & i \in I \\ g_j(x) = 0 & j \in J \end{array}\right\} \text{kényszerek}$$

$$X \subseteq \mathbb{R}^n, \quad f: X \to \mathbb{R}$$

$f(x)$: célfüggvény
f^*: globális minimum

$$X^* = \{x \in X | f(x) = f^*\} = \{x \in X | f(x) \leq f(y) \quad \forall y \in X\}$$

Példa.
Egy hajózható csatornát keresünk a tenger egy megadott területén belül. A következő feladat megoldása szükséges.

a) Határozzuk meg a legkisebb távolságot a felszín és a fenék között, ha h a mélységet leíró függvény és X a megadott terület, például

$$X = \left\{ (x_1, x_2) \middle| \begin{array}{l} a \leq x_1 \leq b \\ c \leq x_2 \leq d \end{array} \right\},$$

akkor becsüljük a globális minimumot, f^*-ot.

b) Határozzuk meg azokat a pontokat, ahol a hajó fennakadhat.
Határozzuk meg

$$f_i^* < \bar{d}, \quad f^* = f_1^* < f_2^* < \cdots < f_l^* < \bar{d} < f_{l+1}^* < \ldots$$

f_i^*: Lokális optimum

$$X_i^* = \{x \in X | f(x) = f_i^*\} \quad - \quad \text{Lokális optimum pontok}$$

c) Határozzuk meg az $L_f(y) = \{x \in X | f(x) \leq y\}$ szintvonalakat.
A megoldhatóság mindig kérdéses az általános esetben. A megoldhatóság esélye apriori információk segítségével növelhető. Ilyen apriori információ pl:

- $f(x), \nabla f(x), \nabla^2 f(x), \ldots$
- Simaság, közelítő lokális minimum, alsó és felső korlátok a globális minimumra.

A globális optimalizáló eljárások elvi felépítése:

Globális fázis A teljes keresési régiót (X) elemzi

Lokális fázis Az ígéretes (optimumhoz közeli) részeket behatóbban vizsgálja.

Adaptáció Az ígéretes részeken sűrűbben mintavételezünk.

A globális optimum karakterizációja

Feltétel nélküli lokális optimumhoz

Szükséges feltétel: $\nabla f = 0$
Elégséges feltételek:

$$\begin{array}{lll} f''(x) > 0 & \textbf{min} & H_f(x) > 0 (\text{pozitív definit}) \\ f''(x) < 0 & \textbf{max} & H_f(x) < 0 (\text{negatív definit}) \end{array}$$

Ha minden lokális optimumot megtalálunk

$$f_1^* < f_2^* < \ldots; \qquad X_i^* = \{x \in X | f(x) = f_i^*\}$$

akkor a globális optimum elégséges feltétele

$$f(x^*) \leq f(x) \quad \forall x \in \bigcup X_i^*$$

Gyakorlatilag ez használhatatlan.

Definíció. Legyen $L_f(y) = \{x \in X | f(x) \leq y\}$ f szintvonalai. Ekkor a

$$G_f = \{y | y \in \mathbb{R}^n, L_f(y) \neq \emptyset\}$$

kifejezést $L_f(y)$ hatásos tartományának nevezzük.

Ekkor $f^* = \min_{y \in G_f} y$. Ez az eredeti globális optimalizálái feladat egy egydimenziós felírása. A G_f halmaz definíciója miatt viszont ez nem egy G.O. probléma.

Hogyan döntsük el egy minimumról, hogy lokális vagy globális?

Definíció. Az $F : y \mapsto L_f(y)$ pont-halmaz–leképezés alulról félig folytonos az $\bar{y} \in G_f$ pontban, ha

$$(\forall x \in L_f(\bar{y}))(\forall \{y^i \in G_f\}_i \to \bar{y})$$
$$(\exists K \in \mathbb{N})(\exists \{x^i \in X\}_i)(\forall i \geq K)(\{x^i\} \to x, x^i \in L_f(y^i))$$

Azaz ha egy $\bar{y}$ szintvonalhoz szintvonalak sorozatával tartunk, akkor megadható ezen sorozat mentén az $\bar{y}$ szintvonal bármely pontjához egy konvergens X-beli pontsorozat.

Tétel. Legyen $f : X \subset \mathbb{R}^n \to \mathbb{R}$ valós értékű függvény. Egy $\bar{y}$ értékre $L_f(y)$ alulról félig folytonos, ha $\forall x \in X$ amire $f(x) = \bar{y}$ esetén ha x egy optimumhely, akkor globális optimumhely.

Megjegyzés. A bizonyítás alapja hogy lokális optimum esetén, ha az y^i szintvonalsorozatra $\forall y^i$

Tegyük fel, hogy találtunk egy jelöltet f^*-ra, amely a z pontban vétetik fel. Affin transzformációval generálható egy g függvény, hogy

$$g(z) = -1 \text{ és } g(x) \leq 0 \quad \forall x \in X$$

Tétel. A konvex

$$G(t) = \int_X |g(x)|^t \, \mathrm{d}x$$

függvény $\lim_{t \to \infty} G(t)$ határértéke ∞, ha z nem globális minimalizáló pont.

Nem praktikus, numerikus integrálás, raszteres keresés...

Hogyan transzformáljunk át egy G.O. problémát L.O. problémává?

Tegyük fel, hogy f folytonos és X kompakt. Legyen $\tilde{X}$ az X legkisebb konvex burka. Legyen u

a) konvex $\tilde{X}$-en,

b) $u(x) \leq f(x), \forall x \in X$,

c) $u(x) >= g(x), \forall x \in X$, ahol g teljesíti a)-t és b)-t.

Ekkor u minimalizáló pontjainak halmaza tartalmazza f globális minimumhelyét.

Ez sem praktikus. u-t nehéz meghatározni és ha több globális minimum van, akkor u-nak végtelen sok minimumhelye van.

Megoldhatóság

f^*, Egy általános folytonos függvényre nem garantált, hogy $f^* + \varepsilon$ véges számú lépésben megtalálható.
X^*, Ez a probléma még csak nem is korrekt kitűzésű, mivel $\exists f : f(x) \leq f^* + \varepsilon$, de $\|x - x^*\| >> \delta$.

1.0.1. Megoldható problémák

- Ismert f^*, vagy alsó és felső korlátok generálhatóak a módszerrel
- f Lipschitz:
$$|f(x) - f(y)| \leq L \cdot \|x - y\| \quad \forall x, y \in X$$
- Folytonossági modulus:
$$w(t) = \max_{d(x,y) \leq t} |f(x) - f(y)|$$
Ha kiértékelünk N pontot
$$x_1, \ldots, x_N \in X$$
$$\tilde{f_N^*} := \min_{i \in \{1,\ldots,N\}} f(x_i) \qquad \text{a közelítő megoldásunk}$$
Ekkor a megoldásunk hibája becsülhető:
$$\left| \tilde{f_N^*} - f^* \right| \leq w(d_N)$$
ahol
$$d_N = \max_{x \in X} \min_{i \in \{1,\ldots,N\}} d(x, x_i)$$
a szétszórtság.

1.1. A lokális minimumok tulajdonságai

A lokális minimum definíciója miatt minden f_i^* minimum értékhez létezik egy olyan környezet

$$M_{i\varepsilon} = \{x \in X | f(x) \leq f_i^* + \varepsilon, \|x - x_i^*\| < \delta, \text{ ahol } x_i^* \in X_i^*\}$$

hogy $\mu(M_{i\varepsilon}) > 0$.

Definíció. Legyen f_i^* lokális minimum érték. Ekkor az f_i^* lokális minimum vonzási tartománya $\text{attr}(f_i^*) \subseteq X$ azon $x \in X$ pontok legnagyobb halmaza, amelyekből a legmeredekebb ereszkedés módszere végtelenül kicsi lépésközzel f_i^*-hoz konvergál $\forall x \in \text{attr}(f_i^*)$ esetén.

Megjegyzés. $\text{attr}(X)$ nem feltétlenül összefüggő. Ha a lokális minimumok száma > 1, akkor lehet nem összefüggő. A vonzási tartomány definiálható minimumhelyre is, $\text{attr}(f_i^*, x_i^*)$, a konvergencia x_i^* minimumhelyhez kell ekkor.

Definíció. f^* elérhető globális minimum, ha

$$\mu(\text{attr}(f^*)) > 0$$

ahol μ egy $\mathbb{R}^n$-beli mérték.

Megjegyzés. A legmeredekebb ereszkedés módszere minden lépésben $-\nabla f$ irányba halad valamilyen lépésközzel, azaz

$$x_{k+1} = x_k + \lambda \cdot d$$

ahol $d = -\nabla f(x_k)$.
Az optimális lépésköz:

$$\lambda^* = \arg\min_{\nu > 0} f(x_k + \nu \cdot d)$$

A végtelenül kicsi lépésköz akkor kell, hogy a legközelebbi lokális optimumot találjuk meg.

Definíció (Medence)**.**

$$\text{bas}(f_i^*, x_i^*) = \left\{ x \in \text{attr}(f_i^*, x_i^*) | f(x) < \hat{f}_i, \hat{f}_i = \min_{x \in \partial\, \text{attr}(f_i^*, x_i^*)} f(x) \right\}$$

$$\partial : \text{ határ}$$

Annak valószínűsége, hogy f_i^*-ot véletlen kiindulási pont választással megtaláljuk X-ben:

$$p_i = \frac{\mu(\text{attr}(f_i^*))}{\mu(X)}$$

A p_1 valószínűség kitüntetett, mert ez a globális optimum megtalálásának valószínűsége.
Ha $p_1 = 1$, akkor bármely $x \in X$-ből indulva az egyetlen (globális) minimumot találjuk.
Ha $p_1 = 0$, akkor a globális minimum nem lényeges.
Ha $p_1 = \max_i p_i$. Jó esélyünk van megtalálni f^*-ot.

1.2. Probabilisztikusan megoldható problémák

Egy G.O. probléma probabilisztikusan megoldható, ha egy eljárás 1 valószínűséggel megtalálja a globális optimumot, ha végtelen sokáig futhat. (gyenge konvergencia)

Ha csak elérhető globális minimum érdekel minket, ($p_1 > 0$) könnyedén konstruálhatunk ilyen eljárást:

Példa. Tiszta véletlen keresés: Véletlen generált mintapontokból lokális keresés, majd a legjobb lokális optimum kiválasztása.

Rács menti keresés: A keresési tartományon egy rács rácspontjaiból indítjuk a lokális keresést. A rács egyre sűrűbb lesz.

1.3. Problémák osztályozása

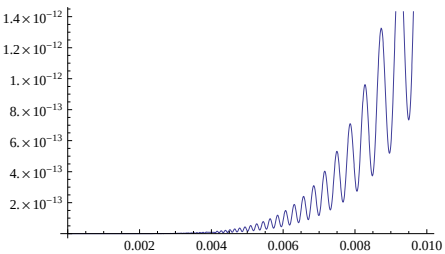
Szempontok:

- Dimenzionalitás
- Tulajdonságok: konvex, konkáv, kvadratikus, Lipschitzes
- Feltétel nélküli($x \in \mathbb{R}^n$), sima korlátok($X = \{x \in \mathbb{R}^n | a \leq x \leq b\}$), feltételes
- Egyéb információk: analitikus megadás, gradiens, Hesse–mátrix

  Megjegyzés. Feketedoboz-probléma: Nincs információnk a függvényről, csak egy kiértékelő „dobozunk” (eljárásunk).

  Lokális optimumok száma (megállási feltételt ad), f értékkészlete X-ben.

1.4. Mintaproblémák



1. ábra. Az $f(x)$ függvény oszcillál a nulla közelében

$$f(x) = \begin{cases} \sum_{i=1}^{n} x_i^6 \left(\sin \frac{1}{x_i} + 2\right) & \forall i : x_i \neq 0 \\ 0 & \text{egyébként} \end{cases}$$

Ha $n = 1$, akkor

$$f(x) = \begin{cases} x^6 \left(\sin \frac{1}{x} + 2\right) & x \neq 0 \\ 0 & \text{egyébként} \end{cases}$$

A globális minimum az $x = 0$ helyen van, értéke 0. A $\sin \frac{1}{x}$ oszcillál a 0 közelében. Az 1. ábrán az $f(x)$ függvény $]0, 0.1]$ intervallumon felvett képe látható.

Ebben az esetben $p_1 = 0$, tehát az $f^* = 0$ minimum nem elérhető minimuma a függvénynek, holott folytonos a 0 közelében. Ez szemlélteti, hogy a folytonosság nem elégséges feltétele az elérhető minimumnak.

Definíció (Rosenbrock függvény)**.**

$$f(x, y) := (1 - x)^2 + 100(y - x^2)^2$$

A függvény négyzetek összege, könnyű látni, hogy $f(1, 1) = 0$. Tehát az $(1, 1)$ pont globális minimumhelye a függvénynek. A minimum egy parabolikus „völgyben” helyezkedik el. Egy optimalizáló eljárás általában gyorsan eljut a völgybe, azonban ott oszcilláció alakul ki, ami miatt az optimumhoz való konvergencia lassú lehet.

A Rosenbrock–függvény egy elterjedt tesztfüggvény optimalizáló eljárások teljesítményének mérésére.

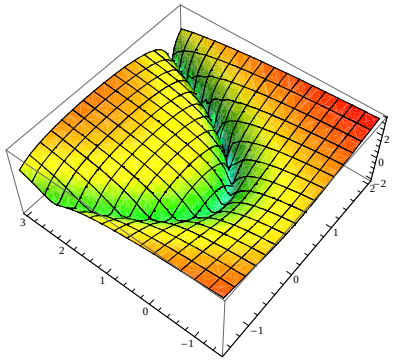
1.5. Globális optimalizáló eljárások osztályozása

Determinisztikus - probabilisztikus eljárások

Definíció (Heurisztikus eljárás)**.** Minden olyan eljárás, amely képes elfogadható megoldást adni, de a helyességet bizonyítani nem tudjuk.

Megjegyzés. Heurisztika $\neq$ probabilisztikus eljárás. A raszter keresés például egy determinisztikus heurisztika.

Definíció (Nemteljes eljárás)**.** Lehetséges, hogy nem a globális optimumhoz konvergál



2. ábra. A Rosenbrock „banán”-függvény a logaritmustérben

Definíció (Aszimptotikusan teljes eljárás)**.** Olyan eljárás, mely biztosan, vagy 1 valószínűséggel eléri a globális optimumot, ha végtelen ideig futhat, de soha nem tudhatjuk, hogy a globális minimumot találtuk-e meg.

Definíció (Teljes eljárás)**.** Olyan eljárás, mely egzakt aritmetikát feltételezve biztosan eljut a globális optimumhoz, ha végtelen ideig futhat, valamint véges sok lépést követően tudja, hogy egy közelítő globális minimumot talált (előre megadott tűrésen belül).

Definíció (Megbízható eljárás)**.** Olyan teljes eljárás, amely kerekítési hibák mellett is komplett.

Definíció (Egzakt eljárás)**.** A szigorú, illetve komplett eljárásokat összefoglaló néven egzakt eljárásoknak nevezzük.

Példa (Nemteljes eljárások)**.** Newton módszer, Genetikus algoritmus

2. Lipschitz-optimalizálás

Definíció (Lipschitz–folytonosság)**.** Azt mondjuk, hogy egy $f : D \subset \mathbb{R}^n \to \mathbb{R}$ valós függvény minden változójában kielégíti a Lipschitz-feltételt (azaz Lipschitz), ha

$$|f(x_1) - f(x_2)| \leq L \|x_1 - x_2\| \quad \forall x_1, x_2 \in D$$

valamely $L > 0$ konstansra. A lehetséges L-ek infimumát Lipschitz konstansnak nevezzük.

Egy optimalizálási feladat Lipschitz-féle, ha f és $\forall i g_i$ mind Lipschitz-folytonos.

Tétel. Legyen f folytonosan differenciálható függvény egy D nyílt halmazon és X egy kompakt részhalmaza D-nek. Ekkor f Lipschitzes X-en.

Bizonyítás. A Lagrange-középértéktételből:

$$f(x) = f(\hat{x}) + \nabla f(\tilde{x})(x - \hat{x}) \quad \text{valamely } \tilde{x} \in [x, \hat{x}]$$

Így

$$|f(x) - f(\hat{x})| = |\nabla f(\tilde{x})(x - \hat{x})| \leq \|\nabla f(\tilde{x})\| \cdot \|x - \hat{x}\|$$

Kompakt halmazon folytonos függvénynek létezik a maximuma, így

$$L := \max_{x \in X} \|\nabla f(x)\|$$

jó lesz. □

Tétel. Legyen X egy kompakt halmaz, $\{f_j\}_{j \in \{1,\ldots,n\}}$ Lipschitz függvények X-en, valamint h legyen egy egyváltozós Lipschitzes függvény f_j értékkészletén. Ekkor a következő függvények szintén Lipschitz folytonosak:

$$f(x) := \sum_{j=1}^{n} c_j f_j(x) \quad c_j \in \mathbb{R}$$

$$\min_{j \in \{1,\ldots,n\}} f_j(x); \max_{j \in \{1,\ldots,n\}} f_j(x); f_i(x) f_j(x) \quad \forall i, j \in \{1, \ldots, n\}$$

$$h(f_i(x)) \quad \forall i \in \{1, \ldots, n\}$$

Bizonyítás. ($\max f_j(x)$ esetére)

$$\left| \max_j f_j(x_1) - \max_j f_j(x_2) \right| \leq \max_j |f_j(x_1) - f_j(x_2)| \leq \max_j L_j \|x_1 - x_2\|$$

ahol L_j az f_j függvényhez tartozó Lipschitz-konstans.

A rács menti kereséssel δ maximális távolsággal

$$\tilde{f} = \min_{i < K} f(x_i)$$

közelítése az f^* optimumnak. $\delta < \frac{\varepsilon}{L}$ választással

$$\left| \tilde{f} - f^* \right| < \varepsilon$$

□

3. Pyavskii-Schubert algoritmus

$$F(x) = f(\hat{x}) - L\,\|x - \hat{x}\| \quad \hat{x} \in X$$

$F(x)$ alsó becslés $f(x)$-re X-en. Az algoritmus egydimenziós függvényekre adott, legyen $X = [a, b]$

$$x_1 := \frac{a+b}{2}, X = [a, b]$$

$$F_1(x) = f(x_1) - L\,|x - x_1|$$

$$x_2 = \arg\min_{x \in X} F_1(x) \quad x_2 \in \{a, b\}$$

$$F_2(x) = \max_{i \in \{1,2\}} \{f(x_i) - L\,|x - x_i|\}$$

$$x_{k+1} = \arg\min_{x \in X} F_k(x) \qquad F_k(x) = \max_{i \in \{1,\dots,k\}} \{f(x_i) - L\,|x - x_i|\}$$

Ha $\tilde{f}$ az f^* optimum egy felső korlátja($\tilde{f} = \min_{i \in \{1,\dots,n\}} f(x_i)$), akkor az

$$A_k = \left\{x \in X | F_k(x) \leq \tilde{f}\right\}$$

halmaz tartalmazza a globális optimumhelyet.

Ha $f''(x) \leq K \quad \forall x \in X$, azaz a második derivált korlátos, a módszer részenként kvadratikus alsó korlát függvénnyel is működik.

Példa.

$$f(x) = \sin x + \sin 3x + \ln x \qquad x \in [3, 7]$$

Minimalizálni akarunk a Pyavskii-Schubert algoritmussal. Adjunk becslést az L Lipschitz konstansra.

$$|f(x_1) - f(x_2)| \leq L\,\|x_2 - x_1\| \qquad \forall x_1, x_2 \in X$$

Minden folytonosan differenciálható függvényhez $\exists L$ Lipschitz konstans egy kompakt tartományon. Nem szükséges feltétlenül a legkisebb Lipschitz konstanst megadni, ezért tagonként becslünk.

$$L = \max_{x \in X} \|\nabla f\|$$

$$f'(x) = \cos x + 3\cos 3x + \frac{1}{x}$$

$$|f'(x)| \leq |\cos x| + |3\cos 3x| + \left|\frac{1}{x}\right| \leq 1 + 3 + \frac{1}{3}$$

Tehát $L = \frac{13}{3}$ egy jó becslés.

Példa. Legyen f Lipschitzes függvény $[a, b]$-n, egy Lipschitz konstans pedig L. Mutassuk meg, hogy ha $\exists x_1, x_2 \in [a, b]$ amire

$$f(x_1) = f(x_2) - L(x_2 - x_1) \quad x_2 > x_1$$

akkor

$$f(x) = f(x_2) - L(x_2 - x) \quad \forall x \in [x_1, x_2]$$

Megoldás: Legyen $x \in (x_1, x_2]$

$$|f(\bar{x}_1) - f(x)| \leq L\,|x - \hat{x}_1|$$

$$-L(x - x_1) \leq f(x_1) - f(x)$$

$$-L(x - x_1) \leq f(x_2) - L(x_2 - x_1) - f(x)$$

$$L(x_2 - x) \leq f(x_2) - f(x)$$

$$f(x_2) - f(x) \leq L(x_2 - x)$$

Példa. Tekintsük az

$$f(x) = x^4 - 12x^3 + 47x^2 - 60x$$

függvényt. Adjunk becslést az L Lipschitz konstansra!

Megoldás:

$$f'(x) = 4x^3 - 36x^2 + 94x - 60$$

$$\max_{x \in X} |f'(x)|$$

$$L = \max_{x \in X} |f'(x)| \leq \max_{x \in X} 4x^3 - \min 36x^2 + \max |94x| - 60 = 4 \cdot 6^3 + 0 + 94 \cdot 6 - 60 = 1368$$

túl nagy. Legyen $L = 100$. Számítuk ki x_5-öt:

$$\begin{array}{ll} x_1 = 3 & f(x_1) = 0 \\ x_2 = 0 & f(x_2) = 0 \\ x_3 = 6 & f(x_3) = 36 \end{array}$$

$$\min_{x \in [x_2, x_1]} F_k(x) = f(x_1) - \frac{x_2 - x_1}{2} L \qquad \text{mert} \quad f(x_1) = f(x_2)$$

$$\min_{x \in [x_1, x_3]} F_k(x) : f(x_1) - L(x - x_1) = f(x_3) - L(x_3 - x)$$

$$f(x_1) - f(x_3) = 2Lx - Lx_1 - Lx_3 = 2Lx - L(x_1 + x_3)$$

$$x = \frac{f(x_1) - f(x_3)}{2L} + \frac{x_1 + x_3}{2} \qquad x_1 < x_3$$

$$\min_{x \in [x_1, x_2]} F_k(x) = f(x_1) - L\frac{f(x_1) - f(x_3)}{2L} - L\frac{x_1 + x_3}{2} + Lx_1$$

$$\min_{x \in [x_1, x_3]} F_k(x) = \frac{f(x_1) + f(x_3)}{2} + L\frac{x_1 - x_3}{2} \qquad x_1 < x_3$$

3.1. A Pyavskii-Schubert algoritmus pszeukódja

Algorithm 1 Pyavskii-Schubert

List=$\{y_1, y_2, \ldots, y_k\}$ ahol $\{y_1 \leq \cdots \leq y_k\} = \{x_1, \ldots, x_k\}$.

Intervallum módszer:

input : $f(x), [a, b]$

$m = \frac{a+b}{2}$

$\mathcal{L} = (I_1 = [a, m], I_2 = [m, b])$ $\quad //I = [l, r]$

$\tilde{f} = \min\{f(a), f(b), f(m)\}$

$\tilde{x} = \arg\min\{f(a), f(b), f(m)\}$

$F_i = \frac{f(l_i)+f(r_i)}{2} - L\frac{r_i - l_i}{2}$

while $\mathcal{L} \neq \emptyset$ **do**

  Vegyük le az I_i intervallumot az $\mathcal{L}$ listából, amire $F_i = \min_{j \in \{1,\ldots,k\}} F_j$

  $m_k = \frac{l_i+r_i}{2} + \frac{f(l_i)-f(r_i)}{2L}$

  Kiértékel($f(m_k)$)

  if $f(m_k) < \tilde{f}$ **then**

    $\tilde{f} = f(m_k); \tilde{x} = m_k$

  for $(i = 1, \ldots, k)$ **do**

    if $F_j > \tilde{f}$ **then**

      Töröl(α,I_j)

    $I_{k+1} = [l_i, m_k], I_{k+2} = [m_k, r_i]$

    Kiértékel(F_{k+1},F_{k+2})

    if $F_{k+1} < \tilde{f}$ **then**

      Beszúr($\mathcal{L}$,I_{k+1})

    if $F_{k+2} < \tilde{f}$ **then**

      Beszúr($\mathcal{L}$,I_{k+2})

  $k = k + 2$

4. Korlátozás és szétválasztás

Szétválasztás: A feladatot részfeladatokra osztjuk

Korlátozás: A feldolgozás során also korlátokat állapítunk meg a globális optimumra, ami révén az optimumot biztosan nem tartalmazó részfeladatok kiküszöbölhetjük.

4.1. Prototipus algoritmus: korlátos, korlátozó feltételekkel rendelkező feladathoz

$\mathcal{L}$ - Részfeladatok listája

$\tilde{f}$ - felső korlát a globális minimumra.

$\mathrm{Lb}\,(y)$ - alsó korlát f-re az Y halmazon.

Algorithm 2 Prototípus algoritmus

```
α = {x}, f̃ = ∞        /inicializálás/
while α ≠ ∅ do
  y kiválasztása L-ből, a kiválasztási szabály alapján
  Kiértékelés: f(v)
  f̃ = min{f̃, f(v)}
  Y felosztása Y_1, ..., Y_p halmazokra. /felosztási szabály/
  for i = 1, ..., p do
    Lb(Y_i) meghatározása ∀i /korlátozási szabály/
    if Lb Y_i ≤ f̃ then
      Y_i-t hozzávesszük L-hez. /kiküszöbölési szabály/
    else
      Y_i-t töröljük.
```

4.1.1. Kiválasztási szabály

a) Véletlen kiválasztás

b) Alsó korlát alapján $\arg\min_{y\in\mathcal{L}} \mathrm{Lb}\,(y)$

c) Méret alapján $\arg\max_{y\in\mathcal{L}} \mathrm{hossz}(y)$

d) Élettartam alapján (F(irst) I(n) F(irst) O(ut)-lista)

Megjegyzés. Ha $|y_i|$ azonos $\forall i \quad \Rightarrow c) \Leftrightarrow d)$

4.1.2. Felosztási szabály

$$Y = \bigcup_{i=1}^{p} Y_i \qquad Y_i \cap Y_j = \emptyset, \quad \forall i \neq j$$

Általában Y egy szimplex, téglalap, vagy kúp.

a) felezés (általában a legnagyobb oldalnál felezek)

b) multiszekció (többfelé vágás)

Jó esetben Y olyan halmaz, amit önmagához hasonló halmazokra oszthatunk. (Pl. háromszögelés)

4.1.3. Korlátozási szabály

Mindig az aktuális algoritmus határozza meg, nem általánosítjuk.

4.1.4. Kiküszöbölési szabály

Miután $\tilde{f}$-et frissítjük, eltávolítunk $\forall y \in \mathcal{L}$-t, amire

$$\mathrm{Lb}\,(y) > \tilde{f}$$

Ha feltételekkel korlátozott problémánk van, akkor a nem megengedett megoldásokat is eltávolítjuk.

Legyen a $\tilde{f}, Y, v, \mathcal{L}$ négyen $\hat{f}_k, Y_k, v_k, \alpha_k$ a k. iteráció után.

Tétel. ha minden végtelen $\{Y_{k_q}\} : Y_{k_q} \supset Y_{k_{q+1}} \quad (q = 1, 2, \dots)$ finomodó partíció-halmazokra a korlátok kielégítik a

$$\lim_{q\to\infty} (\tilde{f}_{k_q} - \mathrm{Lb}(Y_{kq})) = 0$$

feltételt, akkor

$$\lim_{k\to\infty} \mathrm{Lb}\,\mathcal{L}_k = \lim_{k\to\infty} f(v_k) = \lim_{k\to\infty} \tilde{f}_k$$

és $\{v_k\}$ minden v^* torlódási pontja globális optimuma a $\min\{f(x) : x \in X\}$ problémának, ahol $\mathrm{Lb}(\mathcal{L}_k) = \min_{y\in\mathcal{L}_k} \mathrm{Lb}(y)$.

Bizonyítás. Létezik $\{v_k\}$-nak torlódási pontja, mert X kompakt. Legyen b^* a $\{v_k\}$ halmaz torlódási pontja, ekkor létezik olyan $\{v_{k_q}\}$ részsorozat, ami v^*-hoz konvergál, hogy

$$Y_{k_q} \supset Y_{k_{q-1}} \qquad v_{k_q} \in Y_{k_q} (q = 1, 2, \dots)$$

Így f folytonossága miatt

$$\lim_{q\to\infty} f(v_{k_q}) = f(v^*)$$

Az $\mathrm{Lb}(\mathcal{L}_k)$ sorozat monoton növekvő és korlátos az f^* globális minimum által. Ezért a $k \to \infty$ határértéke létezik.
Mindemellett $\tilde{f}_k$ monoton csökkenő és felülről korlátolt f^* által, így van határértéke.
Az alsó korlát szerinti kiválasztást használva, és a

$$\lim_{q\to\infty\left(\tilde{f}_{k_q} - \mathrm{Lb}(y_{k_q})\right)} = 0$$

feltétel mellett

$$\lim_{k\to\infty} \mathrm{Lb}(\alpha_k) = f^* = \lim_{k\to\infty} f(v_k) = f(v^*) = \lim_{k\to\infty} \tilde{f}_k$$

□

Megjegyzés. Ugyanez belátható a méret és élettartam szerinti kiválasztásra is. Sőt, az is igaz, hogy x^* megegyezik a torlódási pontok halmazával.

Tegyük fel, hogy $\mathrm{Lb}(Y_2) = f^*$, de $\mathrm{Lb}(Y_k) < f^*$, minden $k = 1, 3, 4, 5, \dots$ esetén.

Ekkor $\mathrm{Lb}(y_k) \to f^*$, de sosem egyenlő. Mindig Y_1-et felezem, Y_2-be sose megyek.

Ha méret, vagy kor szerint választok, akkor mindkettő igaz lesz.

4.1.5. A Lipschitz-optimalizáció korlátozási szabálya

Legyen Y halmaz adott, szükségünk van $\mathrm{Lb}(y)$-ra.

Legyen Y egy n dimenziós tégla, $[a, b], \quad a, b \in \mathbb{R}^n$.

$$|f(x_1) - f(x_2)| \leq L \|x - y\|$$

$$\mathrm{Lb}(y) = f\left(\frac{a+b}{2}\right) - L\frac{\|a-b\|}{2}$$

A $v_1, \ldots, v_l \in Y$ mintapontra

$$\hat{v}^* = \arg \max_{i=1,\ldots,l} f(v_i)$$

$$\mathrm{Lb}(y) = f(\hat{v}) - L \max_{x \in Y} \|\hat{v} - x\|$$

$$\lim_{q \to \infty} \tilde{f}_{k_q} - \mathrm{Lb}(y_{k_q}) = 0$$

$$|f(x_1) - f(x_2)| \leq L \|x_1 - x_2\| \quad \forall x_1, x_2 \in X$$

$$\lim_{q \to \infty} \delta(Y_{k_q}) = 0$$

A $\delta(Y_{k_q})$ az Y_{k_q} halmaz átmérője.

Definíció. Egy felosztás kimerítő, ha minden beágyazott $\{Y_{k_q}\}$ végtelen sorozatra

$$\delta(Y_{k_q}) \to 0$$

Definíció. Tekintsünk egy tetszőleges n-szimplexet, $V(Y)\{v_0, \ldots, v_n\}$ csúcshalmazzal. Ekkor az adott szimplex radiális felosztásán az alábbit értjük: Legyen $w \in Y, w \neq V(Y)$. Ez reprezentálható a

$$w = \sum_{i=0}^{n} \lambda_i v_i$$

súlyozott összeggel, ahol $\lambda_i \geq 0 \quad \forall i$, és $\sum_{i=0}^{n} \lambda_i = 1$.

Minden olyan i-re, amire $\lambda_i > 0$, definiáljuk az alábbi részhalmazokat:

$$Y_i = \mathrm{conv}\{v_0, \ldots, v_{i-1}, w, v_{i+1}, \ldots, v_n\}$$

5. D.C. Programozás

D.C - konvex függvények különbsége (Difference of Convexes)

Definíció. Egy $f : D \subseteq \mathbb{R}^n \to \mathbb{R}$ (D konvex) függvényt D.C. függvénynek nevezünk, ha léteznek olyan p és q konvex függvények D-n, hogy

$$f(x) = p(x) - q(x) \quad \forall x \in D$$

Definíció (D.C. optimalizálási feladat)**.** Keressük $\min f(x)$-et, ahol

$$x \in C, g_j(x) \leq 0 \quad \forall j = 1, \ldots, m$$

és C zárt, konvex halmaz, f, g_j pedig mind D.C. függvények.

$$f(x) = p(x) - q(x) \Rightarrow \mathrm{Lb}(f(x)) = \mathrm{Lb}(p(x)) - \mathrm{Ub}(q(x))$$

$$\lambda f(v_i) + (1 - \lambda) f(v_j) \quad \text{2 dimenzióban}$$

$$F(x) := \sum \lambda_i f(v_i), \qquad \text{ahol } \lambda = \sum \lambda_i v_i$$

5.1. D.C. függvények tere

Állítás. Legyenek $f, f_i (i = 1, \ldots, m)$ D.C. függvények. Ekkor az alábbi függvények szintén D.C.-beliek:

- $$\sum_{i=1}^{m} \lambda_i f_i(x) \quad \lambda_i \in \mathbb{R}$$
- $$\max_{i=1,\ldots,m} f_i(x); \qquad \min_{i=1,\ldots,m} f_i(x)$$
- $$\prod f_i(x); \quad |f(x)|; \quad f^+(x); \quad f^-(x)$$

Bizonyítás. Pl. $\max f_i(x)$:

$$\max f_i(x) = \max(p_i(x) - q_i(x)) = \max \left(p_i(x) + \sum_{i \neq j} q_j(x) \right) - \sum_j q_j(x)$$

etc... □

Definíció. Egy függvény lokálisan D.C., ha $\forall x_0 \in \mathbb{R}^n : \exists M(x_0, \varepsilon)$ környezet, hogy f D.C. $M(x_0, \varepsilon)$-on.

Tétel. Minden lokálisan D.C. függvény globálisan is D.C

Következmény. Minden $f : \mathbb{R}^n \to \mathbb{R} \in C^2$ függvény D.C.

Bizonyítás. A $\nabla^2 f$ Hesse–mátrix minden elem korlátos, bármely $M(x_0, \varepsilon)$ környezeten. Ezért egy kellően nagy $M > 0$ esetén $f(x) + M \|x\|^2$ konvex $M(x_0, \varepsilon)$-on (mivel a $\nabla^2 f + 2M\mathcal{I}$ pozitív szemidefinit kellően nagy M-re).

Ezért

$$f(x) = f(x) + M \|x\|^2) - M \|x\|^2$$

az f egy D.C. felbontása $M(x_0, \varepsilon$-on, és a fenti tétel miatt bárhol, ha M elég nagy. □

Megjegyzés. Egy $g \circ f$ kompozíció D.C, ha f D.C. és g konvex.

5.2. Kanonikus D.C. programozás

$$\min c^T x$$

ahol $x \in C, g(x) \geq 0$, $c \in R^n$, C zárt, konvex halmaz, $g : \mathbb{R}^n \to \mathbb{R}$ konvex.

Minden D.C. program ilyen alakban írható.

$$\min_{x \in D} f(x)$$

$$g_j(x) \leq 0 (j = 1, \ldots, m)$$

ahol f, g_j mind D.C-k, D zárt, konvex.

Ebből a feladatból származtathatjuk az alábbi alakot:

$$\left.\begin{array}{l} \min t \\ g_0(x) = f(x) - t \\ g_i(x) \leq 0 \quad x \in D \end{array}\right\} \otimes$$

Ekkor t minimalizálásával egyben $f(x)$-et is minimalizáljuk.

$$g_j(x) = p_j(x) - q_j(x)$$

$$\otimes \Leftrightarrow \underbrace{\max_{j=0,\ldots m} g_j(x)}_{=g(x)=p(x)-q(x)\leq 0} \leq 0$$

$$\text{konvex} \left\{ \begin{array}{l} \varphi(x, s) = p(x) - s \leq 0 \\ \psi(x, s) = q(x) - s \geq 0 \end{array} \right.$$

Ezekből:

$$p(x) \leq s; \quad q(x) \geq s \Rightarrow p(x) \leq q(x)$$

$$C = \{x \in D : \varphi(x, s) \leq 0\} \text{ konvex}$$

Példa. Legyen $f(x_1, x_2) = x_1 x_2$. Adjuk meg f D.C. dekompozícióját.

$$\min c^T y, \quad C = (0, \ldots, 0, 1, 0)$$

$$y \in C = \{(x, y_{n+1}, y_{n+2}) | x \in D, f(x, y_{n+1}) \leq 0\}$$

$$p(x) = \frac{(x_1 + x_2)^2}{2} \qquad q(x) = \frac{x_1^2}{2} + \frac{x_2^2}{2}$$

Legyen $f(x) = p(x) - q(x)$ az f D.C. felbontása. Adjunk alsó becslést $f(x)$-re az $X = ([-1,2],[1,3])$ intervallumon.

$$\mathrm{Lb}(f(x)) = \mathrm{Lb}(p(x)) - \mathrm{Ub}(q(x))$$

$$\mathrm{Ub}(q(x)) = \max q(v_i) = \frac{2^2}{2} + \frac{3^2}{2} = 6.5$$

Az $\mathrm{Lb}(p(x))$ alsó becsléshez:

$$\nabla p(x) = \begin{pmatrix} x_1 + x_2 \\ x_1 + x_2 \end{pmatrix} \qquad \nabla p(0,1) = \begin{pmatrix} 1 \\ 1 \end{pmatrix}$$

$$p(0,1) + \nabla p(0,1)^T \left(x - \begin{pmatrix} 0 \\ 1 \end{pmatrix} \right) = \frac{1}{2} + \begin{pmatrix} 1 & 1 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 - 1 \end{pmatrix} = \frac{1}{2} + x_1 + x_2 - 1 = x_1 + x_2 - \frac{1}{2}$$

A most felírt hipersík alsó korlátot ad a függvényre, így a $-\frac{1}{2}$ jó alsó becslés. Ezzel

$$\mathrm{Lb}(f) = -0.5 - 6.5 = -7$$

(Jelen esetben ez persze nem éles, nyilvánvaló, hogy a $p(x)$-re a nulla egy természetesen adódó alsó korlát, de ez nem mindig ilyen triviális.

Keressük

$$\min c^T x$$

ahol $C \subseteq \mathbb{R}^n$ konvex, zárt halmaz, z konvex, $c \in \mathbb{R}^n$. Feltesszük, hogy C egy politóp, amelynek belseje nem üres. Az $F = \{x \in Cg(x) \geq 0\}$ megengedett halmazra megköveteljük, hogy $\mathrm{cl}(\mathrm{int}(F))=F$.

5.3. Egy élkövető algoritmus

5.3.1. Inicializáció

Oldjuk meg a $\min_{x \in C} c^T x$ lineáris programozási feladatot.
Ha a megoldásra $x^0, g(x^0) \geq 0$, megállunk, a megoldás x^0.
Egyébként konstruálunk egy n-szimplexet $S^0 \geq C$, hogy x^0 az S^0 egy csúcsa.
Kiszámítjuk a $V(S^0)$ csúcshalmazt, majd $S = S^0$, $V = V(S^0)$.
Meghatározzuk a $z \in V \quad g(z) = \max_{x \in V} g(x)$ pontot. Ha $g(z) = 0$, megállunk, $F = \emptyset$

Példa.

$$\min -\frac{1}{2}x_1 - \frac{1}{6}x_2$$

$$\begin{aligned} x_2 - x_1 &\leq 2 \\ \frac{1}{3}x_1 + \frac{1}{2}x_2 &\geq 1 \\ 0 \leq x_1 &\leq 6 \\ 0 \leq x_2 &\leq 4 \end{aligned}$$

$$x_1^2 + x_2^2 - 8x_1 - 6x_2 + 16 \geq 0 \quad g(x) \geq 0$$
$$(x_1 - 4)^2 + (x_2 - 3)^2 \geq 0$$

$y := x^0$
for $k = 1, 2, \dots$ **do**
  if $g(z) < 0$ **then**
    STOP, $F = \emptyset$
  if $g(z) = 0$ **then**
    STOP, y optimális
  if $g(z) > 0$ **then**
    Keresünk olyan $[u, v]$ élét S^0-nak, hogy $g(u) \geq 0, g(v) \leq 0$, valamint $c^T v < c^T u$. (Pl. szimplex algoritmus, z kiindulási pontból.)
    Határozzuk meg $s \in [u, v], g(s) = 0$.
    if $s \in C$ **then**
      $y = s, S^k = S^{k-1} \cap \{x : c^T x \leq c^T y\}$
    else
      $S^k = S^{k-1} \cap \{x : l(x) \leq 0\}$, ahol $l(x) \leq 0$ meghatározza a C halmazt, és $l(s) > 0$
      Kiszámítjuk a $V(S^k)$ csúcshalmazt, és a $z \in V^k \quad q = \arg\max_{x \in V^k} g(x)$ pontot.

$$S^0 = \{x : x_1 \leq 6, x_2 \leq 4$$
$$V^0 = \{(6, 4), (6, -2), (-3, 4)\}$$
$$z^0 = (-3, 4)$$

$S^1 \notin C$

$S^1 = S^0 \cap \{x : x_2 - x_1 - 2 \leq 0\}$ olyat kellett választani, ami meghatározza C-t, és kizárja s-et.

$$V^1 = \{(6, 4), (6, -2), (0, 2), (2, 4)\}$$
$$z^1 = (6, -2)$$
$$u = z^1, v = x^0, S^2 = (6, 0.764)$$

$$S^2 \in C \Rightarrow y = S^2, S^2 = S^1 \cap \{x : -\frac{1}{2}x_1 - \frac{1}{6}x_2 \leq -3.127\}$$
$$v^2 = \{(6, 4), (6, 0.764), (4..., 4)\}$$

$z = (6, 0.764), \quad g(z) = 0 \Rightarrow$ STOP

Példa.

$$\min_{x \in ([-4,4],[-2,2])} \left(\min\{(x_1 - 2)^2 + x_2^2, (x_1 + x_2)^2 + x_2^2 - 0.1\}\right)$$

Mi f^* vonzási tartománya? ($f^* = 0.1$)
Mi az $Lf(1)$ szintvonal?
Mennyi p_1 közelítőleg?
Mi a medence?

$$p_1 = \frac{\mu(\text{attr}(f^*))}{\mu(X)} \approx 0.5$$

Példa. Legyen $f : \mathbb{R} \to \mathbb{R}$ konkáv a $(-\infty, a)$ intervallumon, és konvex az $[a, \infty)$ intervallumon.
Adjunk meg f D.C. felbontását az f' derivált segítségével egy megadott pontban.

$$
\begin{aligned}
f(x) &\approx f(a) + f'(a)(x-a) \\
p(x) &= \begin{cases} f(a) + f'(a)(x-a) & \text{ha } x < a \\ f(x) & \text{ha } x \geq a \end{cases} \\
f(x) &= p(x) - q(x) \\
q(x) &= p(x) - f(x) \\
q(x) &= \begin{cases} -f(x) + f(a) + f'(a)(x-a) & \text{ha } x < a \\ 0 & \text{ha } x \geq a \end{cases}
\end{aligned}
$$

II. megoldás:

$$
\begin{aligned}
p_2(x) &= \begin{cases} \frac{1}{2}\left(f(a) + f'(a)(x-a)\right) & x < a \\ f(x) - \frac{1}{2}\left(f(a) + f'(a)(x-a)\right) & x \geq a \end{cases} \\
q_2(x) &= \begin{cases} -f(x) + \frac{1}{2}\left(f(a) + f'(a)(x-a)\right) & x < a \\ \frac{1}{2}\left(f(a) + f'(a)(x-a)\right) & x \geq a \end{cases}
\end{aligned}
$$

5.4. Intervallum analízis

Autómatikusan korlátot szolgáltat egy szubrutin által szolgáltatott függvényre intervallum bemenettel.

5.4.1. Műveletek az intervallumokon

Kell:

$$A \circ B = \{a \circ b | a \in A, b \in B\}$$

Legyenek A, B intervallumok, $0 \notin B$. EKkor

$$
\begin{aligned}
[a,b] + [c,d] &= [a+c, b+d] \\
[a,b] - [c,d] &= [a-d, b-c] \\
[a,b] \cdot [c,d] &= [\min\{ac, ad, bc, bd\}, \max\{ac, ad, bc, bd\}] \\
[a,b]/[c,d] &= [a,b] \cdot \left[\frac{1}{d}, \frac{1}{c}\right]
\end{aligned}
$$

A halmazelméleti definíció ekvivalens az aritmetikai definícióval

5.4.2. Algebrai tulajdonságok

- $$[0,1] - [0,1] = [-1,1]$$

  A kivonás és osztás nem a szorzás és összeadás inverze.

- A szorzás nem disztributív az összeadásra nézve, viszont van szubdisztributivitás:
$$A(B+C) \subseteq AB + AC$$

Példa.

$$x(x-1) = x^2 - x \qquad x \in [0,1]$$
$$[0,1]([0,1]-1) = [0,1][-1,0] = [-1,0]$$
$$[0,1]^2 - [0,1] = [0,1] - [0,1] = [-1,1]$$

- $+, \cdot$ asszociatívak és kommutatívak

$$\Phi(x) \supseteq \{\phi(x) | x \in X\}$$
$$\sin, \cos, \exp, \sqrt{\ }$$
$$\exp(X) = [e^{\underline{\mathrm{x}}}, e^{\bar{x}}]$$
$$X = [\underline{\mathrm{x}}, \bar{x}] \quad \underline{\mathrm{x}} \leq \bar{x}; x, \bar{x} \in \mathbb{R}$$
$$\sqrt{X} = [\sqrt{\underline{\mathrm{x}}}, \sqrt{\bar{x}}]$$

Definíció. Egy $f : \mathbb{R}^n \to \mathbb{R}$ függvényhez definiált tartalmazási függvény:
$$F : \mathcal{I}^n \to \mathcal{I}$$
ami teljesíti, hogy
$$x \in X \Rightarrow f(x) \in F(x); \qquad f(X) = \{f(x) | x \in X\} \subseteq F(X)$$

Ha f elemi függvények elemi aritmetikai műveletekkel képezett függvénye, akkor egy tartalmazási függvényt kapunk.

Tegyük fel, hogy ki tudjuk értékelni a tartalmazási függvényt a gradiensen, bármely intervallumon.
$$F'(x); \nabla F(X)$$

$$f(y) \geq f(x) - L\,\|x-y\|$$
$$f(x) \leq f(\hat{x}) \pm L\,\|\bar{x} - \underline{\mathrm{x}}\| \quad \hat{x} \in X$$
$F_C(X) = f(\hat{x}) + F'(X)(X - \hat{x}) =$ centrális alak, vagy középérték-alak ($\hat{x}$ az X középső pontja)
$$f(x) = f(\hat{x}) + f'(\xi)(x - \hat{x}) \quad \xi \in [\hat{x}, x]$$
$$f(x) \in f(\hat{x}) + F'(X)(X - \hat{x})$$
$$f(X) \subseteq f(\hat{x}) + F'(X)(X, \hat{x})$$

Bauman–alak: Adott X, legjobb $\hat{x}$, amelyik a legjobb lsó korlátot adja és a legrosszabb felsőt.

$$\frac{b - \underline{\mathrm{x}}}{\bar{x} - b} = \frac{U}{L}$$
$$b - \underline{\mathrm{x}} = \frac{U}{L}(\bar{x} - b)$$
$$v\left(1 + \frac{U}{L}\right) = \underline{\mathrm{x}} + \bar{x}\frac{U}{L}$$
$$b = L\underline{\mathrm{x}} + U\bar{x}$$

$$b_- = \frac{L\bar{x} + U\underline{x}}{L - U}$$
$$F_b(x) = f(b) + F'(x)(x - b)1$$
$$b_+ = \frac{L\bar{x} + U\underline{x}}{U - L}$$
$$c - b_+ = b_- - c$$

Példa.

$$f(x, y) = xy + e^- \sqrt{y}$$
$$X = [-1, 2]; \qquad Y = [1, 3]$$

Naív tartalmazás:

$$F(X, Y) = [-1, 2][1, 3] + [e^{-1}, e^2] - [0, \sqrt{3}] = [-3 + \frac{1}{e} - \sqrt{2}, 5 + e^2]$$

$$\nabla f(x, y) = \begin{pmatrix} y + e^x \\ x - \frac{1}{2\sqrt{y}} \end{pmatrix}$$
$$\nabla F(X, Y) = \begin{pmatrix} [1, 3] + [e^{-1}, e^2] \\ [-1, 2] - [\frac{1}{2\sqrt{3}}, \frac{1}{2}] \end{pmatrix} = \begin{pmatrix} [1 + e^{-1}, 3 + e^2] \\ [-1.5, 2 - \frac{1}{2\sqrt{3}}] \end{pmatrix}$$
$$(\hat{x}, \hat{y}) == 0, 1) \qquad f(\hat{x}, \hat{y}) = 0$$
$$F(X, Y) = 0 + \begin{pmatrix} [1 + \frac{1}{e}, 3 + e^2] \\ [-1.5, 2 - \frac{1}{2\sqrt{3}}] \end{pmatrix} ([-1, 2], [0, 2]) = \cdots = [-6 + e^2, 10 - \frac{1}{\sqrt{3}} + 2e^2]$$

Példa.

$$\sin(\pi x^3 y)(y - 1) \quad X = [0, 1/2]; Y = [0, 1]$$

$$\begin{aligned} F(X) &= \sin(\pi[0, 0.5]^3[0, 1]) \cdot ([0, 1] - [1, 1]) = \\ &= \sin(\pi[0, \frac{1}{8}][0, 1])[-1, 0] = \\ &= \sin([0, \pi])[-1, 0] = [0, 0.39][-1, 0] = [-0.39, 0] \end{aligned}$$

5.5. Autómata differenciálás

$f'(x)$

- numerikus deriválás; probléma: hibák
- szimbolikus deriválás; probléma: lassú, beírásoknál hiba.

Láncszabály:

$$f(x) = g(h(x)) \Rightarrow f'(x) = \frac{\mathrm{d}g}{\mathrm{d}h} \frac{\mathrm{d}h}{\mathrm{d}x}$$

Haladó mód: Először $\frac{\mathrm{d}h}{\mathrm{d}x}$-et, majd később $\frac{\mathrm{d}g}{\mathrm{d}h}$-t számítjuk ki. Fordított mód: Ugyanez, de fordítva Haladó mód:

Példa.

$$f(x_1, x_2) = x_1 x_2 + \sin x_1$$

$$\begin{aligned} w_4' &= \cos(w_1) \cdot w_1' \\ w_3' &= w_1' w_2 + w_2' w_1 \\ w_5' &= w_4' + w_3' w \end{aligned}$$

Legyen $(x_1, x_2) = (0, 2)$, ekkor $w_1 = 0, w_1' = 1, w_2 = 2, w_2' = 0$.

$$w_1' = 1; \qquad w_3' = 2 + 0 = 2; \qquad w_5' = 3$$

Fordított mód:

$$\begin{aligned} v_5' &= \frac{\partial f}{\partial v_5} = 1 \\ v_3' &= v_5' \cdot \frac{\partial v_5}{\partial v_3} = v_5' \cdot 1 = 1 \\ v_4' &= v_5' \cdot \frac{\partial v_5}{\partial v_4} = 1 \\ v_2' &= v_3' \frac{\partial v_3}{\partial v_1} = v_3' \cdot v_1 = v_1 \quad x_2' = x_1 \end{aligned}$$

$$\left.\begin{aligned} v_1^{a)} &= v_4' \cdot \tfrac{\partial v_4}{\partial v_1} = v_1 \\ v_1^{b)} &= v_3' \cdot v_2 = v_2 \end{aligned}\right\} \Rightarrow \begin{aligned} v_1' &= v_1^{a)} + v_1^{b)} = \cos v_1 + v_2 \\ x_1' &= \cos(x_1) + x_2 \end{aligned}$$

1. kör: Számítsuk ki a formulákat a v_i'-kre és számítsuk ki az értékeket a csúcsoknál.
2. kör: Számítsuk ki a deriváltakat

Adott (u, u') párra operátor felülbírálást definiálhatunk:

$$\begin{aligned} (u, u') \pm (v, v') &= (u \pm v, u' \pm v') \\ (u, u')(v, v') &= (uv, u'v + uv') \\ \sin(u, u') &= (\sin(u), \cos(u)u') \\ (u, u')^n &= (u^n, nu^{n-1}u') \end{aligned}$$

Példa.

$$f(x) = \sin(\pi x^3), \quad x = [0, 0.5]$$

$$x \to ([0, 0.5], 1)$$

$$x^3 \to ([0, 0.5], 1)^3 = ([0, 0.125], 3[0, 0.5]^2) = ([0, 0.125], [0, 0.75])$$

$$\pi \to ([\pi, \pi], 0)$$

$$\pi x^3 \qquad ([\pi, \pi], 0)([0, 0.125], [0, 0.75]) = ([0, 0.125\pi], [0, 0.75\pi])$$

$$\sin([0, 0.125\pi], [0, 0.75\pi]) = ([0, \sin(0.125\pi)] \cos([0, 0.125\pi])[0, 0.75\pi]) = ([0, 0.35], [0, 2.36])$$

6. Intervallum Newton módszer

Hagyományos Newton-módszer:

$$f(x) = 0; \qquad 0 = f(x) \approx f(x_0) + f'(x_0)(x - {}_0)$$

$$x \approx x_0 - \frac{f(x_0)}{f'(x_0)}$$

Iteráció:

$$x_{k+1} := x_k - \frac{f(x_k)}{f'(x_k)}$$

Megjegyzés. Amikor minimumot keresünt, akkor $f'(x) = 0$ kell, így:

$$x_{k+1} := x_k - \frac{f'(x_k)}{f''(x_k)}$$

Intervallokokra:

$$X_{k+1} = (m(X_k) - f'(m(X_k))) \cap X_k$$

$$m(X) = \frac{\underline{\mathrm{x}} + \bar{x}}{2}$$

$$f'(x) \in f'(x_0) + F''(x_0)(x - x_0)$$

$F''(x_k)$ lehet $(-\infty, \infty)$ is, ilyenor X_{k+1} nem véges, nem megoldható.

Ha $x \in X$, $f'(x) = 0$, akkor $x \in X_{k+1}$.

Ha $X_{k+1} = \emptyset$, akkor $\nexists x \in X_k$, hogy $f'(x) = 0$.

Példa.

$$f(x) = x^2 - x$$

$$X_0 = [0, 1]$$

$$f'(x) = 2x - 1 \qquad f''(x) = 2$$

$$X_1 = (0.5 - \frac{0}{[2, 2]}) \cap [0, 1] = [0.5, 0.5]$$

A minimum $[0, 1]$-en: 0.5

$$X_0 = [1, 2]$$

$$X_1 = (1.5 - \frac{2}{[2, 2]}) \cap [1, 2] = [0.5, 0.5] \cap [1, 2]$$

Nincs minimum $[1, 2]$-ben.

Példa.

$$f(x) = x^4 - 2x^3 + x^2$$

$$f'(x) = 4x^3 - 6x^2 + 2x$$

$$f''(x) = 12x^2 - 12x + 2 = 12x(x - 1) + 2$$

$$X = [0, 2]$$

$$F''(X) = 12[0,2][-1,1] + 2 = 12[-2,2] + 2 = [-22,26]$$

$$X_1 = (1 - \frac{0}{[-22,26]}) \cap [0,2] = (-\infty,\infty) \cap [0,2] = [0,2]$$

$$X = [0.9, 1.1]$$

$$F''(X) = [-1.48, 6.02]$$

$$X_0 = [0.99, 1.01] \qquad F''(X) = [1.64, 2.37]$$

$$X_1 = [0.9989, 1.0021]$$

Állítás. Ha $X_{k+1} \subsetneq X_k$, akkor X_{k+1} tartalmaz pontosan egy optimumot (max. vagy min.)

7. Deriválást nem igénylő eljárások

Tegyük fel, hogy egy $f : \mathbb{R}^n \to \mathbb{R}$ függvény optimumát keressük. Előfordulhat, hogy a deriváltra vonatkozóan nem áll rendelkezésre információ, ez a rész néhány ebben az esetben használható eljárást ismertet.

7.1. Nelder–Mead-algoritmus

A most bemutatott eljárás robusztus, barátságos geometriai leírással bír, így meglehetősen népszerű.

A $P = \{p_0, \ldots, p_n\}$ iteratív halmazt szimplexnek nevezzük, mert $n+1$ pontot tartalmaz egy n-dimenziós térben. Legyen x_0 egy tetszőleges kezdőpont, ekkor az induló P ponthalmaz:

$$P := \{x_0, x_0 + \delta e_1, \ldots, x_0 + \delta e_n\}$$

ahol δ egy skálázási faktor, e_i pedig az i. egységvektor. Az alábbi összetevők lesznek fontosak az algoritmus futása során:

- A P ponthalmaz két „legrosszabb" pontja:

  $$p_{(n)} = \arg\max_{p \in P} f(p); \qquad p_{(n-1)} = \arg\max_{p \in P \setminus p_{(n)}} f(p)$$

  valamint a „legjobb" pontja:

  $$p_{(0)} = \arg\min_{p \in P} f(p)$$

- A „legmagasabb" pontot kivéve az összes pontra vett:

  $$c = \frac{1}{n} \sum_{i \neq (n)} p_i$$

  centroid.

- Egy u.n. *reflexiós* lépés során generált próbapont:

  $$x^{(r)} = c + (c - p_{(n)}$$

- Sikeres reflexiós lépést követően egy u.n. *nyújtási* lépés során generált próbapont:

  $$x^{(e)} = c + 1.5(c - p_{(n)})$$

- Ha a fenti két próba nem ígéretes, a szimplexet egy u.n. *többszörös összehúzás* során a minimális

  $$p_i := \frac{1}{2}(p_i + p_{(0)}), i \in \{0, \ldots, n\}$$

  pont felé zsugorítjuk.

Példa. Tekintsük az $f(x) = 2x_1^2 + x_2^2 - 2x_1x_2 + |x_1 - 3| + |x_2 - 2|$ függvényt. Legyen a kiinduló szimplex a $p_0 = (1,2)^T, p_1 = (1,0)^T, p_2 = (2,1)^T$ pontokkal adva. Az algoritmus először egy reflexiós lépést tesz, az új pont $p_{(1)}$. Azonban a következő iterációban a reflexiós pont se az $f_{(0)} < f(x^{(r)} < f_{(n-1)}$ sem az

$f(x^{(r)}) < f_{(0)}$ feltételeket nem elégíti ki, így a kontrakciós pontot számítjuk ezután. Mivel ennél a függvényérték jobb mint $f(x^{(r)})$, így $p_{(n)}$-t kicseréljük erre a pontra. Azt is láthatjuk, hogy $f(x^{(c)}) < f_{(n-1)}$. Megfigyelhető, hogy ha az optimum a politópon belül van, akkor annak mérete csökken, ezzel közelítve a kilépési feltétel teljesüléséhez.

Algorithm 3 Nelder–Mead

$k = 0$; $P = \{p_0, \ldots, p_n\}$ ahol $p_0 = x_0, p_i = x_0 + \delta e_i, \quad i \in 1, \ldots, n$
Értékeljük ki f-et a $p_1, \ldots, p_n$ pontokban.
Határozzuk meg a $p_{(n)}, p_{(n-1)}$ és $p_{(0)}$ pontokat P-ben. (rendezés)
while $(f_{(n)} - f_{(0)} > \varepsilon)$ **do**
  $c = \frac{1}{n} \sum_{i \neq (n)} p_i$
  $x^{(r)} := c + (c - p_{(n)})$, kiértékeljük $f(x^{(r)})$-t.
  if $(f_{(0)} < f(x^(r)) < f_{(n-1)})$ **then**
    $P = P \setminus \{p_{(n)}\} \cup \{x^{(r)}\}$
  if $(f(x^(r)) < f_{(0)})$ **then**
    $x^{(e)} = c + 1.5(c - p_{(n)})$, kiértékeljük $f(x^{(e)})$
    $P = P \setminus \{p_{(n)}\} \cup \{\arg\min\{f(x^{(e)}), f(x^{(r)})\}\}$
  if $(f(x^{(r)}) \geq f_{(n-1)})$ **then**
    $x^{(c)} = c + 0.5(c - p_{(n)})$, kiértékeljük $f(x^{(c)})$
    if $(f(x^{(c)}) < f(x^{(r)}) < f_{(n)})$ **then**
      $P = P \setminus \{p_{(n)}\} \cup \{x^{(c)}\}$
    else
      if $(f(x^{(c)}) > f(x^{(r)}))$ **then**
        $P = P \setminus \{p_{(n)}\} \cup \{x^{(r)}\}$
      else
        $p_i = \frac{1}{2}(p_i + p_{(0)}), \quad i = 0, \ldots, n$
        Kiértékeljük $f(p_i) \quad i = 1, \ldots, n$
        $P = \{p_0, \ldots, p_n\}$
  $k = k + 1$

7.2. Powell

Ebben az eljárásban irányok egy $(d_1, \ldots, d_n)$ halmazát bővítjük iteratívan, annak érdekében hogy az x^* felé mutató irányt közelítsük. Egy x_0 kezdőpontból indulunk, amit ezúttal jelöljünk $x_1^{(1)}$-gyel. Minden iterációban n lépést teszünk az n irányt felhasználva. Minden lépésben $x_{i+1}^{(k)} = x_i^{(k)} + \lambda d_i$, ahol k az iteráció sorszáma. A λ lépésközről feltesszük hogy optimális, azaz

$$\lambda = \arg\min_{\mu} f(x_i^{(k)} + \mu d_i).$$

Az irányok halmazát a koordinátairányokkal inicializáljuk, azaz $d_i = e_i$ kezdetben. Az irányokat az alábbi módon frissítjük: $d = x_{n+1}^{(k)} - x_1^{(k)}$ a végső irány a k. iterációban. Legyen a következő kezdőpontunk ebben az irányban:

$$x_1^{(k+1)} = x_{n+1}^{(k)} + \lambda d$$

ahol λ az optimális lépésköz. A régebbi irányokat eltoljuk: $d_i = d_{i+1}, i \in 1, \ldots, n-1$. Az utolsó irány a legújabb lesz: $d_n = d$. Az iterációt folytatjuk, egészen addig amíg $\left| f(x_1^{(k+1)} - f(x_1^{(k)})) \right| < \varepsilon$

Algorithm 4 Powell

$k = 0$, $(d_0, \ldots, d_n) = (e_0, \ldots, e_n)$ és $x_1^{(1)}$
repeat
  $k = k + 1$
  for $(i \in \{1, \ldots, n\})$ **do**
    $\lambda = \arg\min_\mu f(x_i^{(k)} + \mu d_i)$
    $x_{i+1}^{(k)} = x_i^{(k)} + \lambda d_i$
  $d = d_{n+1}^{(k)} - x_1^{(k)}$
  $x_1^{(k+1)} = x_{n+1}^{(k)} + \lambda d$, ahol $\lambda = \arg\min_\mu f(x_{n+1}^{(k)} + \mu d)$
  $d_i = d_{i+1}, \quad i = 1, \ldots, n, \quad d_n = d$
until $(\left| f(x_1^{(k+1)}) - f(x_1^{(k)}) \right| < \varepsilon)$

8. Eljárások korlátozott optimalizálásra

Egy általános nemlineáris programozási feladatot felírhatunk az

$$\begin{aligned} &\min f(x) \\ &g_i(x) \leq 0 && i = 1, \ldots, p \text{egyenlőtlenség kényszerek} \\ &g_i(x) = 0 && i = p+1, \ldots, \text{egyenlőségi kényszerek} \end{aligned}$$

Amikor egy ilyen korlátozott feladatot akarunk megoldani, akkor két lehetőségünk van. Vagy beépítjük a kényszereket a célfüggvénybe, ezáltal nemkorlátozott feladattá átalakítva az eredetit (amely bár nem ekvivalens azzal, de jó paraméterekkel a megoldás tart az eredetihez), vagy közvetlenül korlátozzuk a keresést a megengedett területre.

Ebben a részben a korlátok célfüggvénybe építésére látunk példát, majd a Gradiens vetítés módszerével ismerkedünk meg, amely példa a második útra.

8.1. Bűntető-, és korlátozó függvények módszere

8.1.1. Bűntetőfüggvény módszer

A módszer alapötlete a nem megengedett területek bűntetése egy

$$p_\mu(x) = \mu \left(\sum_{i=1}^{p} \max\{g_i(x), 0\} + \sum_{i=p+1}^{m} |g_i(x)| \right)$$

vagy

$$p_\mu(x) = \mu \left(\sum_{i=1}^{p} \max\{g_i(x), 0\}^2 + \sum_{i=p+1}^{m} \left|g_i^2(x)\right| \right)$$

u.n. bűntető függvény segítségével. Ezek a függvények 0-k a megengedett részeken, azonban pozitívak a tiltott részeken. A bűntetőfüggvény céllfüggvényhez adásával minden μ-re egy nemkorlátos feladatot kapunk, amelynek minimalizáló pontja egy közelítése lesz az eredeti megoldásnak, ha μ kellően nagy. A módszerrel több probléma is van

- Nem tudjuk előre, mekkora μ fog kelleni, azonban
- ha μ túl nagy, a feladat rosszul kondíciónált lesz.

A probléma kiküszöbölésére μ értékét fokozatosan növeljük, és a következő minimumkeresést az előző eredményéből indítjuk. Ezáltal a konvergencia gyorsabb lesz, és a rosszul kondíciónáltságot is elkerüljük. Kilépünk, ha az aktuális közelítésre $p_\mu(x^*(\mu)) \leq \varepsilon$, ekkor $x^*(\mu)$-t elfogadjuk egy közelítő megoldásra.

8.2. Korlátozófüggvény-módszer

Minden igyekezet ellenére a bűntetőfüggvény-módszer olykor nem megengedett megoldást ad. Így ha az alkalmazás szigorúan megköveteli a megengedettséget, akkor más módszerre van szükségünk. Az ötlet: Adjunk meg egy olyan függvényt, amely egy „gátat” szab a korlátoknál, így x_k csak a megengedett

Algorithm 5 BűntetőEljárás$(f, g, p, \mu_0, \beta, \varepsilon)$

$k = 1$
$x_k = \arg\min P_\mu(x)$
while $(p_\mu(x_k) > \varepsilon)$ **do**
  $k = k + 1$
  $\mu_k = \beta \cdot \mu_{k-1}$
  $x_k = \arg\min_{x=X} P_\mu(x)$

tartományban lehet. (Emiatt ez a módszer csak egyenlőtlenségi korlátok esetén használható).

Például

$$b_\mu(x) = -\mu \sum_{i=1}^{p} \frac{1}{g_i(x)}$$

és

$$b_\mu(x) = -\mu \sum_{i=1}^{p} \ln(-g_i(x))$$

pozitív értéket ad a szigorúan megengedett pontokra, végtelent ha valamely korlát éles. A korlátozófüggvényt a nem megengedett pontokban nem szükséges definiálnunk. A $B_\mu(x) = f(x) + b_\mu(x)$ új célfüggvényt minimalizálva kapjuk a közelítést. Az algoritmus alapvetően ugyanaz mint a bűntető-függvényeknél, azonban μ-t csökkentjük a növelés helyett (a határon továbbra is nagy lesz). Joggal vehetjük észre, hogy ezzel a módszerrel továbbra is figyelembe kell vennünk bizonyos kényszereket, de az új feladatra ezek egyike sem éles, így bármilyen nemkorlátos eljárást használhatunk, bizonyos óvintézkedések megtétele után.

Algorithm 6 KorlátozóEljárás$(f, g, b, \mu_0, \beta, \varepsilon)$

$k = 1$
$x_k = \arg\min B_\mu(x)$
while $(b_\mu(x_k) > \varepsilon)$ **do**
  $k = k + 1$
  $\mu_k = \frac{\mu_{k-1}}{\beta}$
  $x_k = \arg\min_{x \in X} P_\mu(x)$

8.3. Gradiens vetítés módszer

Ez az eljárás a legmeredekebb ereszkedés módszerének korlátozott problémák megoldására szolgáló módosítása. Minden lépésben, az új irányt úgy módosítjuk, hogy még a megengedett régióban maradjunk, oly módon, hogy a gradienst az aktív korlátokra vetítjük.

A vetítést egy P projekcióval végezzük, $r = -P\nabla f$ módon. Ha M az aktív korlátok Jacobi-mátrixa (azaz oszlopai $\nabla g_i(x)$, azokra a g_i-kre, melyekre $g_i(x) = 0$), akkor

$$P = I - M(M^T M)^{-1} M^T$$

ugyanis tudjuk, hogy minden aktív korlátra r merőleges a korlát gradiensére, azaz $\nabla g_i^T r = 0$. Így

$$M^T r = 0$$

. A legmeredekebb ereszkedés irányát a

$$\begin{aligned} \min & r^T \nabla f \\ & M^T r = 0 \\ & \|r\|_2 = 1 \end{aligned}$$

probléma megoldásával kapjuk meg. A Lagrange–függvényt használva

$$L(r, u, v) = r^T \nabla f + r^T M u + v r^T r$$

ahol $u \in \mathbb{R}^n, v \in \mathbb{R}, \|r\|_2^2 = r^T r$. Az optimalitás szükséges feltétele

$$\frac{\partial L}{\partial r} = \nabla f + M u + 2 v r = 0$$

Beszorozva M^T-al és figyelembe véve hogy $M^T r = 0$

$$M^T \nabla f + M^T M u + 2 v M^T r = M^T \nabla f + M^T M u = 0$$

amiből

$$u = -(M^T M)^{-1} M^T \nabla f$$

Ezt visszaírva az eredeti egyenletbe megkapjuk az

$$r = -\frac{1}{2v}(E - M(M^T M)^{-1} M^T) \nabla f$$

Az $\frac{1}{2v}$ szorzótól eltekinthetünk, r egy iránynak felel meg.

Ha $r = 0$ és $u \geq 0$, akkor a Kunh–Tucker-feltételek állnak, így KKT pontot találtunk. Ha valamelyik Lagrange-multiplikátor negatív, akkor továbbra is találhatunk csökkenő irányt bizonyos negatív u_i-vel bíró korlátok elhagyásával. A negatív u_i jelentése ugyanis, hogy a megfelelő korlát nem éles az ereszkedési irányra. Általában a legkisebb u_i-vel rendelkező korlátot hagyjuk el. Ha $r \neq 0$, akkor megtaláltuk az ereszkedési irányt. Egyébként még több korlátot hagyhatunk el. Ha már nincs több korlát, de $r = 0$, akkor megállhatunk, elértünk egy KKT pontot.

Miután az r megengedett irányt megtaláltuk, meghatározzuk az optimális lépésközt

$$\lambda = \arg\min_{\mu > 0} f(x_k + \mu r) \leq 0$$

úgy, hogy a következő iteráció kielégítse a nem éles feltételeket, azaz $g_i(x_k + \lambda r) \leq 0$. Valójában az első olyan korlát, amely éles lesz az r irány mentén, határozza meg a maximális lépésközt $\lambda_{\max}$-ot. Speciálisan egy $a_i^T x - b_i \leq 0$ lineáris korlát esetén λ-nak ki kell elégítenie az $a^T(x_k + \lambda r) - b_i \leq 0$ egyenlőtlenéget, úgy hogy $\lambda_{\max} \leq \dfrac{b_i - a_i^T x_k}{a_i^T r}$

A λ kiszámítására nemlineáris korlátok esetén azok helyett számolhatunk azok lineáris közelítésével. nemlineáris korlátoknál arra is szükség lehet, hogy egy visszaállítási lépésben gondoskodjunk róla, hogy az új pont nem sérti meg az éles korlátokat.

A legmeredekebb ereszkedés vetítése általánosítható más ereszkedési irányt használó eljárásokra is. Annyit kell tennünk, hogy ∇f helyett a használni kívánt iránnyal implementáljuk az algoritmust.

Algorithm 7 GradProj(f, g, x_0, ε)

```
k = 0
repeat
  r = -(I - M(M^T M)^{-1} M^T) ∇f
  while (r = 0) do
    u = -(M^T M)^{-1} M^T ∇f
    if (min_i u_i < 0) then
      g_i-t eltávolítjuk az aktív korlátok közül.
    else
      return x_k (egy KKT pont)
  λ = arg min_μ f(x_k + μr)
  if (∃i g_i(x_k + λr) < 0) then
    Határozzuk meg λ_max-ot.
    λ = λ_max
  x_{k+1} = x_k + λr
  k = k + 1
until (|x_k - x_{k-1}| < ε)
```

8.4. Inegzakt vonalmenti keresés

Majdnem minden ereszkedési irányt használó eljárásban szükséges minden lépésben egy vonalmenti keresés. Eddig csak optimális lépésközt használtunk, így egzakt keresést tételeztünk fel. Az is igaz, hogy kvadratikus függvényekre ezt kiszámolni könnyű. Egyéb esetben egydimenziós optimalizáló eljárást használhatunk. Ha a minimumtól még távol vagyunk, az optimális lépésköz „túl jó közelítése" általában nem hatékony. A kérdés, honnan tudhatjuk, hogy mennyire messze vagyunk, és a közelítésünk már elegendő? Pontos válasz nincs a kérdésre, de néhány szabályt alkalmazhatunk. Például gyaníthatjuk, hogy $\|\nabla f(x)\| \to 0$, ahogy $x \to x^*$. Hogy elkerüljük a túl nagy, vagy túl kicsi lépésközt, a célfüggvény megfelelő csökkentésére van szükség. Egy kicsi $0 < \alpha < 1$-re

$$\begin{aligned} f_k + (1-\alpha)\lambda \nabla f_k^T r_k <& f(x_k + \lambda r_k) \\ & f(x_k + \lambda r_k) < f_k + \alpha\lambda \nabla f_k^T r_k \end{aligned}$$

teljesülése szükséges. A $\varphi_{r_k}(\lambda) := f(x_k + \lambda r_k)$ jelölést használva, a két egyenlőtlenséget együtt a

$$\varphi_{r_k}(0) + (1-\alpha)\varphi'_{r_k}(0)\lambda < \varphi_{r_k}(\lambda) < \varphi_{r_k}(0) + \alpha\varphi'_{r_k}(0)\lambda$$

alakban írhatjuk. Ezt Goldstein feltételnek nevezzük. A második egyenlőtlenséget, ha magában szerepel Armijo-egyenlőtlenségnek hívjuk. A két egyenlőtlenség ad egy alsó $\underline{\lambda}$ és egy felső $\bar{\lambda}$ korlátot λ-ra. Ez jelenthet több nem összefüggő intervallumot és az optimális megoldás akár kívül is eshet ezeken. Ezt elkerülendő további feltételre van szükségünk: A gradiens az új pontban kisebb legyen, mint a régiben. Az elõzõ jelölésekkel tehát

$$\varphi_{r_k}(\lambda) \leq \sigma\varphi'_{r_k}(0)$$

vagy másképp

$$\nabla f(x_k + \lambda r_k)^T r_k < \sigma \nabla f(x_k)^T r_k$$

Ezt a kritériumot Wolfe-feltételnek nevezzük. Az Armijo- és Wolfe-feltételt együtt szokás Wolfe-feltételeknek nevezni.

A gyakorlatban általában egy visszakövető vonalkeresést végzünk, amíg a kiválasztott feltételek nem teljesülnek. Ennek koncepciója az alábbi: Adott (akár nagy) kezdő lépésköz λ_0, csökkentsük arányosan egy $0 < \beta < 1$ faktorral, amíg a kiválasztott feltételek nem teljesülnek

---

$k = 1$
while (feltételek nem teljesülnek) **do**
  $\lambda_k = \beta\lambda_{k-1}$
  $k = k + 1$

---

9. Determinisztikus GO algoritmusok

9.1. Determinisztikus heurisztika, DIRECT

A heurisztikákra gyakran gondolunk úgy, mint olyan algoritmusokra, melyek adnak egy közelítő megoldást, mindenféle garancia nélkül, hogy az közel van az optimumhoz. A globális optimalizálásban heurisztikák gyakran párosulnak véletlen keresési technikákkal. Egy könnyen érthető példa a rács menti keresés. Hatékonysága könnyen elemezhető, a kiértékelt pontok száma exponenciálisan nő a dimenzióval.
Ami általános a heurisztikákban, nyerni próbálunk a lokális és globális keresésen egy megengedett területen.

A DIviding RECtangles (felosztó téglalapok) algoritmus előre meghatározott N számú mintapontot generál egy tégla által határolt megengedett területen elhelyezkedő rács felett a skálázott

$$x_1 = \frac{1}{2}(1, 1, \ldots, 1)^T$$

középpontból indulva. Ezt követően az x_k pont finomítása az x_k egy környezetében való újabb mintavételezést jelent.

Hogy eldöntsük mik az érdekes területek, minden mindtaponthoz egy (akár változó) u_k sugárvektort tárolunk, hogy leírja az $(x_k - u_k, x_k + u_k)$ téglát. Az u_k hossza és az $f(x_k)$ függvényérték határozza meg, hogy x_k jelölt-e finomításra, vagy sem. Egy α paraméter szabályozza a lokális kontra globális nyereséget. Az algoritmust három döntés jellemzi

- Hogyan válasszuk meg a finomításra szánt pontokat
- Hogyan mintavételezzünk a kiválasztott pont körül
- Hogyan frissítsük az u_k információt.

9.1.1. Kiválasztás és finomítás

A kiválasztás módja megad minden iterációban egy M számú listát a már kiszámolt u_i vektorok méreteivel. Ezt tárolhatjuk rendezve. Minden pontban a hozzárendelt u_k vektor valamely s_j által meghatározott S_j méretosztályba esik. Az alapötlet az, hogy új pontok generálásával az aktuális mintapontok kisebb méretekhez kerülnek. Az előforduló méretek nem ekvidisztánsak, de u_k frissítésének módja miatt egy meghatározott mintázatot követnek. Minket a viszonylag alacsony pontok ($f(x_k)$ kicsi) érdekelnek és azok is a viszonylag feltáratlan részeken ($\|u_k\|$ nagy). Egy Pareto-féle módon minden nemdominált pontot kiválasztunk finomításra. Első körben ez minden olyan pont kiválasztását jelenti, ahol $m_j = \min_{k \in S_j} f(x_k)$. Itt az α paraméter arra szolgál, hogy ne legyen túl lokális a mintavételezés. Az $f^U - \alpha \left| f^U \right|$ értéket jegyezzük meg, $f^U = \min_k f(x_k)$. Ezt a pontot az u.n. nemdominált pontokhoz vesszük. Egy egyenest képzeljünk ebből a pontból felfeé úgy, hogy a kapott görbe konvex maradjon. Lássuk hogyan kivitelezhető ez. A legnagyobb méretosztályból S_1 indulunk, kiválasztjuk az m_1 minimumnak megfelelő pontot és végigmegyünk a j osztályon egészen

$$m_j \geq f^U - \alpha \left| f^U \right| + \frac{s_j}{s_{j-1}}(m_{j-1} - f^U + \alpha \left| f^U \right|)$$

esetig hogy m_j az egyenes fölött legyen az $f^U - \alpha \left|f^U\right|$ és m_{j-1}. Ez azt jelenti hogy az utolsó legalacsonyabb pont nem feltétlenül lesz finomítva, mert a körülötte lévő tér nem elég üres. Ahogy már szerepelt, ez az α csak az paraméterrel szabályozható.

Algorithm 8 select$(f_1, \ldots, f_k, \|u_1\|, \ldots, \|u_k\|, \alpha$

Határozzuk meg $f^U = \min_l f_l$-t.
Rendezzük az $\|u_1\|, \ldots, \|u_k\|$ listát és adjuk meg az $S_1, \ldots, S_M$ osztályokat $s_1 > s_2 > \cdots > s_M$
$m_1 = \min_{k \in 1} f_k, \quad j = 1$
repeat
  kiválasztjuk $\arg\min_{k \in S_j} f_k$-t
  $j = j + 1, \quad m_j = \min_{k \in S_j} f_k$
until $(m_j \geq f^U - \alpha \left|f^U\right| + \frac{s_j}{s_j - 1}(m_{j-1} - f^U + \alpha \left|f^U\right|))$

9.1.2. Finomítás és négyzetek frissítése

Egy pont finomításán az $(x-u, x+u)$ hipertéglából való további mintavételezést értünk. Továbbá a régi x mintapont a további új mintapontokkal együtt kisebb sugárvektort kap, mint u.

Algorithm 9 refine$(x, u,$**global** $k,N)$

Határozzuk meg $I = \arg\max_i u_i$
for $(i \in I)$ **do**
  Értékeljük ki az $f(x - \frac{2}{3}u_i e_i)$ és $f(x + \frac{2}{3}u_i e_i)$
  $w_i = \min\{f(x - \frac{2}{3}u_i e_i), f(x + \frac{2}{3}u_i e_i)\}$
  $k = k + 2$
  if $(k = k + 2)$ **then**
    STOP
for $(i \in I)$ **do**
  $v_i = u$
repeat
  kiválasztjuk $\eta = \arg\max_{i \in I} w_i$-t
  for $(i \in I)$ **do**
    $v_{i\eta} = \frac{1}{3}u_\eta$
    $u_\eta = \frac{1}{3}u_\eta$
  Tároljuk $x_{k-1} = x - \frac{2}{3}u_\eta e_\eta, u_{k-1} = v_\eta$
  Tároljuk $x_k = x + \frac{2}{3}u_\eta e_\eta, u_k = v_\eta$
until $(I = \emptyset)$

9.1.3. DIRECT

Algorithm 10 DIRECT(f, α, N)

$k = 1, x_1 = \frac{1}{2}(1, 1, \ldots, 1)^T, u_1 = \frac{1}{2}(1, 1, \ldots, 1)^T, f_1 = f(x_1)$
repeat
  $J = \text{select}(f_1, \ldots, f_k, \|u_1\|, \ldots, \|u_k\|, \alpha)$
  for $(j \in J)$ **do**
    $\text{refine}(x_j, u_j, k, N)$
until (STOP)

10. Sztochasztikus GO: PRS, Multistart, Szimulált lágyítás

Sztochasztikus eljárások alatt az olyan algoritmusokat értjük, melyek (ál)véletlen számok segítségével generálnak új próbapontokat. Két alapvető megközelítést vizsgálunk meg: Pure Random Search (tiszta véletlen keresés), valamint Multistart. Ezt a szimulált lágyítás heurisztika egy klasszikus változata követi.

10.1. Pure Random Search (PRS)

Egyenletes eloszlásban generáljuk a pontokat a tartományon, és a legkisebb értéket adót eltároljuk mint a globális minimum pont egy közelítését. Kedvelt referenciaalgoritmus, mivel könnyű elemezni. Megmutatható, hogy néhány könnyed feltevés után az

$$y_{(1)} = \min\{f(x_1), \ldots, f(x_N)\}$$

valószínűségi változó eloszlásfüggvénye

$$F_{(1)}(y) = 1 - (1 - \mu(y))^N$$

ahol $\mu(y)$ az y-hoz tartozó nívóhalmaz mértéke, $\mu(y) = P(f(x) \leq y)$.

Egy korai megfigyelés szerint annak valószínűsége hogy az $N + 1$. húzással jobb pontot találunk ha már N húzás volt, $\frac{1}{N+1}$, és K újabb pontra ez a valószínűség $\frac{K}{N+K}$-ra nő.

A sztochasztikus folyamatokra igaz, hogy ha $N \to \infty$, akkor a végén a globális optimumot megtaláljuk. Ez persze csak elméletileg nyugtathat meg minket, valójában célunk elérésének valószínűsége nagyban függ a sikerterület méretétől. Egy klasszikus megközelítés ennek a növelésére lokális keresés használata. Ezt az eljárást *multistart*nak nevezzük.

10.2. Multistart

Legyen $\text{LS}(x) : X \to X$ olyan lokális optimalizáló rutin, amely egy megadott kezdőpontból visszaad egy pontot a tartományban, ami közelítése egy lokális minimumnak. A multistart véletlen generált kezdőpontokból LS-sel generált pontok torlódási pontjait határozza meg.

Általában problémát jelent a megállás kérdése, azaz hogy mikor találtuk már meg az összes optimumot. A megállási szabályt tekintve egy 1987-es eredmény alapján egy nem túl sok feltevést igénylő eredmény született:

Ha N lokális keresést végeztünk, és megtaláltunk w optimum pontot, akkor az optimum pontok becsült $\hat{w}$ száma:

$$\hat{w} = \frac{w(N-1)}{N-w-2}$$

Az alapötlet az, hogy megállunk, amikor w már közeli $\hat{w}$-hez.

Algorithm 11 Multistart$(X, f, \mathrm{LS}, N$

$f^U = \infty$
for $(k = 1, \ldots, N)$ **do**
  Generálunk x pontot X-ben egyenletesen
  $x_k = \mathrm{LS}(x)$
  if $(f(x_k) < f^U)$ **then**
    $f^U = f(x_k), \quad x^U = x_k$

10.2.1. Klaszterezés a lokális keresések számának csökkentésére

Az alapötlet az, hogy értelmetlen számítási teljesítményt fektetni olyan lokális optimalizálásba, amelyben a kiindulási pont valamely már megtalált lokális optimum vonzási tartományában van. Kis függvényértékeket adó pontok medencékbe koncentrálódnak, amik érintkezhetnek lokális minimalizáló eljárások vonzási tartományával. Sima függvények esetén az ilyen tartományok elliptikus karakterisztikával bírnak, amit a Hesse-mátrix ír le az optimum pontokban. Sok klaszterező algoritmus változatot terveztek és nagy előrelépések történtek a 70-es és 80-as években a klaszterek meghatározásához felhasznált információt illetően. Numerikus eredményeket analitikusak váltotak fel.

Most az u.n. MLSL (Multi-Level Single Linkage) algoritmust mutatjuk be. Ez nem közvetlenül formák klaszterekt, de az ötlet, hogy nem indítunk lokális keresést olyan pontból, melyet impliciten már egy másik megtalált optimumhoz rendeltünk. A megtalált optimumokat a Λ halmazban tároljuk. A tűrési távolságot az aktuális k iterációban a

$$r_k = \sqrt{\pi}\left(\sigma V(X)\Gamma(1+\frac{n}{2})\frac{\log k}{k}\right)^{\frac{1}{n}}$$

10.2.2. Alagutazás és feltöltött függvények

Míg a klaszterezés ötletét a 70-es években kutatták, egy másik ötlet is megjelent a 80-as években: Nem érdekel minket az összes lokális optimum, csak le akarunk lépdelni a lokális optimumokon keresztül a globálisig. Számos gyakran hivatkozott cikk jelent meg a témában és további kutatásokat eredményezett. Az egyik ilyen ötlet:

Tegyük fel, hogy egy véges sok minimum ponttal rendelkező sima függvény optimumát keressük. Miután megtaláltunk egy lokális minimumot, a függvényt áttranszformáljuk, hogy egy új kezdőpontot találjunk egy jobb lokális minimumhoz tartozó vonzási tartományban. Ezt megtehetjük egy u.n. alagutazással, vagy feltöltéssel.

Algorithm 12 Multi-level Single Linkage$(X, f, N, \mathrm{LS}, \gamma, \sigma)$

Válasszunk egyenletesen N pontot X-en és értékeljünk ki őket. $\Lambda = \emptyset$
Válasszuk ki a $k = \gamma N$ legalacsonyabb pontot.
for $(i = 1, \ldots, k)$ **do**
  if $(\nexists j < i, (f(_j) < f(x_i)$ÉS $\|x_j - x_i\| < r_k)$ **then**
    Lokális keresés $\mathrm{LS}(x_i)$
    if $(\mathrm{LS}(x_i) \notin \Lambda)$ **then**
      Tároljuk $\mathrm{LS}(x_i)$-t Λ-ban.
while $(|\Lambda| - \hat{w} < 0.5)$ **do**
  $k = k + 1$, veszünk egy x_k mintapontot X-ből.
  if $(\nexists j < k, (f(x_j) < f(x_k)$ ÉS $\|x_j - x_k\| < r_k))$ **then**
    Lokális keresés $\mathrm{LS}(x_k)$
    if $(\mathrm{LS}(x_k) \notin \Lambda)$ **then**
      Tároljuk $\mathrm{LS}(x_k)$-t.

Az alagutazás: Miután megtaláltunk egy x_1^* lokális minimumot egy x_0 kezdőpontból, az algoritmus iteratívan keresi a

$$T_k(x) = \frac{f(x) - f(x_k^*)}{(x - x_k^*)^\alpha} = 0$$

egyenlet megoldását, $\alpha > 0$. Az $x_k \neq x_k^*$ megoldásban a függvényérték azonos $f(x_k) = f(x_k^*)$, így ezt kezdőpontnak használhatjuk egy lokális keresésben hogy elérjük x_{k+1}^*-ot amiben $f(x_{k+1}^*) < f(x_k^*)$, amire aztán ugyanezt végrehajtjuk. Az ötlet csábító, a kérdés csak az alagutazás végrehajtásának hatékonysága.

Egy másik eljárás az úgynevezett feltöltés. A cél ugyanaz mint az alagutazásnál. El kívánjuk érni egy eddig találtaknál jobb minimum vonzási tartományát. Az ötlet azonban itt az, hogy megszűntetjük a stacionárius pontokat a már megtalált optimumok vonzási tartományában. Ehhez az x_k^*-hoz tartozó vonzási tartományt „feltöltjük”. Például az alábbi

$$ff(x) := -\frac{1}{r + f(x)} \exp\left(\frac{\|x - x_k^*\|^2}{\rho}\right)$$

feltöltött függvényt.

Algorithm 13 Feltöltött függvény multistart

$k = 1, \quad x_1^* = \mathrm{LS}\, f(x_0)$
repeat
  Adaptáljuk a ρ és r paramétereket.
  Válasszuk meg ξ-t
  $x_k = \mathrm{LS}\, ff(x_k^* + \xi)$
  $x_{k+1} = \mathrm{LS}\, f(x_k)$
  $k = k + 1$
until $(f(x_k^*) \geq f(x_{k-1}^*))$

10.3. P-algoritmus

Legyen $f^U = \min_i y_i$ a legjobb eddig talált függvényérték, és δ_k egyféle pozitív aspirációs szint a javulásra a k iterációban. A P-algoritmus a következő pontnak azt választja, amelynél az $f^U - \delta_k$ javulás valószínűsége maximális:

$$x_{k+1} = \arg\max_x P(\xi_k(x) < f^U - \delta_k)$$

A fenti megoldása nagyban függ a sztochasztikus modell konstrukciójától. Például ha ξ_k Gauss-eloszlású, akkor a modellt leírja az $m_k(x)$ és $s_k^2(x)$ várható érték és szórás, ekkor a fentivel ekvivalens:

$$x_{k+1} = \arg\max_x \frac{f^U - \delta_k - m_k(x)}{s_k(x)}$$

Vegyük észre hogy $s_k(p_i) = 0$.

10.4. Radiális alapfüggvény

Az alapötlet, hogy interpoláljuk f-et az x pontban az y_i és p_i értékek ismeretében úgy, hogy a közeli pontok jobban számítanak. Ezt az u.n. radiális alapfüggvénnyel érjük el. Pl. $\Theta(r) = \exp(-r^2)$. Legyen most

$$\varphi_k(x) = \sum_i w_i \Theta(\|x - p_i\|)$$

ahol a w_k súlyok meghatározhatók a $\varphi_k(p_j) = y_j$ egyenletekből ($w = \Theta(p)^{-1} y^T$ egyenletrendszer)

10.5. Vezérelt véletlen keresés

Algorithm 14 Vezérelt véletlen keresés

$k = N$
Generáljunk és értékeljünk ki egy P halmazt N egyenletesen választott X-beli ponttal
$y_k = f(\text{pmax}_k) = \max_{p \in P} f(p)$
while $(y_k - \min_{p \in P} f(p) > \alpha)$ **do**
  $k = k + 1$
  Válasszunk egy $\{p_1, \ldots, p_{n+1}\}$ véletlen részhalmazt P-ből
  $x_k := \frac{2}{n} \sum_{i=1}^{n} p_i - p_{n+1}$
  if $(x_k \in X$ ÉS $f(x_k) < y_{k-1})$ **then**
    cseréljük ki pmax_k-t x_k-ra P-ben.
  $y_k = f(\text{pmax}_k) = \max_{p \in P} f(p)$

10.6. UPCR - Málna-algoritmus

Az eljárást elsősorban azért fejlesztették, hogy képesek legyenek egy $S(f^* + \delta)$ szintvonalat lefedni, ami egy konfidenciatartományt reprezentálna nemlineáris paraméterbecslésnél. Az ötlet az, hogy az $S(f^* + \delta)$ fedését keressük P-beli mintapontokkal mintha azok egyenletes eloszlásból lennének, vagy egy u.n. Málna-halmazból $R = \{x \in X | \exists p \in P, \|x - p\| \leq r\}$ ahol r egy kicsi sugár.

Algorithm 15 UCPR$(f, X, N, c, f^* + \delta)$

$k = N$
Generálunk és kiértékelünk N véletlenül egyenletesen választott X-beli pontot (P).
$y_k = f(\text{pmax}_k) = \max_{p \in P} f(p)$
while $(y_k > f^* + \delta)$ **do**
  $k = k + 1$
  Határozzuk meg az átlagos pontközi távolságot (r_k) P-ben.
  Málna halmaz: $R_k = \{x \in X | \exists p \in P, \|x - p_i\| \leq c \cdot r_k\}$
  Generáljuk x_k-t egyenletesen R_k-ban.
  if $(x_k \in X$ ÉS $f(x_k) < y_{k-1})$ **then**
    Cseréljük pmax_k-t $x_k \in P$-re.
  $y_k = f(\text{pmax}_k) = \max_{p \in P} f(p)$

10.7. Szimulált lágyítás

A fenti két algoritmusnak bőséges irodalma van. Jóval nehezebben elemezhetőek, de alkalmazásokban nagyon népszerű módszerek az u.n. meta-heurisztikák. Ide tartozik a Szimulált lágyítás, evolúciós algoritmus, genetikai algoritmus, tabu keresés.
Fontos kérdés, hogy vajon tényleg jobbak tudnak-e lenni ezek a módszerek, mint a klasszikus módszerek valamilyen ötvözete? Most bemutatjuk az u.n. szimulált lágyítást. Ez egy olyan mintavételezési folyamatot ír le a döntési térben, ahol az új mintapontokat a jelenlegi iteráció egy u.n. szomszédságából választjuk. Az új mintapontokat mindig elfogadjunk, ha jobbak, és p valószínűséggel ha rosszabbak. A valószínűség az u.n. hőmérséklettől függ, ami csökken az iterációk során.

Az algoritmus tartalmazza a CR (cooling rate) paramétert, ami a hőmérséklet csökkenését szabályozza. A kezdőhomérsékletet itt fix 1000-nek választjuk. Az algoritmus egy rossz pontot attól függően fogad el, hogy az mennyire rossz, illetve mennyire előrehaladott az algoritmus. Ez egy általános koncepció a szimulált lágyításban. Számos út kínálkozik a szomszédságból való mintavételezésre. Egy dimenzióban intuitívan az $[x_k - \varepsilon, x_k + \varepsilon]$ választás a kézenfekvő. Mivel ezek a heurisztikák kezdetben nem folytonos, hanem egészértékű problémákra irányultak, az egyik első megközelítés szerint a folytonos változókat bitlánccként kódolhatjuk. Szemléltetésül: Az $[3, 7]$ minden pontját egy $(B_1, \ldots, B_9) \in \{0, 1\}^9$ bitlánc reprezentálja, ahol

$$x = 3 + 4\frac{\sum_{i=1}^{9} B_i 2^{i-1}}{511}$$

Ez egy rácsot ír le, ahol minden bitláncnak egy rácspont felel meg, így a rasztertávolság $\frac{4}{511}$. A szomszédságból való mintavételezésen az egyik B_i átbillentését értjük. Az így kapott új pontkat nem feltétlenül tekintenénk a folytonos értelemben szomszédságnak.

Algorithm 16 SA$(X, f, \mathrm{CR}, N$

$f^U = \infty, \quad T_1 = 1000$
Legyen x_1 egyenletes eloszlás szerint generált pont X-ből.
for $(k = 1, \ldots, N)$ **do**
  Generáljuk x-et x_k egy környezetéből
  if $(f(x) < f(x_k))$ **then**
    $x_{k+1} = x$
    if $(f(x) < f^U)$ **then**
      $f^U = f(x)$ és $x^U = x$
  else
    $e^{\frac{f(x_k)-f(x)}{T_k}}$ valószínűséggel $x_{k+1} = x$
  $T_{k+1} = \mathrm{CR} \cdot T_k$