ریاضی و جامعه
شاپا (چاپی): ۶۴۹۳-۲۳۴۵، شاپا(الکترونیکی): ۶۵۰۷-۲۳۴۵
جلد ۵، شماره ۱ (۱۳۹۹)، صص. ۸۱-۶۹

http://www.ui.ac.ir
http://math-sci.ui.ac.ir

معرفی الگوریتم تقریبی دسته‌ای برای حل عددی مسائل بهینه‌سازی ناهموار

نجمه حسینی منجزی

چکیده. الگوریتم تقریبی دسته‌ای جزء الگوریتم‌های مناسب برای حل مسائل بهینه‌سازی ناهموار است. در ابتدا این الگوریتم برای حل مسائل بهینه‌سازی نامقیدِ ناهمواربِ محدب معرفی و پس از آن در طی سال‌ها برای حل مسائل مختلف گسترش داده شد. هدف این مقاله معرفی این الگوریتم و گسترش آن برای حل مسائل بهینه‌سازی ناهموار نامحدب نامقید است. در پایان مقاله الگوریتم تقریبی دسته‌ای برای حل مسائل نامحدب در نرم‌افزار MatLab را کدنویسی و نتایج حاصل از اجرای الگوریتم را برای چند مثال ارائه می‌کنیم.

۱. مقدمه

بهینه‌سازی [۱] شاخه‌ای از ریاضیات است که می‌توان آن‌را متعلق به هر دو بخش ریاضیات محض و کاربردی به شمار آورد. بهینه‌سازی از دو بخش کلیِ مدلسازی و حل مسئله تشکیل شده است. در بخش مدلسازی، مسائل جهان واقعی به صورت ریاضی فرمولبندی می‌شوند. در بخش دیگر، مسائل مختلف بیان و تلاش می‌شود برای حل آن‌ها روش‌هایی ارائه شود. بخش حل مسئله بسیار گسترده است و پژوهشگران با دیدگاه‌ها و روش‌های مختلفی در این زمینه فعالیت می‌کنند. عده‌ای درباره شرایط لازم و کافی جواب‌های مسائل پژوهش انجام می‌دهند و عده‌ای تلاش می‌کنند الگوریتم‌هایی برای حل مسائل مختلف ارائه کنند که توسط ماشین قابل اجرا باشد به طوری که پاسخگوی نیازهای مختلف در مسائل کاربردی باشد.

امروزه بهینه‌سازی به صورت یک اصل در تحلیل بسیاری از مسائل پیچیده استفاده می‌شود و در زمینه‌های مختلف جهان واقعی از جمله پزشکی، اقتصاد، مهندسی، نظریه کنترل و بخش‌های مختلف یادگیری ماشین [۲] ظاهر می‌شود. در حال حاضر زمینه فعالیت این شاخه گستردگی فراوانی یافته و در بسیاری از صنایع کوچک، شرکت‌ها، کتابخانه‌ها، بیمارستان‌ها، طراحی شهرها و حمل و نقل پایه بسیاری از برنامه‌ریزی‌ها شده است. به دلیل کاربرد گسترده این شاخه لزوم طراحی و پیاده‌سازی الگوریتم‌های مناسب برای حل مسائل مختلف اجتناب ناپذیر است.

بهینه‌سازی به معنای بیشینه‌سازی یا کمینه‌سازی یک تابع داده شده است که می‌تواند مقید به مجموعه‌ای از قیود باشد، در این حالت مسئله را بهینه‌سازی مقید [۳] و در غیر این صورت آن را بهینه‌سازی نامقید [۴] می‌نامند. مسائل بهینه‌سازی را

2010 *Mathematics Subject Classification.* 49j52, 65k05, 90C26.
عبارات و کلمات کلیدی. بهینه‌سازی ناهموار، بهینه‌سازی نامحدب،الگوریتم تقریبی دسته‌ای.
دبیرتخصصی رابط: صغری نوبختیان
تاریخ دریافت: ۱۳۹۹/۱۰/۰۳ تاریخ پذیرش: ۱۴۰۰/۰۱/۲۲
نوع مقاله: پژوهشی
http://dx.doi.org/10.22108/msci.2021.126616.1409

[1] Optimization [2] Machine Learning [3] Constrained [4] Unconstrained

از دیدگاه‌های مختلف می‌توان دسته‌بندی کرد. اگر در مسئله بهینه‌سازی لازم باشد چندین تابع به صورت همزمان به عنوان توابع هدف در نظر گرفته شوند (که ممکن است توابع در نظر گرفته‌شده با هم در تناقض نیز باشند) مسئله را یک مسئله بهینه‌سازی چندهدفه [۵] می‌نامند و در غیر این صورت یک مسئله معمولی (تک‌هدفه) داریم. در صورتی که توابع در نظر گرفته‌شده برای توابع هدف و قیود تا مرتبه معینی مشتق‌پذیر پیوسته باشند، مسئله را جزء بهینه‌سازی هموار [۶] دسته‌بندی می‌کنند و اگر حداقل یکی از توابع لزوماً مشتق‌پذیر نباشد، مسئله را یک مسئله بهینه‌سازی ناهموار [۷] می‌نامند. به دلیل اینکه در بهینه‌سازی از مشتق تابع به عنوان یک ابزار استفاده می‌شود و در بهینه‌سازی ناهموار از این ابزار نمی‌توان بهره برد، حل این دسته از مسائل چالش‌های مربوط به خودش را دارد. برای حل مسائل ناهموار به صورت کلی دو راه‌کار داریم. ابتدا اینکه می‌توانیم از روش‌های مشتق آزاد [۸] استفاده کنیم که در این روش‌ها با استفاده از تکنیک‌های جستجو تلاش می‌شود نقطه بهینه [۹] بدست آید. ثانیاً می‌توان از مشتق‌های تعمیم‌یافته مختلف که برای توابع ناهموار تعریف می‌شوند، همانند مشتق تعمیم یافته کلارک [۱۰]، بهره برد (که در اینجا از راه حل دوم استفاده می‌کنیم).

الگوریتم‌های متنوعی برای حل مسائل بهینه‌سازی ناهموار ارائه شده است. بعضی از این الگوریتم‌ها در ابتدا برای حل مسائل بهینه‌سازی هموار ارائه شده‌اند و پس از آن برای مسائل ناهموار توسعه یافته‌اند به عنوان مثال الگوریتم ناحیه اعتماد [۱۱] و روش شبه نیوتن [۱۲] از این دسته می‌باشند [۵، ۹]. از طرف دیگر الگوریتم‌هایی داریم که به طور خاص برای حل مسائل بهینه‌سازی ناهموار ارائه شده‌اند که معمولاً این الگوریتم‌ها در ابتدا برای حل مسائل محدب معرفی شده‌اند و سپس برای حل مسائل نامحدب گسترش یافته‌اند از جمله می‌توان به الگوریتم دسته‌ای [۱۳] و الگوریتم تقریبی دسته‌ای [۱۴] اشاره کرد [۸]. از طرفی الگوریتم‌های داریم که به طور خاص برای حل مسائل نامحدب ناهموار ارائه شده‌اند که در اینجا می‌توان به الگوریتم نمونه‌گیری گرادیان [۱۵] اشاره کرد [۲].

در این مقاله قصد داریم یک مسئله بهینه‌سازی ناهموار نامقید نامحدب در نظر بگیریم و الگوریتم تقریبی دسته‌ای را برای حل آن گسترش دهیم. الگوریتم دسته‌ای اولین‌بار در سال ۱۹۷۸ به‌دست لوماریشال [۱۶] برای حل مسائل بهینه‌سازی نامقید ناهموار محدب ارائه شد [۷]. نتایج عددی قوی در حل مسائل مختلف و نتایج همگرایی مناسب این الگوریتم باعث شد در طی سال‌های اخیر این الگوریتم جزء قوی‌ترین الگوریتم‌ها برای حل مسائل بهینه‌سازی ناهموار محدب درآید و تلاش‌هایی برای گسترش این الگوریتم برای حل مسائل مختلف انجام شود. برای بهبود بیشتر عملکرد این الگوریتم و در صورت امکان کاهش فرض محدب بودن تابع هدف، این الگوریتم به صورت ترکیبی با سایر تکنیک‌های بهینه‌سازی در نظر گرفته شده است که در این بین ترکیب الگوریتم دسته‌ای و تکنیک تقریبی منجر به یک الگوریتم قوی تحت عنوان الگوریتم تقریبی دسته‌ای برای حل مسائل بهینه‌سازی محدب شده است [۸]. پس از آن این الگوریتم با فرضیات مختلف برای حل مسائل بهینه‌سازی نامحدب گسترش یافته است [۴، ۶]. در این مقاله به طور خاص گسترش الگوریتم تقریبی دسته‌ای برای حل مسائل بهینه‌سازی ناهموار نامحدب نامقید مورد مطالعه قرار گرفته است.

ادامه مقاله در چهار بخش به صورت زیر است. در بخش دوم تعاریف مقدماتی از بهینه‌سازی و آنالیز ناهموار بیان می‌شود. روند کلی الگوریتم‌های عددی برای حل مسائل بهینه‌سازی در بخش سوم ارائه شده است. در بخش چهارم الگوریتم تقریبی دسته‌ای برای حل مسائل بهینه‌سازی ناهموار محدب بیان می‌شود و در بخش پنجم این الگوریتم برای حل مسائل بهینه‌سازی ناهموار نامحدب گسترش می‌یابد. به علاوه نتایج حاصل از اجرای این الگوریتم برای حل چندین مسئله بهینه‌سازی ارائه شده است.

[5] Multiobjective Optimization [6] Smooth [7] Nonsmooth [8] Derivative free methods [9] Optimal point [10] Clarke [11] Trust region Algorithm [12] Quasi-Newton Method [13] Bundle Algorithm [14] Proximal Bundle Algorithm [15] Gradient Sampling Method [16] Lemaréchal

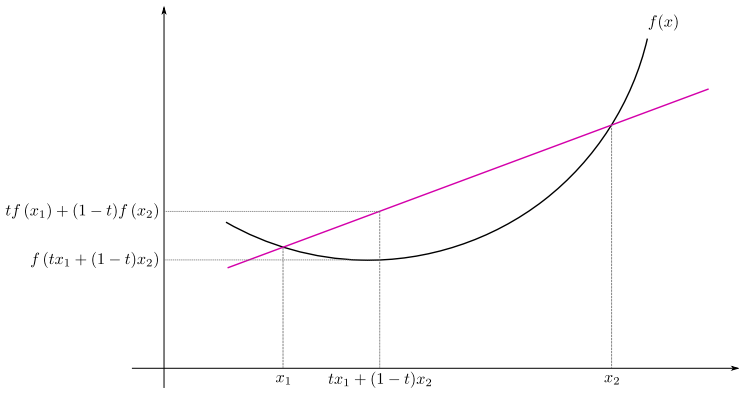
۲. مقدمات بهینه‌سازی و آنالیز ناهموار

ابتدا چند نماد معرفی می‌کنیم و پس از آن مفاهیم مقدماتی از بهینه‌سازی و آنالیز ناهموار را بیان خواهیم کرد. فرض کنید $\mathbb{R}$ و $\mathbb{R}^n$ به ترتیب نشان‌دهنده مجموعه اعداد حقیقی و فضای اقلیدسی nبُعدی باشند. ضرب داخلی دو بردار $u, v \in \mathbb{R}^n$ را به صورت $\langle u, v\rangle = \sum_{i=1}^{n} u_i v_i$ و نرم برداری متناظر با این ضرب داخلی را به صورت $\|u\| = \sqrt{\langle u, u\rangle}$ تعریف می‌کنیم.

تعریف ۱. تابع f را روی زیرمجموعه باز $U \subseteq \mathbb{R}^n$ محدب [۱۷] می‌نامیم هرگاه برای هر دو نقطه $x, y \in U$ داشته باشیم

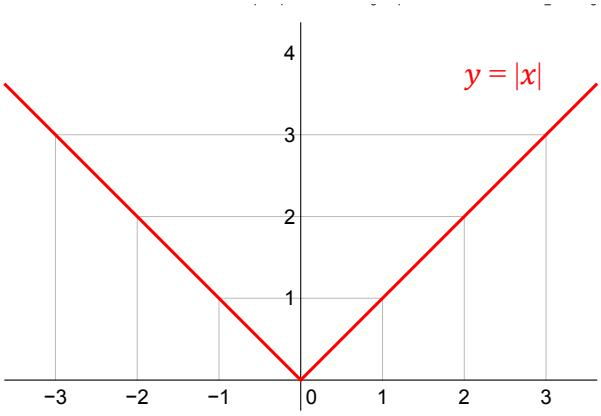
$$f(\lambda x + (1-\lambda)y) \leq \lambda f(x) + (1-\lambda) f(y) \qquad \forall \lambda \in [0, 1].$$

در شکل ۱ نمودار تابعی محدب با جزئیاتی مربوط به تعریف تابع محدب ارائه شده است.



شکل ۱: مثالی از یک تابع محدب

هر تابع مشتق‌ناپذیر را یک تابع ناهموار می‌نامیم. ساده‌ترین تابع ناهموار تابع قدر مطلق است که به صورت $f(x) = |x|$ تعریف می‌شود. این تابع در تمام نقاط غیرصفر مشتق‌پذیر است اما در صفر مشتق‌پذیر نیست. نمودار این تابع در شکل ۱ آمده است.



شکل ۲: نمودار تابع قدرمطلق $f(x) = |x|$

[17] Convex

http://dx.doi.org/10.22108/msci.2021.126616.1409

تعریف ۲. تابع $f : \mathbb{R}^n \to \mathbb{R}$ را در نقطه $x^* \in \mathbb{R}^n$ لیپ‌شیتسی موضعی [۱۸] از مرتبه $L > 0$ گوییم، اگر اسکالر $\delta > 0$ موجود باشد به طوری که

$$|f(x_1) - f(x_2)| \leq L\|x_1 - x_2\| \qquad \forall x_1, x_2 \in B_\delta(x^*).$$

فرض کنید تابع $f : \mathbb{R}^n \to \mathbb{R}$ در نقطه $x \in \mathbb{R}^n$ لیپ‌شیتسی موضعی از مرتبه L است.

تعریف ۳. [۳] مشتق جهتی کلارک [۱۹] تابع f در نقطه x در جهت $d \in \mathbb{R}^n$ به صورت زیر تعریف می‌شود

$$f^\circ(x; d) = \limsup_{\substack{y \to x \\ \alpha \downarrow 0}} \frac{f(y + \alpha d) - f(y)}{\alpha}.$$

تعریف ۴ ([۳]). زیردیفرانسیل [۲۰] کلارک (گرادیان تعمیم‌یافته) تابع f در نقطه x به صورت زیر تعریف می‌شود

$$\partial_c f(x) = \{\xi \in \mathbb{R}^n : \ \langle \xi, d\rangle \leq f^\circ(x; d) \qquad \forall d \in \mathbb{R}^n\},$$

و هر $\xi \in \partial_c f(x)$ یک زیرگرادیان [۲۱] (کلارک) نامیده می‌شود.

تعریف ۵. فرض کنید تابع $f : \mathbb{R}^n \to \mathbb{R}$ محدب است. زیردیفرانسیل محدب متناظر با این تابع به صورت زیر تعریف می‌شود

$$\partial f(x) = \{\xi \in \mathbb{R}^n : \ f(y) \geq f(x) + \langle \xi, y - x\rangle \qquad \forall y \in \mathbb{R}^n\},$$

و برای هر تابع محدب این زیردیفرانسیل بر زیردیفرانسیل کلارک منطبق است.

به عنوان مثال تابع $f(x) = |x|$ را در نظر بگیرید. همان‌گونه که قبلاً اشاره شد این تابع در همه نقاط بجز صفر مشتق‌پذیر است بنابراین

$$\nabla f(x) = \begin{cases} 1 & x > 0 \\ -1 & x < 0 \end{cases}.$$

حال نشان می‌دهیم این تابع محدب و لیپ‌شیتسی است. فرض کنید $x, y \in \mathbb{R}^n$ و $\lambda \in [0, 1]$ آنگاه با استفاده از نامساوی مثلث داریم

$$\begin{aligned} f(\lambda x + (1 - \lambda)y) &= |\lambda x + (1 - \lambda)y| \\ &\leq |\lambda x| + |(1 - \lambda)y| \\ &= |\lambda||x| + |1 - \lambda||y| \\ &= \lambda|x| + (1 - \lambda)|y| \\ &= \lambda f(x) + (1 - \lambda) f(y). \end{aligned}$$

بنابراین تابع f محدب است از طرفی به سادگی مشخص است که این تابع لیپ‌شیتسی موضعی با ثابت $L = 1$ است، زیرا

$$|f(x) - f(y)| = ||x| - |y|| \leq |x - y|.$$

[18] Locally Lipschitz [19] Clarke Directional Derivative [20] Subdifferential [21] Subgradient

بنابراین زیردیفرانسیل کلارک و زیردیفرانسیل محدب برای این تابع یکسان است. بنابراین برای محاسبه زیردیفرانسیل کلارک از تعریف زیردیفرانسیل محدب استفاده می‌کنیم و داریم

$$\begin{aligned}\xi \in \partial f(0) &\iff |y| \geq |0| + \langle \xi, y-0\rangle \qquad \forall y \in \mathbb{R}\\ &\iff |y| \geq \xi y \qquad \forall y \in \mathbb{R}\\ &\iff \xi \leq 1 \quad \text{و} \quad \xi \geq -1\\ &\iff |\xi| \leq 1.\end{aligned}$$

بنابراین داریم $\partial f(0) = \partial|0| = [-1, 1]$ که مجموعه‌ای غیرتهی محدب و فشرده می‌باشد و این ویژگی‌ها برای زیردیفرانسیل هر تابع لیپ‌شیتسی موضعی برقرار است.

۳. روش کار الگوریتم‌های عددی برای حل مسائل بهینه‌سازی

یک مسئله بهینه‌سازی نامقید در حالت کلی به صورت زیر تعریف می‌شود

$$\min \quad f(x), \quad x \in \mathbb{R}^n, \tag{۱}$$

که در آن $f : \mathbb{R}^n \to \mathbb{R}$ تابعی دلخواه و حقیقی مقدار است. الگوریتم‌های بهینه‌سازی معمولاً از یک نقطه اولیه $x^0 \in \mathbb{R}^n$ آغاز می‌شوند. روش کار الگوریتم‌های عددی به این ترتیب است که در هر مرحله با محاسبه یک جهت حرکت و پس از آن یک طول گام مناسب نقطه بعدی بدست می‌آید. فرض کنیم در هر مرحله kاُم و در نقطه x^k از الگوریتم قرار داریم. حال لازم است یک جهت حرکت d_k محاسبه شود و با شروع از نقطه x^k و حرکت در جهت بردار d_k و انتخاب طول گام $\alpha_k \in (0, 1]$ می‌توان نقطه آزمایشی جدید $x^{k+1} = x^k + \alpha_k d_k$ را بدست آورد. اگر الگوریتم از نوع کاهشی باشد (یعنی در هر مرحله تابع هدف نسبت به مرحله قبل کاهشی باشد) آنگاه لازم است نقطه آزمایشی بدست آمده در یک شرط کاهشی مثلاً به صورت زیر صدق کند

$$f(x^{k+1}) < f(x^k) - \delta_k,$$

که در آن $\delta_k > 0$ مقداری است که در هر مرحله از الگوریتم تعیین می‌شود یا می‌تواند در طی اجرای الگوریتم ثابت باشد.

برای طراحی یک الگوریتم بهینه‌سازی چند سوال اساسی مطرح می‌شود که با پاسخ به این سوال‌ها بخش اصلی الگوریتم مشخص می‌شود. اولین سوال این است که جهت حرکت چگونه محاسبه شود؟ دوم اینکه آیا همواره می‌توان یک جهت حرکت کاهشی بدست آورد؟ و سوم اینکه طول گام چگونه محاسبه شود؟

سوال دوم و پاسخ آن بستگی به نوع مسئله دارد. اگر مسئله مورد بررسی یک مسئله بهینه‌سازی هموار باشد همواره حداقل یک جهت حرکت کاهشی برای آن وجود دارد و این جهت به سادگی قابل ارائه می‌باشد. اگر تابع هدف f تابعی مشتق‌پذیر پیوسته باشد همواره یک انتخاب برای جهت کاهشی داریم. اگر تعریف کنیم $d_k = -\nabla f(x^k)$ آنگاه این جهت قطعاً کاهشی است. الگوریتم‌های متفاوت بهینه‌سازی هموار این جهت را به صورت‌های مختلف بکار می‌گیرند و هر کدام منجر به یک الگوریتم بهینه‌سازی متفاوت می‌شوند. معمولاً گرادیان در یک ماتریس ضرب می‌شود؛ در واقع تعریف می‌کنند $d_k = -A_k \nabla f(x^k)$. سپس با در نظر گرفتن مقادیر مختلف برای ماتریس A_k می‌توان الگوریتم‌های متفاوتی بدست آورد. به عنوان مثال اگر بگیریم $A_k = I$، یعنی ماتریس همانی را در نظر بگیریم، الگوریتم تندترین کاهش [۲۲] حاصل می‌شود. اگر $A_k = \nabla^2 f(x^k)$ در نظر گرفته شود روش نیوتن [۲۳] بدست می‌آید. حال اگر مشتق دوم تابع قابل محاسبه نباشد و یا محاسبه آن پرهزینه باشد می‌توان از ماتریس تقریب آن استفاده کرد؛ یعنی $A_k \simeq \nabla^2 f(x^k)$ که در این حالت روش شبه نیوتن حاصل می‌شود.

[22] Steepest Descent Algorithm [23] Newton Method

http://dx.doi.org/10.22108/msci.2021.126616.1409

حال اگر تابع هدف ناهموار باشد محاسبه جهت کاهشی کار ساده‌ای نیست. اگر فرض کنیم تابع هدف f لیپ‌شیتسی موضعی باشد در هر نقطه دلخواه $x \in \mathbb{R}^n$ لزوماً مشتق تعریف نمی‌شود و به‌جای آن زیردیفرانسیل تعریف می‌شود، که در اینجا ما با زیردیفرانسیل کلارک کار می‌کنیم که مجموعه‌ای غیرتهی، محدب و فشرده از $\mathbb{R}^n$ است. شاید ساده‌ترین راهی که به ذهن بیاید این باشد که یک عضو از زیردیفرانسیل یعنی یک زیرگرادیان دلخواه از تابع f در نقطه x^k اختیار کنیم، یعنی $\xi_k \in \partial_c f(x^k)$، آنگاه قرار دهیم $d_k = -\xi_k$. اما برخلاف مشتق، در اینجا این جهت لزوماً کاهشی نیست. به همین دلیل است که حل مسائل بهینه‌سازی ناهموار با چالش روبرو می‌شود و بکار بردن الگوریتم‌های مربوط به مسائل هموار را نمی‌توان به صورت مستقیم برای حل مسائل ناهموار به کار برد. در این میان مسائل بهینه‌سازی ناهموار محدب شرایط مطلوب‌تری نسبت به مسائل ناهموار نامحدب دارند. به دلیل اینکه زیردیفرانسیل کلارک برای تابع محدب همان زیردیفرانسیل محدب می‌باشد، در اینجا یک نامساوی داریم که به حل مسئله کمک می‌کند.

بعد از محاسبه جهت حرکت (که ممکن است کاهشی نباشد) و انتخاب طول گام مناسب نقطه بعدی به صورت x^{k+1} بدست می‌آید و الگوریتم تکرار می‌شود. بنابراین در طی اجرای الگوریتم یک دنباله از نقاط به صورت $\{x^k\}$ بدست می‌آید. همان‌گونه که می‌دانیم در عمل یک الگوریتم نمی‌تواند تعداد تکرارهای نامتناهی داشته باشد و لازم است الگوریتم در جایی متوقف شود. بنابراین الگوریتم نیاز به یک یا چند شرط توقف دارد. معمولاً یک شرط توقف برای بهینگی در نظر گرفته می‌شود و علاوه بر آن ممکن است چند شرط توقف دیگر نیز در نظر گرفته شود. در اینجا دو حالت ممکن است اتفاق بیافتد امکان دارد در طی اجرای الگوریتم و پس از تعداد متناهی تکرار به جواب مورد نظر برسیم و الگوریتم با یافتن جواب متوقف شود یا اینکه یک دنباله از نقاط به صورت $\{x^k\}$ بدست آید به طوری که این دنباله همگرا به جواب باشد.

۴. الگوریتم تقریبی دسته‌ای برای حل عددی مسائل محدب

یکی از الگوریتم‌های معروف برای حل مسائل بهینه‌سازی ناهموار الگوریتم تقریبی دسته‌ای است. این روش در ابتدا برای حل مسائل بهینه‌سازی محدب ارائه شده است در اینجا می‌خواهیم الگوریتم ارائه شده برای حل مسائل محدب را شرح دهیم.

در روش‌های دسته‌ای از دو ویژگی اصلی استفاده می‌شود: اول اینکه اطلاعات مربوط به زیرگرادیان‌ها و مقادیر تابع در تکرارهای قبلی در یک دسته جمع‌آوری می‌شوند و دوم اینکه در روش‌های دسته‌ای در طی اجرای الگوریتم به صورت همزمان دو مجموعه از نقاط تولید می‌شود که یکی مجموعه شامل نقاط جدی [۲۴] و مجموعه دیگر شامل نقاط کم‌اثر [۲۵] است و ممکن است دو مجموعه شامل نقاط مشترکی نیز باشند. مشابه اکثر الگوریتم‌های عددی بهینه‌سازی در اینجا با یک نقطه اولیه $x^\circ \in \mathbb{R}^n$ شروع می‌کنیم (که در بیشتر الگوریتم‌ها این نقطه می‌تواند به صورت دلخواه انتخاب شود) و این نقطه به عنوان اولین نقطه جدی در نظر گرفته می‌شود. فرض می‌کنیم در مرحله ℓام و در نقطه x^k قرار داریم و جهت حرکت d_ℓ محاسبه شده است (هنوز از کاهشی بودن این جهت اطمینان نداریم). تعریف می‌کنیم $y_\ell = x^k + d_\ell$. حال اگر این جهت کاهشی باشد یعنی در واقع در یک شرط کاهش کافی صدق کند یک گام جدی خواهیم داشت و قرار می‌دهیم $x^{k+1} = y_\ell$. در غیر این صورت یک گام کم‌اثر خواهیم داشت و قرار می‌دهیم $y_{\ell+1} = x^k + d_\ell$. بنابراین در طی اجرای الگوریتم دو مجموعه از نقاط تحت عنوان دنباله نقاط جدی $\{x^k\}$ و دنباله نقاط کم‌اثر $\{y_\ell\}$ ساخته می‌شوند. قبلاً به این نکته اشاره کردیم که بعد از محاسبه جهت حرکت لازم است یک طول گام نیز محاسبه شود که در الگوریتم‌های تقریبی دسته‌ای همواره طول گام را یک در نظر می‌گیریم یعنی برای هر $\ell = 1, 2, 3, \ldots$ تعریف می‌کنیم $\alpha_\ell = 1$.

مسئله بهینه‌سازی (۱) را در نظر بگیرید و فرض کنید $f : \mathbb{R}^n \to \mathbb{R}$ تابعی محدب و احتمالاً ناهموار باشد. بدلیل اینکه تابع ناهموار است بنابراین مشتق برای آن تعریف نشده است و ممکن است یافتن نقطه بهینه آن بسادگی امکان‌پذیر نباشد. از این‌رو در هر مرحله به‌جای محاسبه یک جهت حرکت یا جهت کاهشی برای تابع هدف یک مدل برای تابع هدف

[24] Serious Points [25] Null Points

می‌سازیم و با استفاده از این مدل یک زیرمسئله طراحی می‌شود و جهت حرکت برای مدل محاسبه می‌شود. تلاش می‌شود مدل تا حد امکان تقریب خوبی از تابع اصلی باشد و جهت حرکت قابل استفاده برای تابع اصلی باشد. در الگوریتم‌های بهینه‌سازی تلاش می‌شود مسئله به زیرمسئله‌هایی شکسته شود و در هر مرحله از الگوریتم یک زیرمسئله حل شود. معمولاً این زیرمسئله‌ها از نوع مسائل درجه ۲ [۲۶] و یا تکه‌ای خطی [۲۷] هستند. در اکثر الگوریتم‌های ناحیه اعتماد مدل به‌کار رفته در زیرمسئله‌ها درجه ۲ هستند و در الگوریتم‌های دسته‌ای این مدل تکه‌ای خطی در نظر گرفته می‌شود. حال می‌خواهیم برای مسئله بیان‌شده در فوق با استفاده از روش دسته‌ای یک زیرمسئله ارائه کنیم این زیرمسئله شامل یک مدل است که برای مسائل محدب این مدل یک کران پایین برای تابع هدف می‌باشد.

فرض کنید آخرین نقطه جدی محاسبه‌شده x^k و اندیس تکرار جاری ℓ باشد و نقاطی که در تکرارهای قبلی محاسبه شده‌اند به صورت $\{y_i\}_{i<\ell}$ باشند. مجموعه اندیس تکرارهای قبلی را در مجموعه $\mathcal{L}_\ell \subseteq \{0, 1, 2, \ldots, \ell-1\}$ نگهداری می‌کنیم و اطلاعات مربوط به نقاط قبلی را در دسته $\mathcal{B}_\ell \subseteq \cup_{i\in\mathcal{L}_\ell}\{(y_i, \xi_i, f(y_i))\}$ ذخیره می‌کنیم پس داریم $\xi_i \in \partial_c f(y_i)$. چون f تابعی محدب است طبق نامساوی برای زیرگرادیان توابع محدب داریم

$$f(y) \geq f(y_i) + \langle \xi_i, y - y_i\rangle \qquad \forall\, y \in \mathbb{R}^n,$$

که این رابطه برای هر $i < \ell$ برقرار است. رابطه سمت راست بالا خط است. این رابطه را بازنویسی می‌کنیم و با استفاده از آخرین نقطه جاری یعنی x^k می‌نویسیم

$$f(y) \geq f(x^k) + \langle \xi_i, y - x^k\rangle - e_i^k \qquad \forall\, y \in \mathbb{R}^n,\ i \in \mathcal{L}_\ell, \tag{۲}$$

که e_i^k به صورت زیر تعریف می‌شود

$$e_i^k = f(x^k) - f(y_i) - \langle \xi_i, x^k - y_i\rangle.$$

دقت کنید با استفاده از نامساوی که برای زیرگرادیان داریم به‌سادگی نتیجه می‌شود $e_i^k \geq 0$. در حل مسائل بهینه‌سازی محدب این نکته خیلی مهم است و برای مسائل نامحدب این رابطه را در اختیار نداریم. همین امر حل مسائل بهینه‌سازی نامحدب توسط روش دسته‌ای را مشکل می‌سازد و از این‌رو گسترش این الگوریتم برای مسائل نامحدب کار ساده‌ای نیست.

حال چون رابطه (۲) برای هر $i \in \mathcal{L}_\ell$ برقرار است می‌توانیم از عبارت سمت راست ماکسیمم بگیریم و یک تابع تکه‌ای خطی که کران پایین برای تابع هدف نیز هست حاصل می‌شود. این تابع تکه‌ای‌خطی را یک مدل در نقطه x^k در تکرار ℓ می‌نامیم و به صورت زیر نمایش می‌دهیم

$$f(y) \geq f(x^k) + \max_{i\in\mathcal{L}_\ell}\{-e_i^k + \langle \xi_i, y - x^k\rangle\} = M_\ell(y, x^k) \qquad \forall\, y \in \mathbb{R}^n. \tag{۳}$$

حال با حل مسئله بهینه‌سازی زیر

$$\min \quad M_\ell(y, x^k), \quad y \in \mathbb{R}^n,$$

نقطه آزمایشی بعدی را بدست می‌آوریم و درباره جدی یا کم‌اثر بودن این نقطه تصمیم خواهیم گرفت. اگر به‌جای محاسبه نقطه آزمایشی بخواهیم جهت حرکت محاسبه کنیم مسئله فوق را می‌توانیم به صورت زیر بنویسیم

$$\min \quad M_\ell(x^k + d, x^k), \quad d \in \mathbb{R}^n.$$

دقت کنید این مدل را با استفاده از اطلاعات مراحل قبل و نقاط در نزدیکی آخرین نقطه جدی یعنی x^k نوشته‌ایم بنابراین مدل بدست‌آمده یک مدل موضعی [۲۸] است و فقط تا همسایگی محدودی اطراف x^k این مدل می‌تواند تقریبی از تابع f باشد. علاوه بر این برای اینکه مسئله فوق جواب متناهی داشته باشد لازم است جستجو برای محاسبه نقطه آزمایشی بعدی

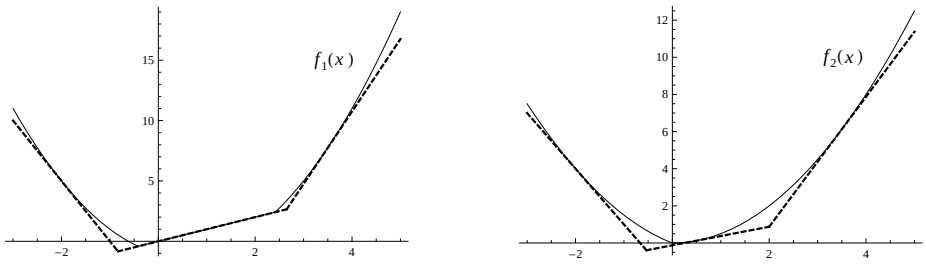
[26] Quadratic [27] Piecewise Linear [28] Local

http://dx.doi.org/10.22108/msci.2021.126616.1409

یا جهت حرکت بعدی را محدود کنیم. بنابراین می‌توانیم از روش ناحیه اعتماد یا روش تقریبی استفاده کنیم که در اینجا ما از روش تقریبی بهره می‌بریم و مدل زیر را می‌نویسیم

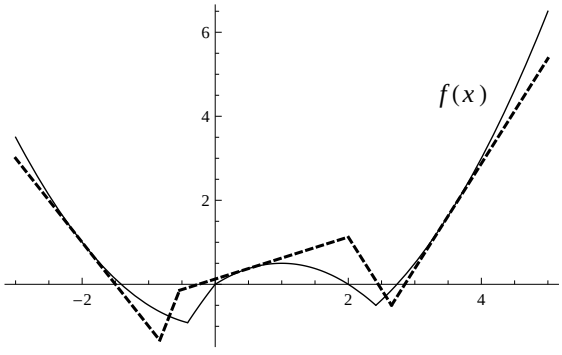
$$\min \quad M_\ell(x^k+d,x^k)+\frac{\mu_\ell}{2}\|d\|^2, \quad d\in\mathbb{R}^n,$$

که $\mu_\ell>0$ و پارامتر تقریبی نامیده می‌شود. با حل این مسئله جهت حرکت d_ℓ محاسبه می‌شود و نقطه آزمایشی x^k+d_ℓ محاسبه می‌شود. اگر این نقطه در شرط کاهشی کافی صدق کند یک نقطه جدی جدید داریم و می‌نویسیم $x^{k+1}=x^k+d_\ell$. حال در ادامه با به روزآوری مقادیر در مدل قبلی مدل جدید را حول نقطه جدی جدید می‌نویسیم و در مرحله بعد با تابع مدل $M_\ell(x^{k+1}+d,x^{k+1})$ کار می‌کنیم. در صورتی که نقطه آزمایشی جدید در شرط کاهش کافی صدق نکند یک نقطه کم‌اثر $y_\ell=x^k+d_\ell$ بدست می‌آید. در این صورت نقطه جدی قبلی بدون تغییر باقی می‌ماند و با استفاده از y_ℓ مدل تقویت می‌شود در واقع قرار می‌دهیم $\mathcal{L}_{\ell+1}=\mathcal{L}_\ell\cup\{\ell\}$ و تابع مدل برای مرحله به صورت $M_{\ell+1}(x^k+d,x^k)$ بدست می‌آید و الگوریتم تکرار می‌شود. در شکل ۳ طبق آنچه در رابطه (۳) بیان شد برای هر کدام از توابع محدب f_1 و f_2 یک مدل تکه‌ای خطی ارائه کرده‌ایم. دقت کنید در اینجا دسته $\mathcal{B}_\ell$ شامل اطلاعات سه نقطه می‌باشد.



شکل ۳: مدل تکه‌ای‌خطی برای توابع محدب f_1 و f_2

همان‌گونه که در بالا نیز اشاره کردیم اگر تابع هدف f محدب نباشد لزوماً مدلی که طبق رابطه (۳) نوشته می‌شود یک کران پایین برای تابع هدف نمی‌باشد به عنوان مثال به مدل بدست‌آمده برای تابع نامحدب f که در شکل ۴ نشان داده‌شده است توجه کنید.



شکل ۴: مدل تکه‌ای‌خطی برای تابع نامحدب f

۵. الگوریتم تقریبی دسته‌ای برای مسائل نامحدب

هدف این بخش گسترش الگوریتم تقریبی دسته‌ای برای حل مسائل بهینه‌سازی نامحدب ناهموار است. مهم‌ترین رابطه‌ای که در مسائل محدب داریم نامساوی (۲) است که منجر به این امر می‌شود که مدل بدست‌آمده یک کران پایین برای تابع باشد و علاوه بر این از این رابطه در اثبات همگرایی الگوریتم نیز استفاده می‌شود. برای توابع نامحدب این رابطه را در اختیار نداریم و سوالی که در اینجا مطرح می‌شود این است که چگونه برای تابع هدف مدل بنویسیم؟ از طرفی برای مسائل محدب e_i^kها همواره نامنفی هستند و این نکته مهمی در اثبات همگرایی الگوریتم می‌باشد که برای مسائل نامحدب این مورد را در اختیار نداریم و این مقدار ممکن است منفی باشد. در بعضی الگوریتم‌های ابتدایی تقریبی دسته‌ای برای رفع مشکل منفی بودن e_i^k در مسائل نامحدب قدرمطلق این مقدار در ساختن مدل در نظر گرفته می‌شد؛ به عنوان مثال [۸] را مشاهده کنید. در اینجا از روش ذکرشده نمی‌خواهیم استفاده کنیم و هدف این است با توجه به ساختار مسئله مدلی طراحی شود. سوال اساسی که در اینجا مطرح می‌شود چگونگی طراحی مدل است. قبل از ارائه مدل برای توابع نامحدب نتیجه‌ای را بیان می‌کنیم

قضیه ۱ ([۴]). *اگر $f:\mathbb{R}^n\to\mathbb{R}$ یک تابع لیپ‌شیتسی موضعی از پایین کراندار و lower-C^2 [۲۹] باشد، آنگاه $\overline{\eta}>0$ وجود دارد به طوری که به‌ازای $\eta\geq\overline{\eta}$ تابع $f+\frac{\eta}{2}\|\cdot-x\|^2$ تابعی محدب است.*

قضیه فوق یک ایده برای تعریف مدل برای توابع نامحدب ارائه می‌دهد و به کمک آن می‌توانیم الگوریتم تقریبی دسته‌ای را برای حل مسائل بهینه‌سازی نامحدب ناهموار گسترش دهیم. به‌جای اینکه همانند روش تقریبی دسته‌ای برای توابع محدب مدل را برای تابع هدف f بنویسیم مدل را برای تابع هدف تقویت‌شده یعنی $f_\eta(,x)=f(\cdot)+\frac{\eta}{2}\|\cdot-x\|^2$ می‌نویسیم. دقت کنید در اینجا کران $\overline{\eta}$ را نداریم اگر این مقدار مشخص بود کافی بود $\eta\geq\overline{\eta}$ بنویسیم و از مدل مربوط به تابع محدب استفاده کنیم. اما برای هر تابعی این کران $\overline{\eta}$ مشخص نیست و بنابراین در اینجا سعی می‌کنیم در هر مرحله از الگوریتم η به صورت دینامیکی و پویا محاسبه شود و در هر مرحله این مقدار به‌روز می‌شود.

فرض کنید الگوریتم در مرحله ℓام قرار دارد و آخرین نقطه جدی محاسبه‌شده x^k باشد. در این مرحله از تابع تقویت‌شده به صورت زیر استفاده می‌کنیم و مدل را برای این تابع می‌نویسیم

$$f_{\eta_\ell}(,x^k)=f(\cdot)+\frac{\eta_\ell}{2}\|\cdot-x^k\|^2.$$

مدل به صورت زیر در می‌آید

$$M_\ell(,x^k)=f(x^k)+\max_{i\in\mathcal{L}_\ell}\{-c_i^k+\langle\xi_i^k,y-x^k\rangle\},$$

که در آن

$$\xi_i\in\partial_c f(y_i),\quad e_i^k=f(x^k)-f(y_i)-\langle\xi_i,x^k-y_i\rangle,\quad c_i^k=e_i^k+\eta_\ell b_i^k,$$

$$\xi_i^k=\xi_i+\eta_\ell d_i^k,\quad d_i^k=y_i-x^k,\quad b_i^k=\frac{\|y_i-x^k\|^2}{2},$$

و مجموعه اندیس‌ها در مرحله ℓ به صورت $\mathcal{L}_\ell\subseteq\{0,1,2,\ldots,\ell-1\}$ و دسته متناظر با آن‌ها به صورت

$$\mathcal{B}_\ell\subseteq\bigcup_{i\in\mathcal{L}_\ell}\left\{(\xi_i,e_i^k,b_i^k,d_i^k)\right\}$$

[۲۹] در اینجا به تعریف این توابع نمی‌پردازیم چون این تعریف فراتر از این مقاله است. خواننده علاقه‌مند می‌تواند این دسته از توابع را در کتاب‌های آنالیز پیشرفته مشاهده کند و در صورت نیاز به منابع بیشتر می‌تواند با نویسنده این مقاله مکاتبه کند. در اینجا تنها برای اینکه مطلب به صورت دقیق بیان شود به این دسته از توابع اشاره شده است.

http://dx.doi.org/10.22108/msci.2021.126616.1409

است. در اینجا هدف این است که c_i^k برای هر $i \in \mathcal{L}_\ell$ نامنفی باشد. برای این هدف η_ℓ را به صورت زیر محاسبه می‌کنیم

$$\eta_\ell \geq \max\left\{\max_{i\in\mathcal{L}_\ell, y_i\neq x^k} \frac{-2e_i^k}{\|y_i - x^k\|^2}, \omega\right\} + \omega, \tag{۴}$$

به طوری که $\omega > 0$ ثابتی مثبت باشد. با استفاده از (۴) برای هر $i \in \mathcal{L}_\ell$ با شرط $y_i \neq x^k$ داریم

$$c_i^k = e_i^k + \frac{\eta_\ell}{2}\|y_i - x^k\|^2 \geq \frac{\omega}{2}\|y_i - x^k\|^2 \geq 0.$$

از طرف دیگر، اگر $y_i = x^k$ آنگاه داریم $c_i^k = 0$. بنابراین برای هر $i \in \mathcal{L}_\ell$ داریم $c_i^k \geq 0$ و شرطی که در حالت محدب برای e_i^k داشتیم در اینجا برای c_i^k برقرار است.

برای تعیین نقطه آزمایشی بعدی $y_{\ell+1}$ در هر مرحله از الگوریتم، یک زیرمسئله به صورت زیر حل می‌کنیم

$$\min \quad M_\ell(y, x^k) + \frac{\mu_\ell}{2}\|y - x^k\|^2, \quad y \in \mathbb{R}^n.$$

به روشنی مشخص است $y_{\ell+1}$ جواب یکتایی است. یا می‌توانیم به‌جای محاسبه نقطه آزمایشی جهت حرکت را به عنوان جواب بهینه مسئله زیر محاسبه کنیم

$$\min \quad M_\ell(x^k + d, x^k) + \frac{\mu_\ell}{2}\|d\|^2, \quad d \in \mathbb{R}^n.$$

واضح است که جواب مسئله یعنی d_ℓ یکتا می‌باشد. در این حالت نقطه آزمایشی را $y_{\ell+1} = x^k + d_\ell$ تعریف می‌کنیم. بر اساس یک شرط کاهش کافی درباره جدی یا کم‌اثر بودن نقطه آزمایشی تصمیم می‌گیریم و با توجه به اینکه نقطه جدی یا کم‌اثر باشد یا یک مدل جدید به‌دست می‌آوریم یا مدل تقویت می‌شود و الگوریتم تکرار می‌شود.

به صورت ساده می‌توان صورت کلی الگوریتم تقریبی دسته‌ای را به صورت زیر بیان کرد.

الگوریتم ۱. الگوریتم تقریبی دسته‌ای:

گام ۰ : نقطه شروع اولیه x^0 را انتخاب نمائید. تعریف کنید $y_0 = x^0$ و مقادیر $f(y_0)$ و $\xi_0 \in \partial f(y_0)$ را محاسبه کنید. تعریف کنید $k = 0$ و $\ell = 1$ و $\mathcal{L}_\ell = \{0\}$ و $\mathcal{B}_\ell = \{(y_0, \xi_0, f(y_0))\}$.

گام ۱ : مقدار $\mu_\ell > 0$ را انتخاب کنید و مینیمم مسئله (۵) را محاسبه کنید و فرض کنید جواب y_ℓ به‌دست آمده باشد. مقدار δ_ℓ را محاسبه کنید. شرط توقف را چک کنید اگر شرط توقف برقرار باشد توقف کنید. در غیر این صورت اگر نقطه به‌دست آمده در شرط کاهش کافی زیر

$$f(y_\ell) < f(x^k) - \delta_\ell,$$

صدق کند به گام ۲ بروید و در غیر این صورت به گام ۳ بروید.

گام ۲ : تعریف کنید $x^{k+1} = y_\ell$ و مدل را متناظر با نقطه جدی جدید به‌روز کنید و قرار دهید $k = k + 1$ و به گام ۱ بروید.

گام ۳ : با اضافه کردن نقطه y_ℓ مدل را تقویت کنید یعنی قرار دهید $\mathcal{L}_{\ell+1} \subseteq \mathcal{L}_\ell \cup \{\ell\}$ و $\mathcal{B}_{\ell+1} \subseteq \mathcal{B}_\ell \cup \{(y_\ell, \xi_\ell, f(y_\ell))\}$. قرار دهید $\ell = \ell + 1$ و به گام ۱ بروید.

لازم است اشاره کنیم که این الگوریتم بسیار آسان است و هیچ سختی‌ای برای انتخاب کردن پارامترهای مناسب ندارد. در هر تکرار در گام ۱ لازم است تنها یک زیرمسئله درجه ۲ حل شود. الگوریتم بالا به‌صورت کلی ارائه شده است و مثلاً درباره نحوه انتخاب μ_ℓ نکاتی بیان نشده است. مقدار δ_ℓ به صورت دقیق تعیین نشده است و جزئیات به‌روزآوری مدل و شرط توقف ذکر نشده است. تمام موارد ذکرشده را می‌توان به روش‌های مختلف اجرا کرد که هر کدام منجر به یک الگوریتم بهینه‌سازی می‌شود. خواننده علاقه‌مند برای جزئیات بیشتر به [۴، ۶] مراجعه کند.

کد الگوریتم ۱ را در برنامه MatLab R2016b پیاده‌سازی کردیم و با یک لپ‌تاپ شخصی با مشخصات

CPU : Intel Core I5-3210 M , 2.5 GHz,

Memory : 4 GB,

اجرا کردیم. الگوریتم را برای حل چند مثال از [۱] به کار گرفتیم و نتیجه اجرای الگوریتم برای هر کدام از مثال‌ها را به صورت یک نمودار که مقدار تابع هدف را بر اساس تعداد تکرارها نشان می‌دهد ارائه داده‌ایم. به‌سادگی از نمودارها مشخص است که این الگوریتم کاهشی است و در هر تکرار مقدار تابع هدف نسبت به تکرار قبل کاهش داشته است. به‌روشنی مشخص است در تمام مثال‌ها الگوریتم موفق به یافتن جواب بهینه شده است.

مثال ۱. مسئله بهینه‌سازی (۱) با تابع هدف زیر را در نظر بگیرید

$$f(x) = \max\{x_1^2 + x_2^4, (2 - x_1)^2 + (2 - x_2)^2, 2\exp(x_2 - x_1)\},$$

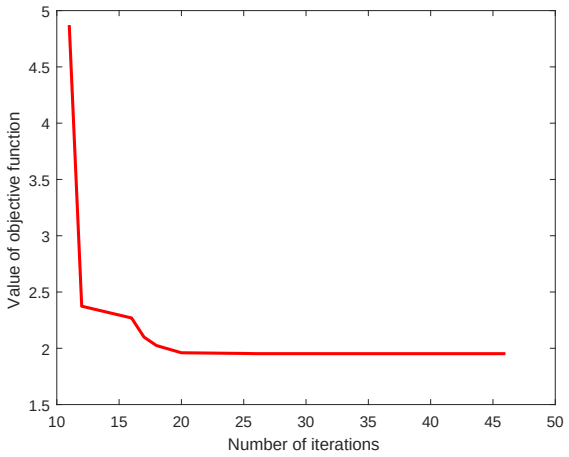
به طوری که

$$\texttt{point Starting}: \ x^\circ = (2, 2)^T,$$

$$\texttt{point Optimum}: \ x^* = (1/139286, 0/899365)^T,$$

$$\texttt{value Optimum}: \ f(x^*) = 1/9522245.$$

که نتایج حاصل از اجرای الگوریتم برای این مثال با نقطه شروع x° بالا در شکل ۵ نمایش داده شده است.



شکل ۵: نتایج حاصل از اجرای الگوریتم برای مثال ۱

مثال ۲. مسئله بهینه‌سازی (۱) با تابع هدف زیر را در نظر بگیرید

$$f(x) = \max\{f_1(x), f_2(x), f_3(x)\},$$

$$f_1(x) = x_1^2 + x_2^2,$$

$$f_2(x) = f_1(x) + 10(-4x_1 - x_2 + 4),$$

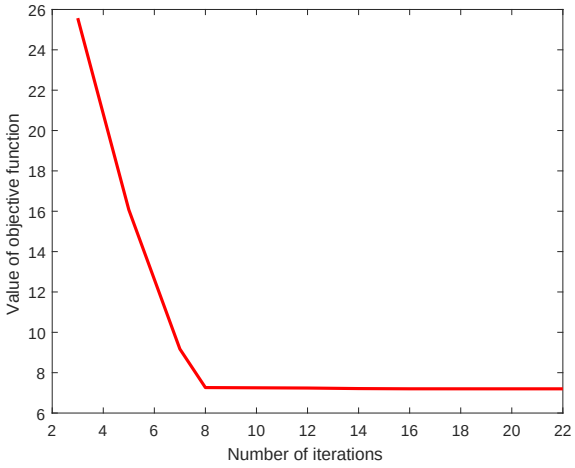
$$f_3(x) = f_1(x) + 10(x_1 - 2x_2 + 6),$$

http://dx.doi.org/10.22108/msci.2021.126616.1409

به طوری که

$$\begin{aligned}&\texttt{point Starting}: \ x^\circ = (-۱, ۵)^T,\\ &\texttt{point Optimum}: \ x^* = (۱/۲, ۱/۴)^T,\\ &\texttt{value Optimum}: \ f(x^*) = ۷/۲.\end{aligned}$$

که نتایج حاصل از اجرای الگوریتم برای این مثال با نقطه شروع x° بالا در شکل ۶ نشان داده شده است.



شکل ۶: نتایج حاصل از اجرای الگوریتم برای مثال ۲

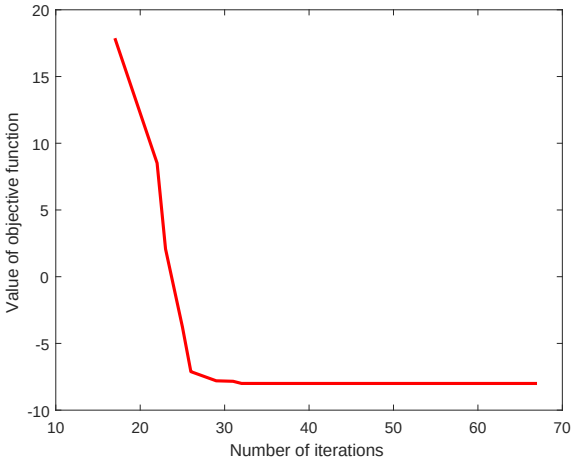
مثال ۳. مسئله بهینه‌سازی (۱) با تابع هدف زیر را در نظر بگیرید

$$f(x) = \begin{cases} ۵\sqrt{۹x_۱^۲ + ۱۶x_۲^۲} & if\ x_۱ \geq |x_۲| \\ ۹x_۱ + ۱۶|x_۲| & if\ ۰ < x_۱ < |x_۲| \\ ۹x_۱ + ۱۶|x_۲| - x_۱^۹ & if\ x_۱ \leq ۰ \end{cases},$$

به طوری که

$$\begin{aligned}&\texttt{point Starting}: \ x^\circ = (۳, ۲)^T,\\ &\texttt{point Optimum}: \ x^* = (-۱, ۰)^T,\\ &\texttt{value Optimum}: \ f(x^*) = -۸.\end{aligned}$$

که نتایج حاصل از اجرای الگوریتم برای این مثال با نقطه شروع x° بالا در شکل ۷ نمایش داده شده است.



شکل ۷: نتایج حاصل از اجرای الگوریتم برای مثال ۳

مراجع

[1] A. Bagirov, N. Karmitsa and M. M. Mäkelä, *Introduction to nonsmooth optimization* Theory, practice and software, Springer, Cham, 2014.

[2] J. Burke, A. Lewis and M. Overton, A robust gradient sampling algorithm for nonsmooth, nonconvex optimization, *SIAM J. Optim.*, **15** (2005) 571–779.

[3] F. H. Clarke, Y. S. Ledyaev, R. J. Stern and P. R. Wolenski, *Nonsmooth analysis and control theory*, Graduate Texts in Mathematics, 178, Springer-Verlag, New York, 1998.

[4] W. Hare and C. Sagastizábal, A redistributed proximal bundle method for nonconvex optimization, *SIAM J. Optim.*, **20** (2010) 2442–2473.

[5] N. Hoseini and S. Nobakhtian, A new trust region method for nonsmooth nonconvex optimization, *Optimization*, **67** (2018) 1265–1286.

[6] N. Hoseini Monjezi and S. Nobakhtian, A new infeasible proximal bundle algorithm for nonsmooth nonconvex constrained optimization, *Comput. Optim. Appl.*, **74** (2019) 443–480.

[7] C. Lemaréchal, *Bundle methods in nonsmooth optimization*, Nonsmooth optimization (Proc. IIASA Workshop, Laxenburg, 1977), IIASA Proc. Ser., 3, Pergamon, Oxford-Elmsford, N. Y., (1978) 79–102.

[8] M. M. Mäkelä and P. Neittaanmäki, *Nonsmooth Optimization: Analysis and Algorithms with Applications to Optimal Control*, World Scientific, Singapore, 1992.

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

نجمه حسینی منجزی

گروه ریاضی کاربردی و علوم کامپیوتر، دانشکده ریاضی و آمار، دانشگاه اصفهان، اصفهان، ایران

hoseini_najmeh@yahoo.com

نجمه حسینی منجزی دوره‌های کارشناسی و کارشناسی ارشد خود را در دانشگاه اصفهان در رشته ریاضی کاربردی گذرانده است. وی در شهریور ماه ۱۳۹۸ مدرک دکتری خود را در رشته ریاضی کاربردی گرایش تحقیق در عملیات از دانشگاه اصفهان اخذ نمود و از مهر ۱۳۹۸ به عنوان پژوهشگر پسادکتری در دانشگاه اصفهان فعالیت می‌کند. علایق پژوهشی وی در زمینه بهینه‌سازی ناهموار، بهینه‌سازی عددی و یادگیری ماشین می‌باشد.

http://dx.doi.org/10.22108/msci.2021.126616.1409