http://www.ui.ac.ir

ریاضی و جامعه
شاپا (چاپی): ۶۴۹۳-۲۳۴۵، شاپا(الکترونیکی): ۶۵۰۷-۲۳۴۵
جلد ۱، شماره ۱ (۱۳۹۵)، صص. ۲۵-۱۳

http://math-sci.iranjournals.ir

آشنایی با روش فیلتر برای حل مسائل برنامه ریزی غیرخطی

نجمه حسینی منجزی

چکیده. یکی از روش های حل مسائل برنامه ریزی غیر خطی که سال ها مورد استفاده قرار گرفته است روش جریمه می باشد. در این مقاله می خواهیم با معرفی مفهوم جدید فیلتر، الگوریتمی برای حل مسائل برنامه ریزی غیر خطی مقید بیان کنیم، که در آن از تابع جریمه استفاده نشود. اگر الگوریتم از فیلتر به جای تابع جریمه استفاده کند، برخی از مشکلات روش جریمه را حل می کند و همچنین همگرایی سرتاسری را نتیجه می دهد.

۱. مقدمه

مسئله برنامه ریزی غیر خطی را به صورت زیر در نظر می گیریم:

$$(۱) \qquad \begin{aligned} min \quad & f(x) \\ & c_i(x) \leq ۰ \qquad i = ۱, ۲, \ldots, l \\ & c_i(x) = ۰ \qquad i = l+۱, \ldots, m \\ & x \in \mathbb{R}^n, \end{aligned}$$

که توابع $f(x) : \mathbb{R}^n \longrightarrow \mathbb{R}$ و $c_i(x) : \mathbb{R}^n \longrightarrow \mathbb{R}$ برای $i \in \{۱, ۲, \ldots, m\}$ توابعی $C^۲$ می باشند و تابع هدف $f(x)$ از پایین کراندار است. یک رویکرد برای حل مسئله مقید جایگزین کردن آن، با زیرمسئله های نامقید می باشد. یعنی به جای حل مسئله (۱.۲) تلاش می شود مسئله

$$(۲) \qquad \begin{aligned} min \quad & F(x) \\ & x \in \mathbb{R}^n, \end{aligned}$$

حل شود.

که در آن تابع $F(x)$ را به روش های مختلف می توان از تابع هدف $f(x)$ و توابع قیودی $c_i(x)$ به دست آورد به طوری که دو ویژگی زیر را داشته باشد:

تاریخ دریافت: ۱۳۹۲/۰۴/۰۳ تاریخ پذیرش: ۱۳۹۲/۰۶/۲۰ .

- تابع $F(x)$ شامل یک جمله جریمه [۱] است. زمانی که قیدی از قیود مسئله نقض شده باشد این جریمه باعث افزایش $F(x)$ می شود و مقدار جریمه به مقدار نقض شدن قید بستگی دارد.
- مینیمم کننده x_F^* تابع $F(x)$ نزدیک ناحیه شدنی و نیز نزدیک به جواب مسئله مقید (۱.۲) باشد.

روش جایگزینی مسئله، با یک مسئله نامقید همراه با یک جمله جریمه را، روش جریمه [۲] می نامند.

روش تابع جریمه دارای مشکلاتی در اجرا می باشد (که در ادامه بیان می کنیم.). انگیزه ما از این مقاله ارائه روشی است که در آن از تابع جریمه استفاده نشود، مشروط بر این که روش جدید سختی های معمول روش جریمه را، نداشته باشد. روش مورد نظر ما در اینجا روش فیلتر [۳] است که در ادامه به جزئیات آن می پردازیم.

روش فیلتر برای اولین بار توسط دونالدسون [۴] در رساله ی دکترای وی در سال ۱۹۹۵ ارائه شد. و پس از آن، این روش برای مسائل بهینه سازی همراه با قیود تساوی و نامساوی [۲، ۳] و دیگر مسائل مورد استفاده قرار گرفت. در سال ۱۹۹۵ اثباتی برای همگرایی آن ارائه نشد و تنها توسط برنامه های عددی مورد بررسی قرار گرفت که نتایج عددی امیدوارکننده بود. در سال ۲۰۰۰ اثباتی برای همگرایی سرتاسری مسئله مقید کلی توسط فلچر [۵] با فرضیات معمولی ارائه شد.

این روش توسط افراد مختلف و برای حل مسائل گوناگون مورد استفاده قرار گرفت و از نظر اثبات همگرایی، الگوریتم و حل عددی گسترش یافته است.

در اینجا قصد نداریم جزئیات را بیان کنیم و یا وارد اثبات شویم، تنها می خواهیم ایده ها را بیان کنیم. این ایده ها مختص مسئله و روش خاصی نمی باشد و می توان برای حل مسائل مختلف از این ایده ها استفاده نمود.

این مقاله به چهار بخش تقسیم شده است. در بخش بعد ابتدا مفهوم تابع جریمه را دقیق تر بیان می کنیم و پس از آن برخی از مشکلات این روش را توضیح می دهیم. در بخش سوم به معرفی الگوریتم فیلتر می پردازیم و در نهایت ایده هایی را برای بدست آوردن نتایج عددی بهتر بیان می کنیم.

۲. روش تابع جریمه

در حل با روش جریمه، مسئله مقید با زیر مسئله های نامقید جایگزین می شود و به جای حل مسئله مقید، در هر مرحله الگوریتم یک زیر مسئله نامقید را حل می کند. بحث خود را با ساختن یک تابع جریمه برای مسئله (۱.۲) شروع می کنیم.

برای قید های $c_i(x) \leq 0$ که $i = 1, 2, \ldots, l$ تابع $c_i^+(x)$ را به صورت زیر تعریف می کنیم:

$$c_i^+(x) = \begin{cases} 0 & c_i(x) \leq 0 \\ c_i(x) & c_i(x) > 0 \end{cases}.$$

و برای قید های $c_i(x) = 0$ که $i = l+1, \ldots, m$ تابع $c_i^+(x)$ را به صورت زیر تعریف می کنیم:

$$c_i^+(x) = max(0, |c_i(x)|).$$

که این تابع به ازای هر x که در آن قید برقرار باشد، صفر است و برای هر x که در قید برقرار نباشد مثبت است. بعلاوه هر چه اندازه برقرار نبودن قید ها بزرگتر باشد $c_i^+(x)$ نیز بزرگتر می باشد. بنابراین $c_i^+(x)$ نوعی تابع جریمه است که می توان برای $c_i(x)$ در نظر گرفت.

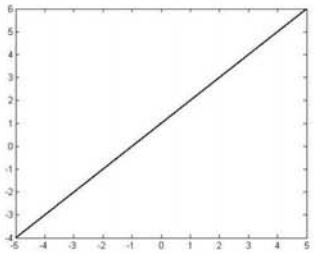
قبل از آن که تابع جریمه را تعریف کنیم برای چند قید ساده $c_i^+(x)$ را بررسی می کنیم.

[1] penalty [2] penalty method [3] filter [4] Donaldson [5] Fletcher

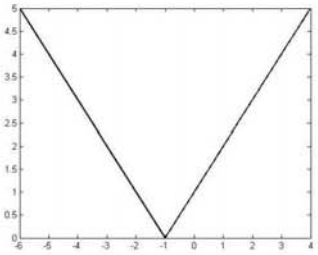
مثال ۱.۲. *در نظر بگیرید* $c(x) = x + ۱ = ۰$ *باشد هنگامی که* $x \in \mathbb{R}$ *در این صورت:*

$$c^+(x) = \begin{cases} ۰ & x = -۱ \\ |x+۱| & x \neq -۱ \end{cases},$$

که مطابق شکل های ۱ و ۲ به ترتیب تابع $c(x)$ *کاملاً هموار و تابع* $c^+(x)$ *مشتق پذیر نمی باشد.*



شکل ۱: تابع کاملاً هموار $c(x) = x + ۱$.

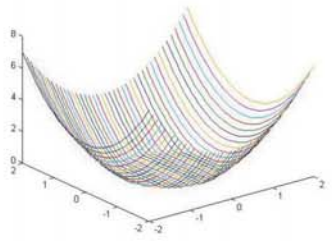


شکل ۲: تابع مشتق ناپذیر $c^+(x)$.

مثال ۲.۲. *فرض کنید* $c(x) = x^۳ \leq ۰$ *وقتی* $x \in \mathbb{R}$ *باشد . در این صورت:*

$$c^+(x) = \begin{cases} ۰ & x \leq ۰ \\ x^۳ & x > ۰. \end{cases}$$

که مطابق شکل ۳ و ۴ به ترتیب $c(x)$ *و* $c^+(x)$ *هر دو مشتق پذیر می باشند.*



شکل ۶: تابع مشتق ناپذیر $c^+(x)$.

مثال های (۱.۲) و (۳.۲) نشان می دهند که، برای مسائل ساده حتی اگر $c(x)$ کاملاً هموار باشد ممکن است $c^+(x)$ هموارى لازم را نداشته باشد.

با توجه به جریمه $c^+(x)$ ،یک تعریف معقول برای تابع جریمه به صورت زیر است:

$$Q(x,\mu) = f(x) + \frac{1}{\mu}\sum_{i=1}^{m} c_i^+(x), \tag{۳}$$

که در آن μ عددی مثبت است و پارامتر جریمه نامیده می شود و این تابع را، تابع جریمه قدر مطلق می نامیم. نقش پارامتر جریمه واضح است. هنگامی که $\mu \longrightarrow 0$ جریمه مربوطه افزایش می یابد.

طبق مثال های فوق، حتی اگر تابع هدف $f(x)$ و توابع $c_i(x)$ ها کاملاً هموار باشند تابع تعریف شده (۳) لزوماً دارای مشتق های جزئی مرتبه اول پیوسته نمی باشد.

اکنون می توان تعریف تابع $Q(x,\mu)$ را چنان تغییر داد که دارای مشتق های جزئی مرتبه اول پیوسته باشد:

$$Q(x,\mu) = f(x) + \frac{1}{\mu}\sum_{i=1}^{m} (c_i^+(x))^2. \tag{۴}$$

قضیه ۴.۲. *اگر تابع* $g(x)$ *مشتق های جزئی مرتبه اول پیوسته بر* $\mathbb{R}^n$ *داشته باشد. آن گاه تابع* $h(x) = (g^+(x))^2$ *نیز چنین است.*

پس اگر $Q(x,\mu)$ را به صورت (۴) تعریف کنیم، آن گاه مشتق پذیر می شود. که این تابع را، تابع جریمه درجه ۲ می نامند.

پس در هر مرحله از الگوریتم روش جریمه، یک مسئله به فرم:

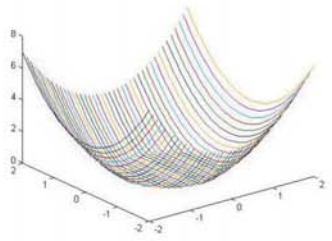
$$min \qquad Q(x,\mu_k), \tag{۵}$$

حل می شود. که در آن

$$Q(x,\mu) = f(x) + \frac{1}{\mu}\sum_{i=1}^{m} (c_i^+(x))^2,$$

هموار می باشد. از طرفی حل مسئله (۵) به صورت دقیق پر هزینه است، و معمولاً به صورت تقریبی حل می شود و برای حل آن می توان از روش های موجود برای مسائل برنامه ریزی نامقید هموار استفاده کرد.

در روش جریمه انتخاب درست پارامتر μ مهم و تعیین کننده می باشد. به مثال زیر توجه کنید.



شکل ۶: تابع مشتق ناپذیر $c^+(x)$.

مثال های (۱.۲) و (۳.۲) نشان می دهند که، برای مسائل ساده حتی اگر $c(x)$ کاملاً هموار باشد ممکن است $c^+(x)$ هموارى لازم را نداشته باشد.

با توجه به جریمه $c^+(x)$ ،یک تعریف معقول برای تابع جریمه به صورت زیر است:

$$Q(x,\mu) = f(x) + \frac{1}{\mu}\sum_{i=1}^{m} c_i^+(x), \tag{۳}$$

که در آن μ عددی مثبت است و پارامتر جریمه نامیده می شود و این تابع را، تابع جریمه قدر مطلق می نامیم. نقش پارامتر جریمه واضح است. هنگامی که $\mu \longrightarrow 0$ جریمه مربوطه افزایش می یابد.

طبق مثال های فوق، حتی اگر تابع هدف $f(x)$ و توابع $c_i(x)$ ها کاملاً هموار باشند تابع تعریف شده (۳) لزوماً دارای مشتق های جزئی مرتبه اول پیوسته نمی باشد.

اکنون می توان تعریف تابع $Q(x,\mu)$ را چنان تغییر داد که دارای مشتق های جزئی مرتبه اول پیوسته باشد:

$$Q(x,\mu) = f(x) + \frac{1}{\mu}\sum_{i=1}^{m} (c_i^+(x))^2. \tag{۴}$$

قضیه ۴.۲. *اگر تابع* $g(x)$ *مشتق های جزئی مرتبه اول پیوسته بر* $\mathbb{R}^n$ *داشته باشد. آن گاه تابع* $h(x) = (g^+(x))^2$ *نیز چنین است.*

پس اگر $Q(x,\mu)$ را به صورت (۴) تعریف کنیم، آن گاه مشتق پذیر می شود. که این تابع را، تابع جریمه درجه ۲ می نامند.

پس در هر مرحله از الگوریتم روش جریمه، یک مسئله به فرم:

$$min \qquad Q(x,\mu_k), \tag{۵}$$

حل می شود. که در آن

$$Q(x,\mu) = f(x) + \frac{1}{\mu}\sum_{i=1}^{m} (c_i^+(x))^2,$$

هموار می باشد. از طرفی حل مسئله (۵) به صورت دقیق پر هزینه است، و معمولاً به صورت تقریبی حل می شود و برای حل آن می توان از روش های موجود برای مسائل برنامه ریزی نامقید هموار استفاده کرد.

در روش جریمه انتخاب درست پارامتر μ مهم و تعیین کننده می باشد. به مثال زیر توجه کنید.

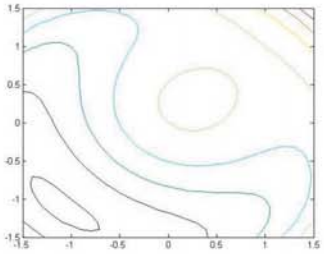
مثال ۵.۲. *مسئله مقید زیر را در نظر بگیرید:*

$$\begin{aligned} min \quad & x_۱ + x_۲ \\ & x_۱^۲ + x_۲^۲ - ۲ = ۰ \\ & x_۱, x_۲ \in \mathbb{R}, \end{aligned}$$

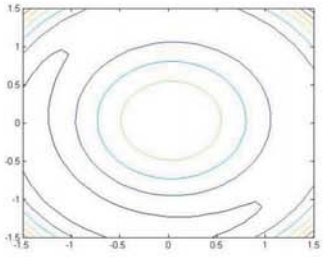
که جواب دقیق آن $x^* = (-۱, -۱)^t$ *می باشد. تابع جریمه مسئله به صورت زیر تعریف می شود:*

$$Q(x, \mu) = x_۱ + x_۲ + \frac{۱}{\mu}(x_۱^۲ + x_۲^۲ - ۲)^۲.$$

اگر $\mu = ۲$ *آنگاه مجموعه ترازهای* $Q(x, \mu)$ *در شکل ۷ و اگر* $\mu = ۰/۲$ *مجموعه ترازهای آن در شکل ۸ نشان داده شده است.*



شکل ۷: مجموعه ترازهای تابع $Q(x, \mu)$ *با* $\mu = ۲$.



شکل ۸: مجموعه ترازهای تابع $Q(x, \mu^*)$ *با* $\mu = ۰/۲$.

پس انتخاب پارامتر جریمه بسیار مهم و تعیین کننده است.
حال می خواهیم الگوریتم روش تابع جریمه حالت کلی را بیان کنیم.

الگوریتم ۶.۲. *الگوریتم روش تابع جریمه*

مرحله ۰ : *نقطه شروع اولیه* x_0^s *و* $\mu_0 > 0$ *را انتخاب نمائید و قرار دهید* $k = 0$.
مرحله ۱ : *مینیمم تقریبی مسئله* $Q(x, \mu_k)$ *را با شروع از* x_k^s *حساب نمائید و فرض کنید جواب* x_k *بدست آمده باشد.*
مرحله ۲ : *اگر همگرایی برقرار باشد توقف کنید و* x_k *جواب تقریبی است.*
در غیر اینصورت پارامتر جریمه μ_{k+1} *را انتخاب نمائید.* $x_{k+1}^s = x_k$ *را نقطه شروع اولیه و* $k = k + 1$ *قرار دهید و به مرحله ۲ بروید.*

این الگوریتم حالت کلی است و روش خاصی برای حل $Q(x, \mu)$ معرفی نکردیم. البته می توان از هر روش حل مسائل نامقید هموار استفاده کرد. و روشی برای انتخاب پارامتر جریمه بیان نکردیم.

یکی از مشکلات روش جریمه انتخاب مناسب پارامتر جریمه می باشد همان گونه که در مثال (۵.۲) نشان داده شد پارامتر μ در مجموعه ی ترازها و بنابراین در جواب مسئله تعیین کننده است از طرفی روش کلی برای انتخاب پارامتر جریمه وجود ندارد. معمولاً یک کران بالا برای μ وجود دارد که به ازای ان تابع $Q(x, \mu)$ مینیمم ندارد ولی این کران را نمی توان بدست آورد. انتخاب خیلی بزرگ برای μ می تواند باعث تولید نقطه ناموجهی برای (۱.۲) یا باعث افزایش غیر کراندار در تابع هدف شود. از طرف دیگر انتخاب خیلی کوچک برای μ باعث کاهش اثر تابع هدف در تابع جریمه و کندی همگرایی می شود.

مشکل دیگر تابع جریمه مشتق ناپذیری آن است. که در بالا راه حلی برای رفع این مشکل ارائه شد اما اگر به مشتقات مراتب بالا تر نیاز داشته باشیم این راه حل پاسخگو نمی باشد.

۳. روش فیلتر

می توان مسئله (۱.۲) را به این صورت در نظر گرفت که همزمان به دنبال دو هدف هستیم. اولین هدف، مینیمم نمودن تابع هدف $f(x)$ و دومین هدف برقرار بودن قیدهای مسئله می باشد. دو هدف را می توان بصورت زیر نوشت:

$$min \quad f(x), \tag{۶}$$

و

$$min \quad h(c(x)), \tag{۷}$$

به طوری که تابع $h(c(x))$ به صورت زیر تعریف می شود:

$$h(c(x)) = \|c^+(x)\|_1 = \sum_{i=1}^{m} c_i^+(x).$$

که در اینجا از نرم $\|.\|_1$ استفاده کرده ایم اما می توان نرم های دیگر را نیز بکار برد.

در روش تابع جریمه توابع (۶) و (۷) ترکیب می شوند و یک مسئله مینیمم سازی نامقید می سازند. اما ما ترجیح می دهیم که (۶) و (۷) را به عنوان اهداف جدا از هم ببینیم و از روش های بهینه سازی چند هدفه استفاده کنیم. البته موقعیت ما از بهینه سازی چند هدفه متفاوت است. زیرا ضروری است ابتدا $h(c(x))$ را مینیمم کنیم. یعنی مینیمم سازی $h(c(x))$ تقدم دارد.

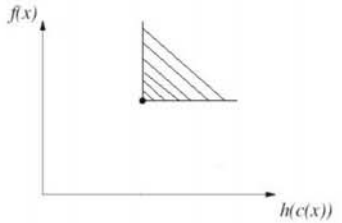
شایسته است از مفهوم غلبه بهینه سازی چند هدفه استفاده کنیم. فرض کنیم (f^k, h^k) نشان دهنده ی $(f(x), h(x))$ در نقطه ی x_k باشد.

تعریف ۱.۳. *یک جفت (f^k, h^k) برجفت دیگر (f^l, h^l) غالب است اگر وتنها اگر $f^k \leq f^l$ و $h^k \leq h^l$ باشد.*

در این صورت می توان گفت (f^l, h^l) مغلوب (f^k, h^k) می باشد.
در برنامه ریزی غیر خطی تعریف بالا به این معنی است که x_k حداقل بخوبی x_l می باشد. با توجه به این مفهوم، حال تعریف فیلتر را بیان می کنیم.

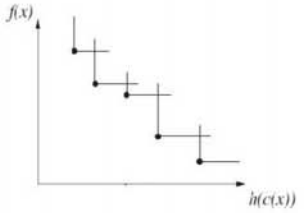
تعریف ۲.۳. *یک فیلتر یک لیست از جفت های (f^l, h^l) است که در آن هیچ جفتی بر جفت دیگر غالب نمی باشد.*

در فیلتر نقطه (f^k, h^k) قابل قبول است. اگر (f^k, h^k) مغلوب هیچ جفت دیگری در فیلتر نباشد.
پس هر عضوی که در فیلتر وارد می شود یک بلاک از نقاط غیر قابل قبول تولید می کند. مثلاً در شکل ۹ قسمت هاشور خورده نقاط غیر قابل قبول در فیلتر را نشان می دهد که توسط یک نقطه تولید شده است.



شکل ۹: مجموعه نقاط غیرقابل قبول تولید شده توسط یک نقطه.

چون هر نقطه یک بلاک از نقاط غیر قابل قبول تولید می کند پس در هر فیلتر یک مجموعه از نقاط غیر قابل قبول داریم.
فیلتر را می توان به صورت گرافیکی در صفحه (f, h) نشان داد. که در شکل ۱۰ نشان داده شده است.



شکل ۱۰: نمایش گرافیکی یک فیلتر.

ایده ی کلیدی حل با استفاده از روش فیلتر به این صورت است که به جای استفاده از کاهش در تابع جریمه این شرط را قرار می دهیم که وقتی در مرحله k ام نقطه ی جدید x_{k+1} را بدست می اوریم. الزاماً این نقطه باید در فیلتر قابل قبول باشد.
حال نقطه ی x_{k+1} توسط فیلتر قابل قبول است اگر نقطه متناظر (f^{k+1}, h^{k+1}) به وسیله هیچ نقطه ای در فیلتر مغلوب

نباشد. یعنی:
یک نقطه متناظر با l در فیلتر وجود داشته باشد که

$$f^{k+1} \leq f^l \qquad یا \qquad h^{k+1} \leq h^l. \tag{۸}$$

در غیر این صورت این مرحله رد می شود. بنابراین شرط معمول نزول در تابع جریمه را با لزوم پذیرفتن نقطه جدید در فیلتر جایگزین می‌کنیم.

حال الگوریتم حالت کلی روش فیلتر را بیان می کنیم.

الگوریتم ۳.۳. *الگوریتم روش فیلتر*

مرحله ۰ : *نقطه شروع اولیه x_0 را انتخاب نمایید و قرار دهید $k = 0$.*
مرحله ۱ : *تقریب درجه ۲ مسئله (۱.۲) را با نقطه x_k بدست بیاورید.*
مرحله ۲ : *قرار دهید $x_0 = x_k$ و مسئله درجه ۲ را حل کنید و نقطه آزمایشی x_{k+1} را بدست آورید.*
مرحله ۳ : *اگر x_{k+1} در فیلتر قابل قبول باشد. نقطه ی متناظر (f^{k+1}, h^{k+1}) را به فیلتر اضافه کنید. و به مرحله ۴ بروید.*
در غیر این صورت تقریب درجه ۲ را بازنویسی نمائید و به مرحله ۲ بروید.
مرحله ۴ : *اگر همگرایی برقرار شد متوقف شوید و x_{k+1} را به عنوان جواب تقریبی بپذیرید.*
در غیر این صورت قرار دهید $k = k + 1$ و به مرحله ۱ بروید.

در مرحله ۲ بیان کردیم تقریب درجه ۲ را حل کنید که در اینجا روش خاصی را برای حل آن مشخص نمی کنیم و مثلاً می توان از روش ناحیه قابل اعتماد [۶] و یا روش جستجو خطی [۷] استفاده کرد. و در مرحله ۳ بیان کردیم تقریب درجه ۲ را باز نویسی کن. مثلاً در روش ناحیه قابل اعتماد با کاهش شعاع ناحیه اعتماد و در روش جستجو خطی با کاهش طول گام، این بازنویسی را انجام می دهیم برای جزئیات بیشتر به [۴] مراجعه کنید.

لازم است اشاره کنیم که این الگوریتم بسیار آسان است. و هیچ سختی برای انتخاب کردن پارامترهای مناسب ندارد. در این روش افزایش در h^k اجازه داده شده و یک غیر یکنواختی نسبت به روش جریمه داریم که یک مزیت برای حل مسئله می باشد.

مقاله [۱] این روش را از نقطه نظر عددی بررسی می کند و این روش را برای تعداد زیادی مسئله بکار می‌برد و نتایج عددی بسیار امیدوار کننده است.

در مقاله [۳] این روش برای مسائل برنامه ریزی مقید با قیود تساوی بیان شده است و همگرایی سراسری آن تحت فرضیات مناسب نشان داده شده است.

در مقاله [۲] این روش برای مسائل حالت کلی (۱.۲) بیان شده است و همگرایی سراسری آن تحت فرضیات معقول اثبات شده است که بر توانایی این روش تأکید می کند.

مثال زیر مسئله ۷ از مجموعه مسائل معرفی شده در [۵] می‌باشد. کد الگوریتم معرفی شده در [۳] را با برنامه $MatLab$ ۲۰۱۲ نوشته‌ایم و با یک نوت بوک شخصی با مشخصات

CPU : Intel Core I5-3210 M , 2.5 GHz
Memory : 4 GB ,

اجرا کردیم. نتیجه مرحله آخر را در ادامه بیان می‌کنیم.

[6] trust region [7] line search

مثال ۴.۳. *مسئله بهینه سازی زیر را در نظر بگیرید:*

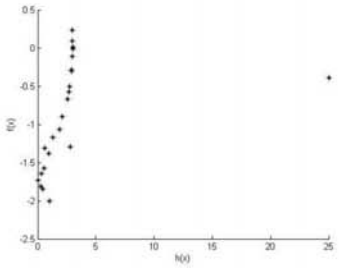
$$\begin{aligned} \min \quad & \log(1+x_1^2)-x_2 \\ & (1+x_1^2)^2+x_2^2-4=0, \end{aligned}$$

نتیجه حاصل از آخرین مرحله اجرای الگوریتم به صورت زیر بدست آمد:

```
*******************************************************
 This point is K.K.T point.
*******************************************************
 X =
   -0.0000
    1.7321

 coun= 26                    countlim=   200
 f(x)= -1.7321e+00           h(x)=  2.7360e-10
```

که این نتایج نشان می‌دهد الگوریتم در ۲۶ تکرار به پایان رسید و به نقطه‌ی KKT رسیده‌ایم.
در شکل زیر نتایج حاصل از اجرای الگوریتم به صورت گرافیکی نمایش داده شده است که هر $*$ نشان دهنده‌ی یک نقطه $(h(x), f(x))$ می‌باشد که نقطه x متناظر با آن یک نقطه در فیلتر می‌باشد.
الگوریتم چنان تنظیم شده است که نقطه‌ای که در آن $h(x)=0$ همان نقطه مورد نظر می‌باشد.



شکل ۱۱ : نتیجه حاصل از مثال

۴. ایده‌هایی برای افزایش سرعت همگرایی

در این بخش ایده هایی را برای افزایش سرعت همگرایی ارائه می کنیم. اثبات هایی که در [۲، ۳] ارائه شده است از ایده های معرفی شده در این بخش استفاده نمی کنند یعنی از نظر آنالیزی و اثبات همگرایی سراسری به این موارد در الگوریتم نیاز نداریم اما از نظر عددی این موارد بسیار مهم می باشند و باعث افزایش سرعت همگرایی می شوند و از پدیده هایی از جمله دور [۸] جلوگیری می کنند.

[8] cycle

۱.۴. کران بالا برای قیود. می توان یک شرط دیگر برای قابل قبول شدن نقاط در فیلتر اضافه نمائیم. هر نقطه در فیلتر قابل قبول است اگر علاوه بر (۸) داشته باشیم

$$h(c(x)) \leq u,$$

که u یک کران بالا برای قیود است. این شرط به آسانی قابل پیاده سازی در الگوریتم می باشد به این صورت که ورودی های فیلتر باید در $(-\infty, u]$ قرار داشته باشند. یک دلیل برای اضافه کردن این شرط این است که از قابل قبول شدن دنباله‌ای به صورت:

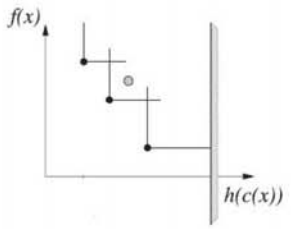
$$f^{k+1} < f^k \quad و \quad h^{k+1} > h^k,$$

در فیلتر جلوگیری می کند. اگر چه این دنباله تابع هدف را کاهش می دهد اما نقاط ناموجه تولید می کند و در هر مرحله بدتر می شود.

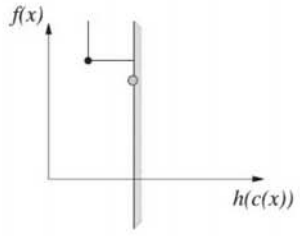
از طرفی می توانیم بعد از انجام چندین مرحله این کران را به روز کنیم. که این کار باعث می شود از پدیده دور جلوگیری شود. به عنوان مثال می توان کران را به صورت زیر، بعد از انجام j مرحله به روز کرد:

$$u = max(h^{k+1}, h^{k+2}, \cdots, h^{k+j}, \frac{u}{10}).$$

شکل های زیر به ترتیب یک فیلتر را قبل و بعد از به روز شدن کران بالا نشان می دهند.



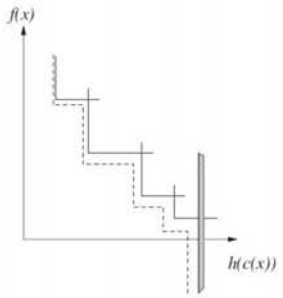
شکل ۱۲: قبل از به روز شدن کران بالا .



شکل ۱۳: بعد از بروز شدن کران بالا.

۲.۴. کاهش کافی. ممکن است وضعیتی به وجود آید که در آن نقاط در صفحه (f,h) در نزدیکی فیلتر تجمع کنند اما پیشرفتی نداشته باشند. یک راه استاندارد برای حل این مشکل در الگوریتم های تابع جریمه، لزوم کاهش کافی در تابع جریمه در هر مرحله می باشد. یک ایده ی مشابه در فیلتر این است که یک پوشش زیر فیلتر قرار دهیم. که از پذیرفتن نقاطی در فیلتر که به اندازه دلخواه نزدیک فیلتر باشند جلوگیری کنیم.

این ایده در شکل ۱۴ بیان شده:



شکل ۱۴: یک پوشش برای فیلتر .

در آن پوشش توسط خط چین مشخص شده است.

بنابراین یک مرحله جدید x^{k+1} در صورتی در فیلتر قابل قبول است که:

$$h^{k+1} \leq \beta h^k \quad \text{یا} \quad f^{k+1} \leq \gamma f^k,$$

که $\beta, \gamma \in (0,1)$ ثابت می باشند.

۳.۴. روش شمال غربی. توسط الگوریتم بالا ممکن است دنباله ای تولید شود که در آن دنباله $\{f_k\}$ به طور یکنواخت افزایش و $\{h_k\}$ به طور یکنواخت کاهش یابد بدون اینکه به جواب نزدیک شود. بنابراین برای جلوگیری از این امکان می توانیم یک شرط دیگر به صورت

$$f^{k+1} + \mu h^{k+1} \leq f^k + \mu h^k,$$

به شروط لازم اضافه کنیم. که در اینجا ثابت μ همانند پارامتر جریمه عمل می کند. اما بدلیل وجود شرط های دیگر اهمیت پارامتر جریمه را ندارد. و انتخاب آن چندان تعیین کننده نیست.

مراجع

[1] R. Fletcher and S. Leyffer, Nonlinear programming without a penalty function, *Math. Program.*, **91** (2002) 239–269.

[2] R. Fletcher, S. Leyffer and P. L. Toint, On the global convergence of a filter-SQP algorithm, *SIAM J. Optim.*, **13** (2002) 44–59.

[3] X. Zhu and D. Pu, A line search filter algorithm with inexact step computations for equality constrained optimization, *Appl. Num. Math.*, **62** (2012) 212-223.

[4] J. Nocedal and S. J. Wright, *Numerical Optimization*, 2nd ed., Springer, New York, 2006.

[5] W. Hock and K. Schittkowski, *Test Example for nonlinear programming codes*, Springer-Verlag, Berlin-New York, 1981.

نجمه حسینی منجزی
اصفهان، خیابان هزارجریب، دانشگاه اصفهان، گروه ریاضی
hoseini_najmeh@yahoo.com

نجمه حسینی منجزی متولد مرداد ماه ۱۳۶۸ در شهر نجف آباد اصفهان است. وی در سال ۱۳۸۶ وارد مقطع کارشناسی رشته ریاضی کاربردی دانشگاه اصفهان شد و در سال ۱۳۹۰ با سهمیه ی استعداد درخشان وارد مقطع کارشناسی ارشد رشته ریاضی کاربردی گرایش تحقیق در عملیات شد در تاریخ ۱۳۹۲/۶/۱۶ تحت نظر خانم دکتر صغری نوبختیان از پایان نامه ی خود که در زمینه بهینه سازی عددی بود با درجه عالی دفاع کرد. علایق پژوهشی وی در زمینه بهینه سازی عددی می باشد.