# تمرین سری هفتم درس روش‌های عددی بهینه‌سازی

امیرحسین افشارراد
۹۵۱۰۱۰۷۷

۶ دی ۱۳۹۸

## ۱ مروری بر نتایج تمرین چهارم

آن چه در ادامه‌ی این صفحه مشاهده می‌کنید، دقیقاً همان گزارشی است که برای تمرین سری چهارم (پیاده‌سازی الگوریتم‌های بهینه‌سازی برای دو تابع داده‌شده به کمک الگوریتم GSS) تحویل داده شد. در ادامه نتایج به دست آمده در تمرین جدید را با این نتایج مقایسه می‌کنیم.

**نکته ۱.** برای انجام line search به کمک تابع GSS، بازه‌ی جست‌وجو را به صورت $[0, 100]$ در نظر گرفتیم. نقطه‌ی شروع این بازه به وضوح صفر است، امّا نقطه‌ی پایانی آن می‌تواند متفاوت انتخاب شود و این انتخاب بر تعداد function evaluationها نیز اثر دارد.

**نکته ۲.** مقدار $\beta$ در اصلاح مسأله‌ی همگرایی روش نیوتن برابر صفر در نظر گرفته شده است.

**نکته ۳.** طبق توصیه‌ی صورت تمرین برای بررسی جواب‌ها با سایر دانشجویان، نگارنده‌ی این گزارش مقادیر عددی فوق را با آقای بهراد منیری چک کرده است و تشابه جواب‌های این گزارش با جواب‌های ایشان نیز به همین دلیل می‌باشد.

| | final $\mathbf{x}$ | final $f$ | number of iterations | number of function evaluations | number of gradient evaluations | number of Hessian evaluations |
|---|---|---|---|---|---|---|
| Rosenbrock function | $\begin{bmatrix}1.0033\\1.0067\end{bmatrix}$ | $1.1050\times10^{-5}$ | 184 | 12513 | 184 | 0 |
| Powel function | $\begin{bmatrix}0.2302\\-0.0230\\0.1101\\0.1190\end{bmatrix}$ | $0.0054$ | 181 | 12309 | 181 | 0 |

Table 1: Steepest Descent Method

| | final $\mathbf{x}$ | final $f$ | number of iterations | number of function evaluations | number of gradient evaluations | number of Hessian evaluations |
|---|---|---|---|---|---|---|
| Rosenbrock function | $\begin{bmatrix}1.0000\\1.0000\end{bmatrix}$ | $8.9059\times10^{-24}$ | 2 | 137 | 2 | 2 |
| Powel function | $\begin{bmatrix}-0.0001\\-0.0002\\0.8448\\0.8448\end{bmatrix}\times10^{-3}$ | $1.7404\times10^{-11}$ | 11 | 749 | 11 | 11 |

Table 2: Newton Method

## ۲ نتایج جدید پس از پیاده‌سازی Line Search با شرایط وُلف

**نکته ۱.** دو تابع برای پیاده‌سازی Line Search نوشته شده است؛ با نام‌های linesearch و zoom که به ترتیب مربوط به مراحل bracketing و sectioning هستند و بدنه‌ی اصلی کد آن‌ها از کتاب نوسدال گرفته شده است و برخی جزئیات به آن اضافه شده است.

**نکته ۲.** سه پارامتر مهم مسأله به صورت $c_1 = 10^{-4}$ و $c_2 = 0.1$ و $\bar{f} = 0.001$ انتخاب شده‌اند. $c_1$ و $c_2$ ضرایب موجود در شروط اوّل و دوم ولف هستند و $\bar{f}$، ثابتی است که (مطابق با توضیحات جزوه‌ی درس) برای تعیین $\alpha_{\max}$ و نیز اتمام زودهنگام فرایند linesearch (در صورت لزوم) به کار می‌رود.

**نکته ۳.** در آغاز کد مربوط به تابع zoom، نیاز به انجام یک درون‌یابی داریم. این درون‌یابی را با یک تابع درجه دوم انجام داده‌ایم و در ادامه مطابق با توضیحات جزوه، مقدار $\alpha_j$ جدید را از مقایسه‌ی محلّ مینیمم این سهمی با نقاط ابتدا و انتهای بازه به دست آورده‌ایم.

**نکته ۴.** مطابق با توضیحات جزوه‌ی درس، برای حالتی که $|\phi'(a_j)|$ از مقداری مانند $\epsilon$ کوچک‌تر می‌شود، الگوریتم sectioning متوقّف می‌شود و پیامی نیز بر روی صفحه چاپ می‌شود که همین موضوع را اطّلاع‌رسانی می‌کند. البته لازم به ذکر است در حلّ دو مسأله‌ی بهینه‌سازی مربوط به این تمرین، چنین موردی پیش نمی‌آید. همچنین مقدار $\epsilon$ مذکور برابر با $10^{-10}$ انتخاب شده است.

**نکته ۵.** مقدار $\alpha_1$ در تابع linesearch برابر با ۱ در نظر گرفته شده است. این یکی از روش‌هایی است که در جزوه پیشنهاد شده است، و همچنین در جزوه گفته شده که این روش برای الگوریتم نیوتن مناسب‌تر است؛ و جالب است که نهایتاً در نتایج نیز می‌بینیم که عملکرد الگوریتم نیوتن بهتر است (و شاید دلیل آن مربوط به این موضوع باشد).

در ادامه نتایج را مشاهده می‌کنید:

| | final $\mathbf{x}$ | final $f$ | number of iterations | number of function evaluations | number of gradient evaluations | number of Hessian evaluations |
|---|---|---|---|---|---|---|
| Rosenbrock function | $\begin{bmatrix}1.3507\\1.8255\end{bmatrix}$ | 0.1231 | 3 | 120 | 54 | 0 |
| Powel function | $\begin{bmatrix}0.2389\\-0.0238\\0.1139\\0.1237\end{bmatrix}$ | 0.0062 | 213 | 13266 | 8051 | 0 |

Table 3: Steepest Descent Method

| | final $\mathbf{x}$ | final $f$ | number of iterations | number of function evaluations | number of gradient evaluations | number of Hessian evaluations |
|---|---|---|---|---|---|---|
| Rosenbrock function | $\begin{bmatrix}1.0000\\1.0000\end{bmatrix}$ | 0 | 2 | 5 | 4 | 2 |
| Powel function | $\begin{bmatrix}0.0000\\-0.0000\\0.0011\\0.0011\end{bmatrix}$ | $3.5879\times10^{-11}$ | 12 | 35 | 33 | 12 |

Table 4: Newton Method

حال نتایج را با آن چه در صفحه‌ی قبل مشاهده می‌شود (مربوط به تمرین سری چهارم) مقایسه می‌کنیم:

۱. تابع روزنبروک با الگوریتم SD:
اگرچه حجم محاسبات کم‌تر شده، امّا دقّت جواب نهایی نیز کم‌تر شده است و به نظر می‌آید برای آن که به همان دقّت برسیم، لازم است حدّ آستانه‌ی اتمام الگوریتم را کم‌تر کنیم. البته خوب است توجّه کنیم که شرطی که در این کدها برای اتمام الگوریتم به کار رفته است، ناظر به اختلاف $x_k$ و $x_{k+1}$ است؛ و بدیهی است که این شرط لزوماً منتج به رسیدن به نقطه‌ی مینیمم نمی‌شود. در اصل این اتّفاق بسیار محتمل است که در فرایند تکرار الگوریتم بهینه‌سازی، در جایی الگوریتم بسیار کند شود و اگر به آن زمان کافی داده شود، دوباره سریع‌تر شده و به سمت مینیمم اصلی حرکت کند. (مانند آن که الگوریتم دارد از جایی سخت ـ مثلاً اطراف یک نقطه‌ی زینی ـ عبور می‌کند و مجبور است با گام‌های کوچک جلو برود؛ به گونه‌ای که ممکن است به نظر برسد که کار الگوریتم دارد تمام می‌شود؛ ولی اگر زمان کافی داده شود الگوریتم نهایتاً از این مقطع عبور می‌کند.) بنابراین احتمالاً می‌توان عملکرد الگوریتم را در مسأله‌ی بهینه‌سازی با کاهش مقدار آستانه بهتر کرد.

ضمناً یک نکته‌ی دیگر نیز شایان توجّه است و آن نکته این است که از مقایسه‌ی این نتیجه با نتیجه‌ی تمرین ۴، نتیجه می‌شود که پیاده‌سازی line search با الگوریتم GSS باعث شده بود تا نتیجه‌ی بسیار بهتری برای این مسأله حاصل شود، ولی نکته‌ی مهم آن است که این موضوع چندان قابل قضاوت نیست. در واقع مطابق با نکات بالای صفحه، اوّلاً پارامترهای زیادی در پیاده‌سازی به کمک شروط ولف مؤثّر هستند، و اصولاً نمی‌دانیم دقیقاً چه مقداری به عنوان $\alpha$ در هر گام از line search به دست می‌آید؛ و ممکن است با تغییری اندک در یکی از پارامترها، الگوریتم به گونه‌ای تغییر کند که پاسخش به این مسأله بهتر باشد، و البته شاید در عوض پاسخش به مسأله‌ی دیگری نیز بدتر شود!

۲. تابع پاول با الگوریتم SD:
در این مورد مشاهده می‌شود که دقّت جواب الگوریتم مشابه با دقّتی است که در تمرین چهارم به دست آورده بودیم؛ با این تفاوت که تعداد دفعات تکرار الگوریتم و حجم محاسبات افزایش یافته است. در این مورد نیز استدلالی مشابه با آن چه در قسمت قبل گفتیم برقرار است. یعنی ممکن بود با تغییر در روش انتخاب پارامترهای الگوریت، سریع‌تر به جواب برسیم. همچنین در این مقطع خوب است نکته‌ی ۵ صفحه‌ی قبل را نیز بررسی کنید.

۳. تابع روزنبروک با الگوریتم نیوتن:
مشاهده می‌شود که جواب همان دقّت قبلی را دارد (در واقع چون این تابع فرم مربّعی دارد، انتظار داریم که در دو تکرار روش نیوتن به جواب دقیق برسیم، و این نتیجه دقیقاً مشاهده شده است.) با این تفاوت که تعداد function evaluationها بسیار کم‌تر شده است. این نتیجه از محاسن استفاده از شروط ولف برای line search است، چرا که در روش قبلی، تعداد زیادی محاسبه‌ی تابع برای مینیمم‌کردن تابع $\phi$ استفاده می‌شود؛ در حالی که با این روش جدید، خیلی زود و با تعداد کمتری محاسبه‌ی تابع، مقداری مناسب برای $\alpha$ پیدا می‌شود و الگوریتم پیش‌روی می‌کند.

۴. تابع پاول با الگوریتم نیوتن:
مجدّداً مشاهده می‌شود که دقّت الگوریتم نسبت به تمرین چهارم تغییر نکرده، ولی حجم محاسبات کم شده و تعداد دفعاتی که تابع محاسبه می‌شود، تغییر معناداری کرده است. علّت این امر نیز مشابه با آن چیزی است که در قسمت قبل توضیح دادیم. البته یک نکته وجود دارد و آن نکته این است که در هر دو مورد اخیر، مشاهده می‌شود که تعداد محاسبات مربوط به گرادیان تابع (در مقایسه با روش GSS) افزایش یافته است. علّت این امر آن است که برای بررسی شروط ولف، نیاز به محاسبه‌ی مشتق تابع $\phi$ داریم و برای این کار، به گرادیان تابع اصلی نیاز پیدا می‌کنیم (در حالی که در GSS اصلاً چنین نیازی پیش نمی‌آید). با این حال خوب است دقّت کنیم که اگر چه تعداد محاسبات گرادیان اندکی افزوده شده است، امّا کاهش محاسبات خود تابع (که در GSS بسیار زیاد است) در مجموع غالب است، و در کل حجم محاسبات الگوریتم را کم‌تر کرده است.

نهایتاً به عنوان جمع‌بندی به نظر می‌رسد که الگوریتم جدید به صورت معناداری اجرای روش نیوتن را بهبود بخشیده است، در حالی که این بهبود برای روش SD چندان چشم‌گیر نیست. یکی از عللی که می‌تواند در این موضوع مؤثّر باشد، انتخاب $\alpha_1$ در تابع linesearch است (نکته‌ی ۵ صفحه‌ی قبل) که در آن، روشی استفاده شده است که برای الگوریتم نیوتن مناسب‌تر است. احتمالاً بتوان با تغییر این مورد، و نیز با تغییر دادن سایر پارامترها، به حالتی بهینه دست یافت که الگوریتم برای مسائل بهینه‌سازی‌ای که در این تمرین با آن‌ها مواجه هستیم بهتر عمل کند، ولی در حالت کلّی تضمینی وجود ندارد که برای هر مسأله‌ی بهینه‌سازی دلخواهی، عملکرد این الگوریتم‌ها بهتر از پیاده‌سازی به کمک GSS باشد. (البته این نکته را می‌دانیم که شروط ولف، همگرایی گلوبال را به هم نمی‌ریزند، ولی از قضایای ریاضی، تضمینی روی حجم محاسبات و زمان رسیدن به مینیمم ندارم.)