

رویکردی مبتنی بر الگوریتم ژنتیکی به حل مسئله قطب‌گماری با بازخور خروجی

ناصر ساداتی (دانشیار)

آزمایشگاه سیستم‌های هوشمند

دانشکده‌ی مهندسی برق، دانشگاه صنعتی شریف

در این نوشتار، روش جدیدی برای حل مسئله‌ی قطب‌گماری با بازخور (فیدبک) خروجی ثابت پیشنهاد شده است. روش پیشنهادی بر مبنای الگوریتم ژنتیکی شکل گرفته و می‌تواند تمامی قطب‌های سیستم حلقه‌بسته را با دقت بسیار در مکان‌های مورد نظر جای دهد. به شرط آن که جوابی برای مسئله وجود داشته باشد. همچنین نشان داده شده است که چگونه این الگوریتم، منجر به حل مشکلات موجود در روش‌های کلاسیک قطب‌گماری با بازخور خروجی ثابت شده است. مثال‌های عددی در پایان این نوشتار، کاربرد روش پیشنهادی را نشان می‌دهد.

مقدمه

سیستم خطی تغییرناپذیر با زمان زیر را در نظر بگیرید:

$$\begin{aligned} \dot{x}(t) &= A x(t) + B u(t) \\ y(t) &= C x(t) \end{aligned} \quad (1)$$

نیست، به‌علاوه بعضی از این الگوریتم‌ها فقط می‌توانند قطب‌های سیستم حلقه‌بسته را در مکان‌های مجزا جای دهند و در برخی دیگر از این روش‌ها، قطب‌های سیستم حلقه‌بسته نمی‌توانند منطبق بر قطب‌های سیستم حلقه‌باز باشد. [۱۰-۱۶]

تعیین همزمان مقادیر ویژه و بردارهای ویژه (ساختار ویژه)، روش دیگری برای حل مسئله‌ی قطب‌گماری با بازخور خروجی است. اگر این روش در حوزه‌ی فرکانس به کار گرفته شود، معادله‌ی مشخصه‌ی سیستم حلقه‌بسته با استفاده از خواص دترمینان‌ها و مشتق‌های آنها مورد بحث قرار می‌گیرد و در نتیجه، شکل بسته‌ی برای ماتریس K به دست می‌آید که برحسب مجموعه‌ی از بردارهای پارامتری بیان می‌شود. [۷-۱۱] از سوی دیگر، نتیجه‌ی تعیین همزمان مقادیر ویژه و بردارهای ویژه در حوزه‌ی زمان، معادلات سیلوستر دوگانه است که از حل آنها ماتریس بهره‌ی دلخواه به دست می‌آید. [۱۲-۱۳]

که در آن، $x \in \mathbb{R}^n$ ، $u \in \mathbb{R}^m$ و $y \in \mathbb{R}^l$. همچنین A ، B و C ماتریس‌های ثابت حقیقی با ابعاد مناسب‌اند. ماتریس‌های B و C ، از رتبه‌ی کامل فرض شده و جفت (A, B) کنترل‌پذیر و جفت (C, A) رؤیت‌پذیرند. مسئله‌ی تحت بررسی، یافتن ماتریس K با ابعاد $m \times n$ به گونه‌ی است که قطب‌های سیستم حلقه‌بسته تحت قانون بازخور $u = Ky$ در مکان‌های متقارن $\Lambda = \{\lambda_1, \lambda_2, \dots, \lambda_n\}$ قرار گیرند. در دو دهه‌ی اخیر، مسئله‌ی قطب‌گماری با بازخور خروجی ثابت، برای سیستم‌های چندورودی-چندخروجی مورد مطالعه قرار گرفته است. اولین نتایج نشان می‌دهد که اگر سیستم کنترل‌پذیر و رؤیت‌پذیر باشد، ماتریس بازخور خروجی ثابت می‌تواند $\max\{m, l\}$ قطب سیستم حلقه‌بسته را تقریباً به دلخواه تعیین کند. [۳-۱] مطالعات بیشتر نیز نشان داده است که تحت شرایط مشخصی، $\min\{n, m+l-1\}$ قطب به دلخواه تعیین می‌شود. [۴-۶] بنابراین اگر $n \leq m+l-1$ باشد، قطب‌گماری کامل با بازخور خروجی امکان‌پذیر است. اخیراً نیز ثابت شده است که شرط $m \times l \geq n$ که محدودیت کمتری ایجاد می‌کند — می‌تواند برای بررسی امکان قطب‌گماری مورد آزمایش قرار گیرد. [۱۴-۱۶] در اکثر این روش‌ها، الگوریتم یافتن ماتریس بازخور خروجی از لحاظ عددی مناسب

مسئله‌ی قطب‌گماری با بازخور خروجی را می‌توان از دو جنبه مورد مطالعه قرار داد:

- ۱) امکان قطب‌گماری با بازخور خروجی برای هر سیستم؛
- ۲) چگونگی به دست آوردن ماتریس بازخور خروجی برای قطب‌گماری دلخواه.

در این نوشتار، جنبه‌ی دوم مسئله‌ی قطب‌گماری مورد مطالعه قرار گرفته است. یعنی با فرض وجود جواب برای مسئله، الگوریتم جدیدی طراحی شده است تا قطب‌گماری با بازخور خروجی را انجام

دهد. در روش پیشنهادی، برای جستجوی ماتریس بهره‌ی مناسب، از الگوریتم ژنتیکی استفاده می‌شود و سپس ماتریس‌های به دست آمده با الگوریتم تکرارگرای دیگری تنظیم می‌شود. در این روش هیچ پیش‌فرضی درباره‌ی توزیع قطب‌های سیستم حلقه‌بسته وجود ندارد. در حالت کلی، شرط $m \times l \geq n$ نیز تضمین می‌کند که آزادی کافی در حل این مسئله وجود داشته باشد. اگرچه روش پیشنهادی می‌تواند مسئله را در حالت $m \times l < n$ نیز حل کند، به شرط آن که پاسخی برای مسئله وجود داشته باشد.

الگوریتم تکرارگرا

در این بخش، به شرح مختصر الگوریتم تکرارگرایی که از منبع شماره‌ی ۲۰ استخراج شده است می‌پردازیم. علت استفاده از این الگوریتم در روش پیشنهادی، نزدیک کردن ماتریس‌های بهره‌ی موجود است به ماتریس بهره‌ی مورد نظر که قطب‌گذاری دلخواه را انجام می‌دهد. ماتریس‌های بهره به عنوان عضوهای جمعیت (کروموزوم‌ها) در الگوریتم ژنتیکی کد شده است. همچنین در این بخش، علت اصلی ناکارایی الگوریتم تکرارگرا در یافتن ماتریس بازخور خروجی شرح داده می‌شود.

الف) الگوریتم تکرارگرای مقدماتی

سیستم توصیف شده با رابطه‌ی ۱ را در نظر بگیرید. اگر بازخور خطی $n \times K$ به کار گرفته شود، معادله‌ی مشخصه‌ی سیستم حلقه‌بسته به صورت زیر نتیجه می‌شود:

$$P_{closed}(s) = \det[sI - A - BKC] \quad (2)$$

حالت از دو روش مشابه - روش ستونی و روش سطری - می‌توان به عنوان راه حل تکرارگرای بهره جست. در روش ستونی، ماتریس بازخور خروجی K را بر حسب ستون‌هایش بسط می‌دهیم:

$$K = KI_{1 \times l} = \sum_{i=1}^l k_i \bar{e}_i = \sum_{i=1}^l k_i \bar{e}_i + K_j \bar{e}_j = K_j + k_j \bar{e}_j \quad (i \neq j) \quad (3)$$

که در آن i, j امین سطر ماتریس واحد $(l \times l)$ و K_j همان ماتریس K است که در ستون l ام آن صفر قرار می‌گیرد. بنابراین معادله‌ی مشخصه‌ی سیستم حلقه‌بسته به صورت زیر درمی‌آید:

$$\begin{aligned} P_{closed}(s) &= \det[sI - A - BK_j C - BK_j e_j^T C] \\ &= \det[sI - A - BK_j C] \cdot \det[sI - A - BK_j C]^{-1} BK_j e_j^T C \\ &= \det[sI - A - BK_j C] - e_j^T C \operatorname{adj}[sI - A - BK_j C] BK_j \\ &= p_j(s) - N_j(s) K_j \end{aligned} \quad (4)$$

$$\begin{aligned} \Lambda &= \{\lambda_{1j}\} U \{\lambda_{2j}\} U \dots U \{\lambda_{lj}\} \\ &= \Lambda_1 U \Lambda_2 \dots U \Lambda_l \end{aligned} \quad (5)$$

به نحوی که Λ_j ، برای به دست آوردن K_j استفاده می‌شود. الگوریتم تکرارگرای زیر برای قطب‌گذاری با بازخور خروجی، بر مبنای نکات فوق شکل می‌گیرد.

الگوریتم تکرارگرا:

$$\text{گام ۱: } K = K^0$$

گام ۲: به ازای $l = 1, 2, \dots, m$ ، مجموعه معادلات خطی زیر را حل کنید:

$$K_j = \begin{bmatrix} P_j(\lambda_{1j}) - P_{closed}(\lambda_{1j}) \\ P_j(\lambda_{2j}) - P_{closed}(\lambda_{2j}) \\ \vdots \\ P_j(\lambda_{lj}) - P_{closed}(\lambda_{lj}) \end{bmatrix} \quad (6)$$

که در آن $\lambda_{ij} \in \Lambda_j$ ، برای هر K_j به دست آمده، مقادیر قبلی را با مقادیر جدید جایگزین کنید.

گام ۳: مقادیر ویژه $\lambda_{i-closed}$ ماتریس سیستم حلقه‌بسته $A+BKC$ را محاسبه کنید. اگر

$$\sum_{i=1}^m (\lambda_i - \lambda_{i-closed}) < \epsilon$$

که $\epsilon > 0$ تولرانس مشخص است، تکرار را متوقف کنید. در غیر این صورت به گام ۲ برگردید.

نکته ۱: در معادله‌ی ۶، $P_{closed}(\lambda_{ij})$ برابر صفر است، زیرا $P_{closed}(s)$ معادله‌ی مشخصه سیستم حلقه‌بسته‌ی را که باید در قطب‌های مورد نظر برابر صفر شود، تعیین می‌کند. بنابراین معادله‌ی ۶ را می‌توان به شکل زیر نوشت:

$$K_j = \begin{bmatrix} P_j(\lambda_{1j}) \\ P_j(\lambda_{2j}) \\ \vdots \\ P_j(\lambda_{lj}) \end{bmatrix} \quad (7)$$

نکته ۲: الگوریتم سطری نیز به همین ترتیب و با در نظر گرفتن

الگوریتم ژنتیکی پیشنهادی

الگوریتم ژنتیکی راهی مناسب برای یافتن نقاط بهینه‌ی سراسری است، بویژه هنگامی که مسئله غیرخطی است و روش‌های متداول بهینه‌سازی که مبتنی بر مشتق‌گیری هستند، ناتوان از حل مسائل پیچیده‌اند. هر تسلی در الگوریتم ژنتیکی، از کروموزوم‌هایی تشکیل می‌شود که معرف شکل گذشته‌ی پاسخ‌های ممکن برای مسئله‌اند. تابع برازندگی که توسط طراح الگوریتم انتخاب می‌شود، تعیین می‌کند که هر کروموزوم تا چه میزان به بهترین پاسخ نزدیک است. این تابع مقدار برازندگی را برای هر کروموزوم مشخص می‌کند. بر مبنای این مقادیر شایستگی، بهترین کروموزوم‌ها به عنوان والدین نسل بعد انتخاب می‌شوند. والدین به صورت اتفاقی با هم ترکیب شده و سپس جهش داده می‌شوند تا جمعیت فرزندان را به وجود آورند. با تکرار این مراحل، کروموزوم‌ها به سمت بهترین جواب تغییر می‌کنند تا هنگامی که با برآورده شدن معیار توقف، اجرای الگوریتم پایان یابد. [۲۱-۲۳]

الگوریتم پیشنهادی مرحله‌ی دیگری دارد که متشکل از الگوریتم تکرارگرایی ارائه شده در بخش قبل است. در هر تکرار الگوریتم ژنتیکی، بعد از تشکیل جمعیت جدید، الگوریتم تکرارگرایی برای هر کروموزوم موجود در جمعیت اجرا می‌شود. گزینش، ترکیب و جهش به جای آن که بر کروموزوم‌های عضو جمعیت مستقیماً اثر کنند بر کروموزوم‌های به دست آمده از الگوریتم تکرارگرایی عمل می‌کنند.

الف) کد کردن و تولید جمعیت اولیه

پیش از این به اختصار، عوامل مؤثر بر همگرایی الگوریتم تکرارگرایی تشریح کردیم. از آنجا که الگوریتم پیشنهادی از الگوریتم تکرارگرایی استفاده می‌کند، باید شرایط اولیه‌ی گوناگونی برای اجرای آن فراهم کند. این شرایط اولیه به عنوان کروموزوم‌های جمعیت در الگوریتم ژنتیکی کد می‌شوند. برای پوشاندن تمامی شرایط اولیه، هر کروموزوم به نحوی کد می‌شود که شامل قسمت‌های زیر باشد:

۱. قطب‌های مطلوب سیستم حلقه‌بسته که چیدن آنها با ترتیب اتفاقی باعث توزیع متفاوت این مقادیر در زیرمجموعه‌های الگوریتم تکرارگرایی می‌شود.
۲. علامت درایه‌های ماتریس بازخور خروجی که مجموعاً $m \times l$ بیت یا مقدار ۱ یا ۰ هستند.
۳. قسمت صحیح درایه‌های ماتریس بازخور خروجی، که از سه رقم سه‌بین صفر تا نه تشکیل می‌شود. به این معنا که اندازه‌ی هر درایه به هزار محدود است.
۴. قسمت اعشاری درایه‌های ماتریس بازخور خروجی که از $m \times l$ مقدار بین صفر تا یک تشکیل می‌شود.

بسط ماتریس K نسبت به سطرهایش ساخته می‌شود. مزیت ایجاد الگوریتم‌های سطری و ستونی جداگانه این است که در بعضی مسائل، فقط الگوریتم ستونی به یافتن پاسخ مورد نظر کمک می‌کند و در مسائل دیگر، الگوریتم سطری تنها راه دستیابی به جواب است.

نکته‌ی ۳: مجموعه معادلات ۶ مشروط بر آن که $m \times l \geq n$ حداقل یک جواب دارد. این شرط به این معناست که تعداد معادلات از تعداد مجهولات بیشتر نباشد. در حالت خاص که $m \times l < n$ مجموعه معادلات ۶ ممکن است دارای جواب باشد یا نباشد.

نکته‌ی ۴: برای اجرای الگوریتم، مجموعه‌ی n باید به زیرمجموعه‌های مجزا (n_i) تقسیم شود. در صورتی که $m \times l > n$ باشد ولی m (یا l) را بخش نکند، n_i ها ممکن است تعداد عضوهای متفاوتی داشته باشند. در این حالت، از آنجا که تعداد معادلات از تعداد مجهولات کمتر است، از لحاظ محاسباتی بهتر است یک راه حل مشخص را یا ثابت فرض کردن برخی درایه‌های K در طول تکرارها آزمایش کنیم. [۲۰]

ب) کاستی‌های الگوریتم تکرارگرایی

الگوریتم تکرارگرایی فوق مستقیماً نمی‌تواند برای به دست آوردن ماتریس بهره یا بازخور خروجی به کار رود. عواملی که بر همگرایی الگوریتم تکرارگرایی تأثیر می‌گذارند عبارتند از:

۱. مقادیر اولیه‌ی درایه‌های ماتریس بازخور خروجی (K);
 ۲. توزیع قطب‌های دلخواه در m (یا l) زیرمجموعه;
 ۳. نوع الگوریتم مورد استفاده (سطری یا ستونی);
 ۴. تعیین درایه‌های ثابت ماتریس بهره و مقادیر ثابت آنها، در حالتی که $m \times l > n$ است ولی m (یا l) را بخش نکند.
- از آنجا که همگرایی الگوریتم وابستگی زیادی به هر یک از عوامل فوق دارد، نامشخص بودن هر یک از عوامل فوق پیش از اجرای الگوریتم، مانع از همگرایی مطلوب آن می‌شود.

در حالتی که مجموعه‌ی قطب‌های دلخواه سیستم حلقه‌بسته شامل مقادیر تکراری باشد، اگر جای دادن قطب‌های تکراری در زیرمجموعه‌های متفاوت امکانپذیر باشد، الگوریتم همگرا می‌شود. در غیر اینصورت، معادلات یکسانی برای مقادیر تکراری بوجود می‌آید که خاصیت یکسان بودن قطب‌ها را در مجموعه معادلات منعکس نمی‌کند. علاوه بر آن که شرط توزیع اجباری قطب‌های تکراری در زیرمجموعه‌های متفاوت، همگرایی الگوریتم را تحت تأثیر قرار می‌دهد. قطب‌هایی که تکرار آنها بزرگ‌تر از m یا l باشد با این الگوریتم به دست نمی‌آیند.

۲	۴	۱	۳	۱	۱	۱	۰	۳	۲	۴	۵	۷	۰	۰	۲	۰	۴	۱۰	۶	۷	۰	۹	۸	۰/۲۱	۰/۰۱	۰/۳۴	۰/۷۶	۰/۱	۰/۵	۱	۴	۵
عناصر بیت		قسمت اختاری عناصر		قسمت صحیح عناصر ماتریس بهره										علامت عناصر		قطب‌های																
تشانگر الگوریتم		ماتریس بهره												ماتریس بهره		حلقه‌بسته با																
																ترتیب تصادفی																

شکل ۱. نمونه‌یی از کروموزوم موجود در جمعیت.

مجموعه از معادلات خطی، حتی اگر تعداد مجهولات بیشتر از معادلات باشد، به صورت مناسبی حل شود. این انتخاب همچنین محدودیت ثابت کردن برخی درایه‌های ماتریس بازخور خروجی در حالت $m \times l > n$ را از بین می‌برد.

اگر مجموعه قطب‌های سیستم حلقه‌بسته شامل مقادیر یکسان باشد، معادله‌ی ۴ باید به نحوی تغییر یابد که معادلات متفاوتی برای قطب‌های یکسان به وجود آید. ابتدا معادله‌ی ۴ به صورت زیر نوشته می‌شود:

$$P_{closed}(s) / \det[sI - A - BK_j C] = 1 - e_j^T C (sI - A - BK_j C)^{-1} BK_j \quad (8)$$

اگر λ_{ij} ، i امین مقدار ویژه در زیرمجموعه‌ی r تکرار باشد، آنگاه:

$$P_{closed}(s) |_{s=\lambda_{ij}} = 0 \quad (9)$$

$$d^k / ds^k P_{closed}(s) |_{s=\lambda_{ij}} = 0 ; k = 1, 2, \dots, r-1 \quad (10)$$

توجه کنید که: [۸]

$$d^k / ds^k (sI - A) |_{s=\lambda_{ij}} = (-1)^k (\lambda_{ij} I - A)^{-(k+1)} \quad (11)$$

حال با مشتق‌گیری درجه‌ی k از طرفین معادله‌ی ۸ و جایگزینی رابطه‌های ۹، ۱۰ و ۱۱ در آن، معادله‌ی برداری (۱۲) زیر به دست می‌آید:

$$e_j^T C (\lambda_{ij} I - A - BK_j C)^{-1} BK_j = 1$$

$$e_j^T C (-1)^K (\lambda_{ij} I - A - BK_j C)^{-(K+1)} BK_j = 0$$

$$K = 1, 2, \dots, r-1 \quad (12)$$

معادلات ۱۲ مستقل‌اند و برقراری همزمان آنها با تعیین r قطب یکسان به صورت همزمان معادل است. توجه کنید که برای به دست آوردن ماتریس بازخور خروجی حقیقی، قطب‌های مزدوج مختلط باید در یک زیرمجموعه قرار گیرند. همچنین، اگر قطب‌های دلخواه سیستم حلقه‌بسته شامل مقادیر مزدوج مختلط باشند، معادله‌ی ۸ را می‌توان به نحوی تغییر داد که خطاهای عددی ناشی از محاسبه‌ی $(\lambda_{ij} I - A - BK_j C)^{-1}$ در حقیقی بودن ماتریس بازخور خروجی

۵. بیت مشخصه‌ی الگوریتم، که نشانگر نوع الگوریتم تکرارگرایی مورد استفاده است.

۶. $m \times l - n$ به درایه‌های ماتریس بازخور خروجی در حالت $m \times l > n$ اشاره دارد. این اشاره گرها مشخص می‌کنند که چه تعداد قطب در هر زیرمجموعه باید جای بگیرد.

به عنوان مثال، سیستم درجه‌ی چهار با سه ورودی و دو خروجی را در نظر بگیرید. قطب‌های دلخواه سیستم حلقه‌بسته -1 ، -2 ، -3 و -4 است. در این حالت ۲ اشاره‌گر $(3 \times 2 - 4 = 2)$ وجود دارد. در شکل ۱، یکی از کروموزوم‌های موجود در جمعیت نشان داده شده است. این کروموزوم شکل گذشته‌ی ماتریس بهره‌ی زیر است:

$$K = \begin{bmatrix} -32/21 & 45/0.1 \\ 2/34 & -41/76 \\ -67/1 & 98/5 \end{bmatrix}$$

از آنجا که بیت مشخصه‌ی الگوریتم ۱ است، الگوریتم تکرارگرایی ردیفی برای این کروموزوم به کار گرفته می‌شود. $\{-4\}$ و $\{-1\}$ و $\{-3\}$ زیرمجموعه‌هایی هستند که برای حل مجموعه معادلات متناظر با سطر اول، دوم و سوم ماتریس K به کار می‌روند. الگوریتم ژنتیکی پیشنهادی، با تولید اتفاقی 50 کروموزوم آغاز می‌شود. تعداد این کروموزوم‌ها قبل از اجرای الگوریتم ممکن است تغییر کند.

ب) پیشرفت جمعیت

در هر مرحله از الگوریتم ژنتیکی، کروموزوم‌های موجود توسط الگوریتم تکرارگرایی تغییر می‌یابند. الگوریتم سطری یا ستونی هر کروموزوم را به نحوی تکامل می‌دهد که اختلاف بین قطب‌های دلخواه و قطب‌های سیستم حلقه‌بسته کوچک شود. الگوریتم تکرارگرایی می‌تواند تا ده بار برای هر کروموزوم اجرا شود. اگر نتیجه‌ی نکرارها و اگرایی درایه‌های ماتریس K باشد، الگوریتم تکرارگرایی متوقف می‌شود.

روش‌های مختلفی برای حل مجموعه معادلات ۶ وجود دارد. چون نرم‌افزار مورد استفاده MATLAB است، از تابع "pinv" برای یافتن جواب استفاده می‌شود. این انتخاب اجازه می‌دهد که هر

هر کروموزوم نشان می‌دهد که چه مقدار قطب‌های سیستم حلقه‌بسته به مقادیر مورد نظر نزدیک‌اند. واضح است که یکی از آسان‌ترین روش‌ها برای تعریف تابع برازندگی، در نظر گرفتن مجموع مربعات فاصله‌ی بین قطب‌های مطلوب و قطب‌های سیستم حلقه‌بسته است. مقدار برازندگی هر کروموزوم می‌تواند برابر یا معکوس این فاصله تعریف شود. اما آزمایش بر روی سیستم‌های مختلف نشان می‌دهد که با این معیار، الگوریتم در دام نقاط بهینه‌ی محلی گرفتار می‌شود. اگر M_j را λ مین کروموزوم جمعیت و K_j را ماتریس بهره‌ی متناظر با آن فرض کنیم، آنگاه:

$$S = (s_{i_1}, s_{i_2}, \dots, s_{i_n}) = \text{eigen}(A + B K_j C)$$

حال قطب‌های دلخواه سیستم حلقه‌بسته را به نحوی مرتب می‌کنیم که هر قطب در مکان نزدیک‌ترین قطب در بردار S قرار گیرد:

$$\Lambda_{new} = (\lambda_{i_1}, \lambda_{i_2}, \dots, \lambda_{i_n})$$

درصد فاصله‌ی نسبی ماتریس بهره (K_j) نیز به صورت زیر تعریف می‌شود:

$$\text{Rel-distance}(K_j) = 100 \times \text{absolute} \{ (\lambda_{i_1} - s_{i_1}) / \lambda_{i_1}, (\lambda_{i_2} - s_{i_2}) / \lambda_{i_2}, \dots, (\lambda_{i_n} - s_{i_n}) / \lambda_{i_n} \} \quad (15)$$

اگر λ_i صفر باشد، $(\lambda_i - s_i) / \lambda_i$ جایگزین $(\lambda_i - s_i)$ می‌شود. در نهایت، مقدار برازندگی کروموزوم M_j طبق رابطه‌ی زیر تعریف می‌شود:

$$\text{Fitness}(M_j) = (\text{Max} \{ \text{Rel-distance}(K_j) \})^{-2} \quad (16)$$

این تعریف از تابع برازندگی، الگوریتم را قادر می‌سازد که قطب‌های سیستم حلقه‌بسته را نزدیک به مقادیر کوچک همانند مقادیر بزرگ قرار دهد. تعداد کروموزوم‌های انتخاب شده برای مرحله‌ی ترکیب در تمامی نسل‌ها 40 در نظر گرفته می‌شود (با فرض این که 50 کروموزوم در جمعیت وجود دارد).

تقاطع (ترکیب)

عملگر تقاطع در الگوریتم ژنتیکی برای تولید فرزندان از والدین به کار می‌رود. از آنجا که هر کروموزوم در الگوریتم طراحی شده از اعدادی در محدوده‌های گوناگون تشکیل شده است، عملگر تقاطع بر هر قسمت جداگانه اثر می‌کند. برای دستیابی به بهترین نتیجه، جابه‌جایی 5 نقطه‌ی در این الگوریتم به کار گرفته می‌شود؛ یک نقطه در قسمت علامت، سه نقطه در بخش صحیح درایه‌ها و یک نقطه در بخش اعشاری و بیت مشخصه‌ی الگوریتم. هر جفت از کروموزوم‌هایی که برای تقاطع انتخاب می‌شوند با احتمال مشخص

نداشته باشد. فرض کنید λ_{ij} و λ_{i+j} دو قطب مزدوج مختلط در زیر مجموعه‌ی λ باشند، آنگاه:

$$\begin{aligned} e_j^T C (\lambda_{ij} I - A - BK_j C)^{-1} BK_j &= 1 \\ e_j^T C (\lambda_{i+j} I - A - BK_j C)^{-1} BK_j &= 1 \end{aligned} \quad (13)$$

از آنجا که $(\lambda_{ij} I - A - BK_j C)^{-1}$ و $(\lambda_{i+j} I - A - BK_j C)^{-1}$ نیز مزدوج مختلط‌اند، به آسانی می‌توان نشان داد که:

$$\begin{aligned} e_j^T C \text{Re} \{ (\lambda_{ij} I - A - BK_j C)^{-1} BK_j \} &= 1 \\ e_j^T C \text{Imag} \{ (\lambda_{ij} I - A - BK_j C)^{-1} BK_j \} &= 0 \end{aligned} \quad (14)$$

یکی از معادلات فوق برای اولین قطب مختلط و دیگری برای مزدوج آن به کار خواهد رفت.

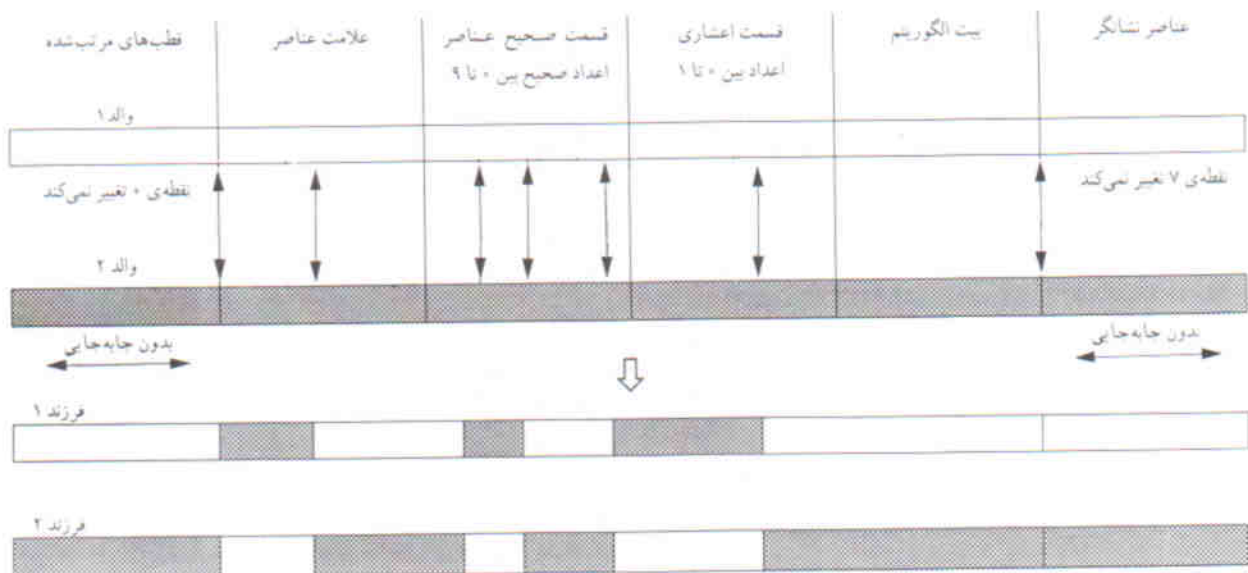
در پایان الگوریتم تکرارگرا، اگر ماتریس بهره‌ی به دست آمده محدود بماند (همگرایی)، کروموزوم مشابهی به الگوریتم اصلی بازگردانده می‌شود، به غیر از قسمت ماتریس بهره که با ماتریس به دست آمده از تکرارها جایگزین می‌شود. در غیر این صورت، قطب‌ها در قسمت اول این کروموزوم به صورت اتفاقی جابه‌جا می‌شوند و بیت مشخصه‌ی الگوریتم نیز تغییر می‌کند. افزون بر آن، قسمت بهره نیز جایگزین آخرین ماتریس بهره‌ی به دست آمده قبل از واگرایی الگوریتم می‌شود. همگرایی الگوریتم تکرارگرا با آزمایش وضعیت ماتریس $(\lambda_{ij} I - A - BK_j C)$ مشخص می‌شود. وقتی درایه‌های K به سمت بی‌نهایت میل می‌کنند، $\lambda_{ij} I - A - BK_j C$ منفرد می‌شود.

ج) مراحل ژنتیکی الگوریتم پیشنهادی

برای یافتن ماتریس بهره‌ی که قطب‌های سیستم حلقه‌بسته را در مکان‌های مشخص جای دهد، الگوریتم ژنتیکی هر یک از کروموزوم‌ها را طی نسل‌های متناهی تکامل می‌دهد. عملگرهای ژنتیکی جمعیت فرزندان را از جمعیت والدین تولید می‌کند تا خصوصیات جدیدی در جمعیت به وجود آید، اگرچه فرزندان همواره بخشی از خصوصیات والدین را به ارث می‌برند. ترکیب خصوصیات جدید و قدیم، همراه با تنظیم ماتریس‌های بهره از طریق الگوریتم تکرارگرا، جمعیت را به سمت نقطه‌ی بهینه سوق می‌دهد.

گزینش و تابع برازندگی

الگوریتم ژنتیکی بهترین کروموزوم‌های هر نسل را به عنوان والدین نسل بعد انتخاب می‌کند. برای گزینش بهترین کروموزوم‌ها سازوکار چرخ رولت به کار گرفته می‌شود، همان طور که برای ارزیابی هر کروموزوم معیار برازندگی باید تعریف شود. مقدار برازندگی متناظر با



شکل ۲. نمونه‌یی از ترکیب کروموزوم‌ها.

جدول ۸. قوانین جهش در الگوریتم پیشنهادی

تأثیر جهش بر عناصر انتخاب شده	وضعیت تولید عدد تصادفی	
طرب در ۱-	برای هر مقدار یک عدد تصادفی تولید می‌شود	قسمت علامت
بیت تصادفی جدید جایگزین می‌شود	برای هر مقدار یک عدد تصادفی تولید می‌شود	قسمت صحیح
اضافه کردن یک عدد تصادفی	برای هر مقدار یک عدد تصادفی تولید می‌شود	قسمت اعشاری
۱ به ۰ تغییر می‌کند و بالعکس	برای کلیه‌ی مقادیر یک عدد تصادفی تولید می‌شود	بیت الگوریتم
مجموعه‌ی جدیدی از عناصر نشانگر جایگزین نشانگرهای قبلی می‌شوند	برای کلیه‌ی مقادیر یک عدد تصادفی تولید می‌شود	نشانگر

تمامی روش‌های فوق برای جهش به این منظور به کار گرفته می‌شوند که خواص جدیدی را در جمعیت بوجود آورند تا در کوتاه‌ترین زمان ممکن پاسخ مطلوب به دست آید.

ساختار کلی الگوریتم پیشنهادی

تاکنون تمامی مراحل الگوریتم پیشنهادی را شرح داده‌ایم. حال با دانستن این مراحل نشان خواهیم داد که چگونه این الگوریتم، ماتریس بازخورد خروجی مناسب برای قطب‌گماری دلخواه را می‌یابد.

ماتریس بهره‌ی مورد نظر طی ۴ مرحله محاسبه می‌شود که هر یک از این مراحل از ساختار ژنتیکی برخوردار است. در مرحله‌ی اول، پس از تولید جمعیت اولیه به صورت اتفاقی، کروموزوم‌ها توسط الگوریتم تکرارگر اندکی کامل می‌شوند. در این مرحله جهش به قسمت اعشاری درایه‌های ماتریس بهره اثر نمی‌کند. نرخ جهش ۸٪ است مگر در قسمت اشاره‌گرها که نرخ جهش در آن به ۲۵٪ افزایش می‌یابد. در پایان هر یک از تکرارهای الگوریتم، پنج والد برتر نیز به همراه فرزندان به نسل بعد راه می‌یابند.

در نقاط انتخاب شده شکسته شده و جایجایی بین نقاط زوج و فرد صورت می‌گیرد. در نتیجه دو کروموزوم فرزند به جمعیت والدین اضافه می‌شوند. شکل ۲ چگونگی انجام این کار را نشان می‌دهد. در این الگوریتم، احتمال جابه‌جایی ۰/۸ است که به این معناست که $0.8 \times 4 \times 2 = 6.4$ کروموزوم در جمعیت وجود خواهد داشت.

جهش

در این بخش، قوانین کلی جهش شرح داده خواهد شد. جهش همانند تقاطع قسمت به قسمت انجام می‌شود. برای قسمت اول هر کروموزوم، عددی اتفاقی تولید می‌شود و اگر این عدد از احتمال جهش P_{mut} کوچکتر باشد، قطب‌های دلخواه به صورت جدید چیده می‌شوند. قوانین جهش برای قسمت‌های بعد طبق جدول ۸ است.

چیدن قطب‌های مطلوب می‌تواند در دستیابی به بهترین توزیع آنها در زیرمجموعه‌های سطری یا ستونی در الگوریتم تکرارگر کمک کند. جهش درایه‌های ماتریس بهره، جستجوی سریع ماتریس بهره‌ی مطلوب را به همراه دارد. تغییر مداوم بیت مشخصه‌ی الگوریتم، الگوریتم ژنتیکی را از افتادن در دام نقاط بهینه‌ی محلی می‌رهاند.

به‌طور همزمان - حل شود. اجرای الگوریتم پس از ۴۸ تکرار الگوریتم ژنتیکی، ماتریس زیر را نتیجه می‌دهد:

$$K = \begin{bmatrix} 0.966213894488 & -2/48568.676619 \\ 0.36516322599 & -0.133786.84.34 \end{bmatrix}$$

چنان‌که مشاهده می‌شود این ماتریس، قطب‌های سیستم حلقه‌بسته را در $1/00010827540.847$ ، -0.99992236711536 ، $1/0002618660147$ و $-2/0001394662447$ قرار می‌دهد.

مثال ۲. در این مثال، مسئله‌ی را بررسی می‌کنیم که در آن یکی از قطب‌های سیستم حلقه‌بسته باید منطبق بر یکی از قطب‌های سیستم حلقه‌باز باشد. برای این منظور، سیستم توصیف شده با تحقق زیر را در نظر بگیرید:

$$A = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 \\ 0 & 0 & -3 & 0 \\ 0 & 0 & 0 & -4 \end{bmatrix} \quad B = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 0 \\ 1 & 1 \end{bmatrix}$$

$$C = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{bmatrix}$$

مجموعه قطب‌های مطلوب سیستم حلقه‌بسته عبارتند از: ۱-، ۲-، ۳- و ۵- بعد از ۸ تکرار ماتریس زیر به دست می‌آید:

$$K = \begin{bmatrix} 1/4 & 1/2 \\ -16/2 & -1/6 \end{bmatrix}$$

قطب‌های سیستم حلقه‌بسته با به کارگیری این ماتریس، در نقاط ۱-، ۲-، ۳- و ۵- قرار می‌گیرند.

نتیجه‌گیری

در این نوشتار، روش جدیدی برای حل مسئله قطب‌گماری با بازخور خروجی ارائه شده است. الگوریتم طراحی شده محدودیتی بر مجموعه قطب‌های مورد نظر اعمال نمی‌کند و این مجموعه می‌تواند شامل مقادیر یکسان، یا قطب‌های سیستم حلقه‌باز باشد.

قطب‌گماری با بازخور خروجی، حل دستگامی از معادلات غیرخطی است. در الگوریتم طراحی شده، از طریق تبدیل این مسئله به چند مجموعه از معادلات خطی، جواب جستجو می‌شود. بخش ژنتیکی الگوریتم، بهترین ماتریس‌های بهره‌را از الگوریتم دیگری که الگوریتم تکرارگرا نامیده می‌شود انتخاب و نیز شرایط اولیه‌ی لازم را برای الگوریتم تکرارگرا فراهم می‌کند. آزمایش الگوریتم بر روی سیستم‌های مختلف، کارایی آن را به اثبات رسانده است.

در پایان هر تکرار الگوریتم ژنتیکی، مجموع مربعات خطای ناشی از هر کروموزوم محاسبه می‌شود. در صورتی که این مجموع از یک کمتر باشد، الگوریتم به مرحله‌ی بعدی می‌رود. کروموزوم انتخاب شده «کروموزوم مرجع» نامیده می‌شود.

در آغاز مرحله‌ی بعد، برای تولید جمعیت اولیه، کروموزوم مرجع ۵۰ بار تکرار می‌شود. آنگاه قسمت قطب‌های دلخواه در بخش اول کروموزوم‌ها، با ترتیب جدیدی چیده می‌شود و بیت مشخصه‌ی الگوریتم نیز تغییر می‌کند. تمامی کروموزوم‌ها در جمعیت جدید، مجموع مربعات خطای یکسانی خواهند داشت. اگرچه به علت تغییرات ایجاد شده، اجرای الگوریتم تکرارگرا در مورد هر یک از آنها نتایج مختلفی به همراه خواهد داشت.

مراحل ۲، ۳ و ۴ ساختاری مشابه ساختار مرحله‌ی ۱ دارند. در این مراحل، جهش بر قسمت اعشاری درایه‌های ماتریس بهره اثر می‌کند. هر قسمت اعشاری که برای جهش انتخاب شود با عددی اتفاقی بین $0/1$ و $0/1$ جمع می‌شود. الگوریتم مراحل ۲، ۳ و ۴ با یافتن کروموزومی که مجموع مربعات خطای آن کمتر از 10^{-2} ، 10^{-4} و 10^{-6} باشد پایان می‌پذیرد.

مثال‌های عددی

مثال‌هایی که در این بخش انتخاب شده‌اند سیستم‌هایی را معرفی می‌کنند که الگوریتم‌های دیگر ناتوان از یافتن ماتریس بازخور خروجی مناسب برای قطب‌گماری دلخواه بوده‌اند. این مثال‌ها و مثال‌های متعدد دیگر نشان می‌دهد که هیچ‌گونه محدودیتی برای مجموعه قطب‌های مطلوب سیستم حلقه‌بسته وجود ندارد و این مجموعه می‌تواند شامل مقادیر یکسان و یا قطب‌های سیستم حلقه‌باز باشد.

مثال ۱. سیستم توصیف شده با تحقق زیر را در نظر بگیرید:

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 \\ -1 & 0 & 2 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & -3 \end{bmatrix} \quad B = \begin{bmatrix} 0 & -1 \\ 1 & 0 \\ 0 & 3 \\ 0 & 1 \end{bmatrix}$$

$$C = \begin{bmatrix} 1 & -1 & 0 & 0 \\ 0 & 0 & 1 & 2 \end{bmatrix}$$

مرجع شماره ۹، این مثال را با قرار دادن ۲ قطب در ۲ و ۱- حل، و سپس بهترین مکان ممکن برای ۲ قطب دیگر را جستجو کرده است. یعنی الگوریتم ۹ توانسته به‌طور همزمان ۴ قطب را تعیین کند. با استفاده از الگوریتم طراحی شده، ۲ قطب دیگر نیز در ۱- قرار داده می‌شوند تا مسئله در حالت دشوار - قرار دادن سه قطب در ۱-

1. Davison, E.J., "On pole assignment in linear systems with incomplete state feedback," *IEEE Trans. Automatic Control*, pp. 348-351 (June 1970).
2. Davison, E.J., Chatterjee, "A note on pole assignment in linear systems with incomplete state feedback," *IEEE Trans. Automatic Control*, pp. 98-99 (February 1971).
3. Sridhar, B. and Lindorff, D.P., "Pole placement with constant gain output feedback," *INT. J. Control*, **18**(5), pp. 993-1003 (1973).
4. Kimura, H. "Pole assignment by gain output feedback," *IEEE Trans. Automatic Control*, pp. 516-519 (August 1975).
5. Davison, E.J. and Wang, S.H., "On pole assignment in linear multivariable systems using output feedback," *IEEE Trans. Automatic Control*, pp. 509-515 (August 1975).
6. Kimura, H., "A further result on the problem of pole assignment by output feedback," *IEEE Trans. Automatic Control*, pp. 458-463 (June 1977).
7. Fahmy, M.M. and O'Reilly, J., "Multistage parametric eigenstructure assignment by output feedback control," *INT. J. Control*, **48**(1), pp. 97-116 (1988).
8. Fahmy, M.M. and O'Reilly, J., "Parametric eigenstructure assignment by output feedback control; the case of multiple eigenvalues," *INT. J. Control*, **48**(4), pp. 1519-1535 (1988).
9. Roppenecker, G. and O'Reilly, J., "Parametric output feedback controller design," *Automatica*, **25**(2), pp. 259-265 (1989).
10. Seraji, H., "A new method for pole placement using output feedback," *INT. J. Control*, **28**(1), pp. 147-155 (1979).
11. Calvo-Ramon, J.R., "Eigenstructure assignment by output feedback and residue analysis," *IEEE Trans. Automatic Control*, **AC-31**(3), pp. 247-249 (March 1986).
12. Kwon, B.H. and Youn, M.J., "Eigenvalue-Generalized eigenvector assignment by output feedback," *IEEE Trans. Automatic Control*, **AC-32**(5), pp. 417-421 (March 1987).
13. Duan, G.R., "Solutions of the equation $AV+BW=VF$ and their application to eigenstructure assignment in linear systems," *IEEE Trans. Automatic Control*, **38**(2), pp. 276-280 (February 1993).
14. Syrmos, V.L. and Lewis, L., "Output feedback eigenstructure assignment using two Sylvester equations," *IEEE Trans. Automatic Control*, **38**(3), pp. 495-499 (March 1993).
15. Alexandridis, A.T. and Paaraskevopoulos, P.N., "A new approach to eigenstructure by output feedback," *IEEE Trans. Automatic Control*, **41**(7), pp. 1046-1050 (July 1996).
16. Byrnes, C.I., Anderson, B.D., "Output feedback and generic stabilizability," *SIAM J. Contr. Optim.*, **22**, pp. 362-380 (May 1984).
17. Tarokh, M. and "Approach to pole assignment by centralised and decentralised output feedback," *IEE Proc., D*, **136**, (March 1989).
18. Herman, R. and Martin, C.I., "Application of algebraic geometry to systems theory: Part I," *IEEE Trans. Automatic Control*, **AC-22**, pp. 19-25 (1977).
19. Giannakopoulos, C. and Karcianas, N., "Pole assignment of strictly proper and proper systems by constant output feedback," *INT. J. Control*, pp. 543-565 (1985).
20. Lee, T.H., Wang O.G. and Koh, E.K., "An iterative algorithm for pole placement by output feedback," *IEEE Trans. Automatic Control*, **39**(3), pp. 565-568 (March 1994).
21. Goldberg, D.E., "Genetic algorithms and rule learning in dynamic system control," in *Proc. 1st Int. Conf. on Genetic Algorithms and Their Applications*, Hillsdale, NJ: Lawrence Erlbaum, pp. 8-15 (1985).
22. Goldberg, D.E., *Genetic algorithms in search, optimization and machine learning*, Reading, MA: Addison-Wesley, (1989).
23. Davis, L., Ed., *Handbook of Genetic Algorithms*, New York: Van Nostrand Reinhold, (1991).