بررسی و بهینه سازی تقویت کننده قدرت با استفاده از ژنتیک الگوریتم

هادی آریاکیا(1) ، مسعود مصدق(1) ، ابراهیمی راد (2)

(1) دانشگاه آزاد اسلامی واحد تهران مرکز (2) دانشگاه تهران دانشکده برق و کامپیوتر

چکیده - الگوریتم‌های ژنتیک از اصول انتخاب طبیعی داروین برای یافتن فرمول بهینه جهت پیش‌بینی یا تطبیق الگو استفاده می‌کند. الگوریتم‌های ژنتیک اغلب گزینه خوبی برای تکنیک‌های پیش‌بینی برمبنای رگرسیون هستند. مختصراً گفته می‌شود که الگوریتم ژنتیک (یا *GA*) یک تکنیک برنامه‌نویسی است که از تکامل ژنتیکی به عنوان یک الگوی حل مسئله استفاده می‌کند. در این مقاله سعی بر آن داریم تا یک تقویت کننده قدرت را بررسی نموده و با استفاده از ژنتیک الگوریتم پارامترهای بهینه را برای مدار یافته و اقدام به بررسی نتایج بدست آمده نماییم

کلید واژه- ژنتیک الگوریتم ، تقویت کننده قدرت ، معایب ژنتیک الگوریتم ، معایب ژنتیک الگوریتم

1- مقدمه

قانون انتخاب طبیعی بدین صورت است که تنها گونه‌هایی از یک جمعیت ادامه نسل می‌دهند که بهترین خصوصیات را داشته باشند و آنهایی که این خصوصیات را نداشته باشند به تدریج و در طی زمان از بین می‌روند. البته درست‌تر آنست که بگوییم طبیعت مناسب ترین‌ها را انتخاب می‌کند نه بهترین‌ها. گاهی در طبیعت گونه‌های متکامل‌تری به وجود می‌آیند که نمی‌توان گفت صرفا حاصل تکامل تدریجی گونه قبلی هستند. آنچه که ممکن است تا حدی علت این رویداد را توضیح دهد مفهومیست به نام : تصادف یا جهش. بر اساس این ویژگی‌ها ژنتیک الگوریتم سعی بر آن داریم تا یک تقویت کننده قدرت را مورد بررسی و بهینه سازی قرار دهیم. لذا در ابتدا ساختار یک تقویت کننده قدرت را بررسی و با توجه به نیازها کامل و سپس مقادیر برخی از پارامترهای آن را با ژنتیک الگوریتم بدست آورده و در نهایت با استفاده از مقادیر بدست آمده و اعمال آن مقادیر به مدل ، نتایج را بررسی می‌نماییم لذا این مقاله را به 3 بخش کلی تقسیم می‌کنیم :

بخش اول : توضیحی بر ژنتیک الگوریتم

بخش دوم : بررسی یک تقویت کننده قدرت و بدست آوردن تابع تبدیل سیستم

بخش سوم : بررسی و تحلیل پارامترهای مورد نظر توسط ژنتیک الگوریتم

2- توضیحی بر ژنتیک الگوریتم

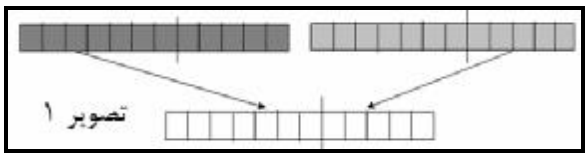
الگوریتم ژنتیک GA یک تکنیک جستجو در علم کامپیوتربرای یافتن راه حل بهینه و مسائل جستجو است.الگوریتم های ژنتیک یکی از انواع الگوریتم های تکاملی اند که از علم زیست شناسی مثل وراثت، جهش،انتخاب ناگهانی ، انتخاب طبیعی و ترکیب الهام گرفته شده .[2]

در دهه هفتاد میلادی دانشمندی به نام جان هلند ایده استفاده از الگوریتم ژنتیک را در بهینه‌سازی‌های مهندسی مطرح کرد. ایده اساسی این الگوریتم انتقال خصوصیات موروثی توسط ژن‌هاست. فرض کنید مجموعه خصوصیات انسان توسط کروموزوم‌های اول به نسل بعدی منتقل می‌شوند. هر ژن در این کروموزوم‌ها نماینده یک خصوصیت است که بصورت همزمان دو اتفاق برای کروموزوم‌ها می‌افتد. اتفاق اول موتاسیون (Mutation) است. موتاسیون به این صورت است که بعضی ژن‌ها بصورت کاملا تصادفی تغییر می‌کنند(البته تعداد این گونه ژن‌ها بسیار کم می‌باشد). اتفاق دیگر چسبیدن ابتدای یک کروموزوم به انتهای یک کروموزوم دیگر است؛ این مساله با نام Crossover شناخته می‌شود(البته این اتفاق به تعداد بسیار بیشتری نسبت به موتاسیون رخ می‌دهد). این همان چیزیست که مثلا باعث می‌شود تا فرزند تعدادی از خصوصیات پدر و تعدادی از خصوصیات مادر را با هم به ارث ببرد و از شبیه شدن تام فرزند به تنها یکی از والدین جلوگیری می‌کند.[1]

در ابتدا تعداد مشخصی از ورودی ها،X1,X2,…,Xn که متعلق به فضای نمونه X هستند را انتخاب می کنیم و آنها را در یک عدد برداری X=(x1,x2,…xn) نمایش می‌دهیم ؛ در مهندسی نرم افزار اصطلاحاً به آنها ارگانیسم یا کروموزوم گفته می شود و به گروه کروموزوم ها Colony یا جمعیت می گوییم،در هر دوره ، Colony رشد می کند و بر اساس قوانین مشخصی که حاکی از تکامل زیستی است تکامل می یابد.

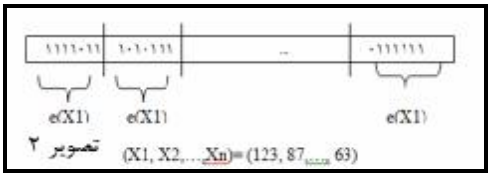
برای هر کروموزوم Xi ،ما یک ارزش تناسب (Fitness) داریم‌که آن راf(Xi) هم می‌نامیم،عناصر قویتر یا کروموزوم هایی که ارزش تناسب آنها به بهینه Colony نزدیکتر است شانس بیشتری برای زنده ماندن در طول دوره های دیگر و دوباره تولید شدن را دارند و ضعیفترها محکوم به نابودی اند. به عبارت دیگر الگوریتم ورودی هایی که به جواب بهینه نزدیکترندرانگه داشته واز بقیه صرف نظر می کند.

یک گام مهم دیگر درالگوریتم،تولد است که در هر دوره یکبار اتفاق می افتد. محتویات دو کروموزومی که در فرآیند تولید شرکت می کنند با هم ترکیب میشوند تا 2 کروموزوم جدید که ما انها را فرزند می نامیم ایجاد کنند.این هیوریستیک به ما اجازه می دهد تا 2 تا از بهترین ها را برای ایجاد یکی بهتر از آنها با هم ترکیب کنیم.مانند آنچه در شکل 1 مشخص می‌باشد. (evolution)به علاوه در طول هر دوره،یک سری از کروموزوم ها ممکن است جهش یابند(Mutation)[4] .



هر ورودی x در یک عدد برداری $X=(x_1,x_2,..,x_n)$ قرار دارد .برای اجرای الگوریتم ژنتیک مان باید هر ورودی را به یک کروموزوم تبدیل کنیم.می توانیم این را با داشتن log(n) بیت برای هر عنصرو تبدیل ارزش Xi انجام دهیم

مثل شکل 2



می توانیم از هر روش کد کردن برای اعداد استفاده کنیم.در دوره 0، یک دسته از ورودی های X را به صورت تصادفی انتخاب می کنیم. بعد برای هر دوره iام ما ارزش مقدار Fitness را تولید،تغییر وانتخاب را اعمال می کنیم.الگوریتم وقتی پایان می یابد که به معیارمان برسیم.

عموماً راه حلها به صورت 2 تایی 0و1 نشان داده می شوند ولی روشهای نمایش دیگری هم وجود دارد.تکامل از یک مجموعه کاملاً تصادفی از موجودیت ها شروع می شود و در نسلهای بعدی تکرار می شود.در هر نسل،مناسبترین ها و نه بهترین ها انتخاب می شوند.

روش های مختلفی برای الگوریتم های ژنتیک وجود دارند که می توان برای انتخاب ژنوم ها از آنها استفاده کرد.اما روش های لیست شده در پایین از معمولترین روش ها هستند.

انتخاب Elitist :مناسبترین عضو هر اجتماع انتخاب می شود.

انتخاب Roulette : یک روش انتخاب است که در آن عنصری که عدد برازش(تناسب)بیشتری داشته باشد،انتخاب می شود.

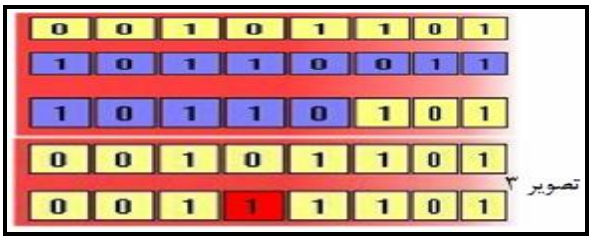
انتخاب Scaling :به موازات افزایش متوسط عدد برازش جامعه،سنگینی انتخاب هم بیشتر می شودوجزئی تر.این روش وقتی کاربرد دارد که مجموعه دارای عناصری باشد که عدد برازش بزرگی دارند وفقط تفاوت های کوچکی آنها را از هم تفکیک می کند.

انتخاب Tournament : یک زیر مجموعه از صفات یک جامعه انتخاب می شوندواعضای آن مجموعه با هم رقابت می کنندو سرانجام فقط یک صفت از هر زیر گروه برای تولید انتخاب می شوند.

بعضی از روشهای دیگر عبارتند از: Rank Selection, Generational Selection, Steady-State Selection .Hierarchical Selection

در روش Crossover 2 کروموزوم برای معاوضه سگمنتهای کدشان انتخاب می شوند.این فرآیند بر اساس فرآیند ترکیب کروموزوم ها در طول تولید مثل در موجودات زنده شبیه سازی شده. اغلب روش های معمول

Crossover شامل Single-point Crossoverهستند ، که نقطه تعویض در جایی تصادفی بین ژنوم ها است.بخش اول قبل از نقطه ،و بخش دوم سگمنت بعد از آن ادامه پیدا می کند،که هر قسمت برگرفته از یک والد است،که 50/50 انتخاب شده.



تصویر ۳

تصویر 3 تاثیر هر یک از عملگر های ژنتیک را روی کروموزوم های 8 بیتی نشان می دهد.ردیف بالاتر 2 ژنوم را نشان می دهد که نقطه تعویض بین 5امین و 6امین مکان در ژنوم قرار گرفته،ایجاد یک ژنوم جدید از پیوند این 2 والد بدست می آیند.شکل 2وم ژنومی را نشان می دهد که دچار جهش شده و 0 در آن مکان به 1 تبدیل شده .

3- بررسی یک تقویت کننده قدرت و بدست آوردن تابع تبدیل سیستم :

تقویت کننده‌ای که در ابتدا می‌خواهیم به بررسی آن بپردازیم، دارای شمای کلی بصورت شکل 4 می‌باشد. البته در انتهای این بخش یک کنترل کننده پیش فاز در مسیر فیدبک استفاده خواهیم کرد.با پیش فرض‌ها زیر کار تحیل و بدست آورد تابع تبدیل سیستم را شروع می‌کنیم :

مشخصات تقویت کننده عملیاتی :

بهره حلقه باز : $A1=10^6$

فرکانس قطع پایین : $f1=10Hz$

فرکانس قطع بالا : $f2=1MHz$

افت ولتاژ : 2.5 V

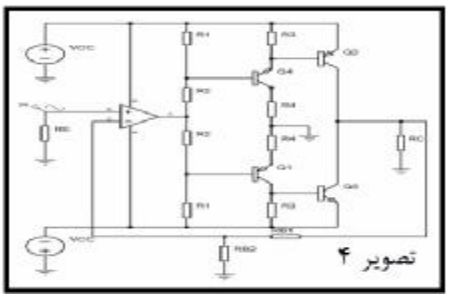
مشخصات تقویت کنندۀ ترانزیستوری :

بهره حلقه باز : $A1=10$

فرکانس قطع بالا : $f2=10KHz$

افت ولتاژ : 1.5 V

توابع تبدیل حلقه باز و حلقه بسته در تصویر 5 مشخص می باشند

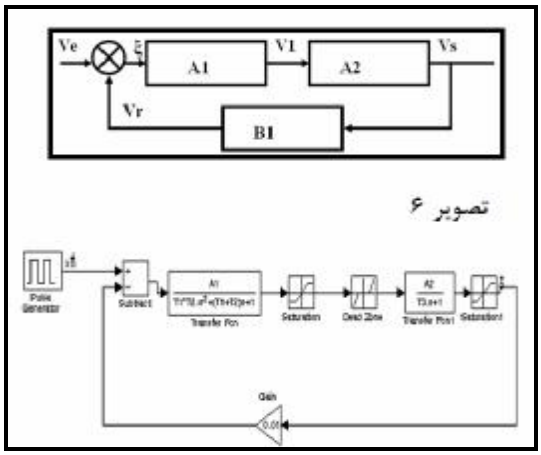

تصویر ۴

$$OLTF = Vr/\xi = A_1A_2A_3$$

$$CLTF = Vs/Ve = \frac{A_1A_2}{1+A_1A_2B_1}$$

تصویر ۵

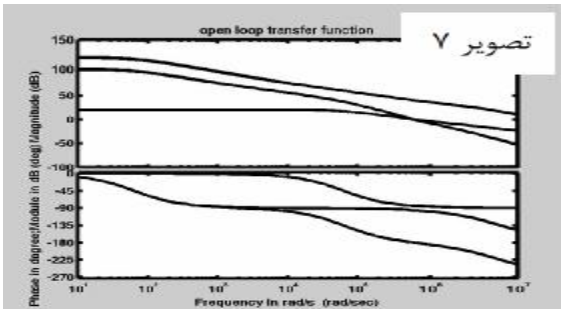
مدل سیمولینک تقویت کننده بصورت شکل 6 می‌باشد :



تصویر ۶

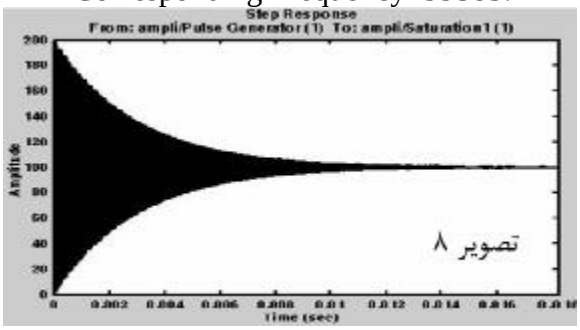
که در آن $T_1=1/2\prod f_1$ $T_2=1/2\prod f_2$ $T_3=1/2\prod f_3$ $B_1=0.01$

با وارد کردن تابع تبدیل نمودار بود و مارجین را برای این مدار حساب می‌نماییم که در تصویر 7 و 8 رسم شده است. برنامه m فایل مربوطه با نام ampli1.m در ضمیمه موجود این مقاله موجود می‌باشد.



تصویر ۷

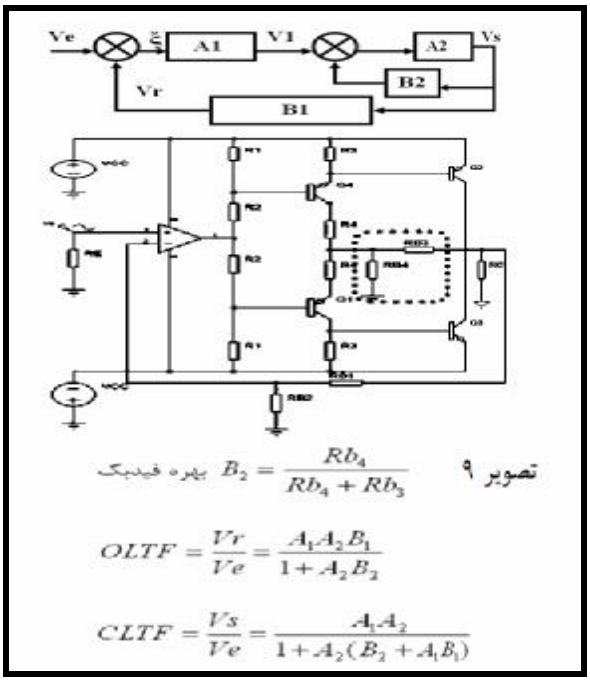
Gain margin in dB=1.011
Corresponding Frequency=100147.9888
Phase margin n degrees=0.064197
Corresponding Frequency=99583.2111



تصویر ۸

همانطور که مشاهده می‌فرمایید حاشیه بهره و فرکانس بر هم افتادگی بهره به ترتیب 1.011 و 100KHz و همچنین حاشیه فاز و فرکانس برهم افتادگی فاز 99KHz می‌باشد و سیستم در آستانه ناپایداری است.

اینبار سعی میکنیم با استفاده از فیدبک در بخش تقویت کننده ترانزیستوری پایداری را افزایش دهیم. شکل مدار ، نمودار بلوکی و مدل سیمولینک مدار را در تصویر 9 می‌توانید مشاهده نمایید



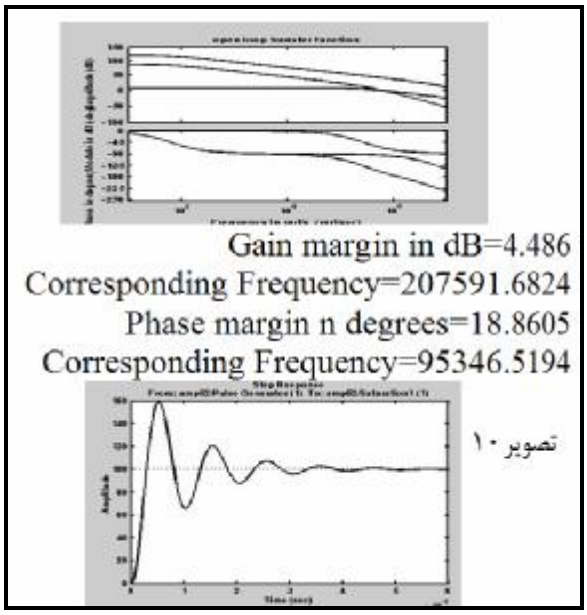
تصویر ۹

بهره فیدبک $B_2 = \frac{Rb_4}{Rb_4 + Rb_3}$

$$OLTF = \frac{Vr}{Ve} = \frac{A_1 A_2 B_1}{1 + A_2 B_2}$$

$$CLTF = \frac{Vs}{Ve} = \frac{A_1 A_2}{1 + A_2 (B_2 + A_1 B_1)}$$

برای بدست آوردن تابع تبدیل کلی سیستم و نمودارهای بود از فایل Ampli2.m استفاده شده است که در بخش ضمیمه موجود می‌باشد. نتایج بدست آمده و نمودارهای بود و پاسخ پله را می‌توانید در تصویر 10 مشاهده نمایید



تصویر ۱۰

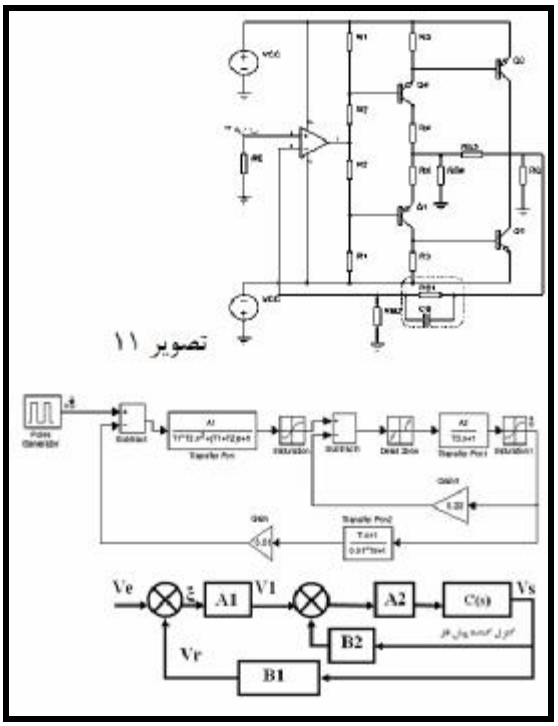
همانطور که از نمودارتصویر 10 و نتایج بدست آمده از قبل مشخص می‌باشد، حاشیه فاز و فرکانس بر هم افتادگی بهره از 0.06درجه و 99.6KHz به 18.9 درجه و 95.3 KHz رسیده است. هر چند که این بهبود مطلوب ما می‌بود اما بازهم مقادیر قابل اطمینانی برای یک تقویت کننده نمی‌باشد

راه دیگری که معمولا برای افزایش فاز سیستم‌ها مورد استفاده قرار می‌گیرد، استفاده از کنترل کننده پس فاز است. اما از آنجا که کنترل کننده پس فاز تاخیری را بر سیستم تحمیل می‌کند، پاسخ سیستم را کند می‌نماید که باعث ایجاد نوسانهای شدید در خروجی تقویت کننده می‌شود.

لذا یک کنترل کننده پیش‌فاز استفاده گردید و چون کنترل کننده پیش فاز رفتاری شبیه مشتق‌گیر دارد بهره و سرعت پاسخ سیستم را افزایش‌می‌دهد. در شکل 11 مدار با فیدبک پیش‌فاز و همچنین مدل سیمولینک آنرا مشاهده می‌نمایید

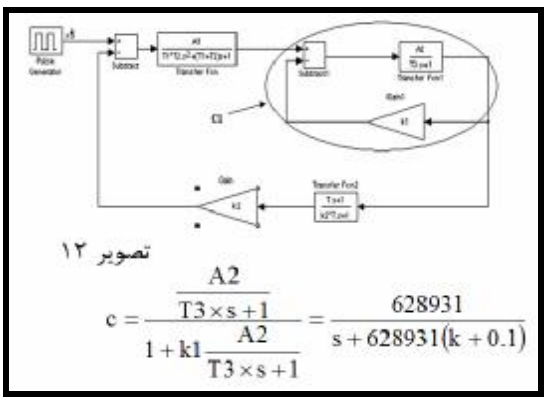
4- بررسی و تحلیل پارامترهای مورد نظر توسط ژنتیک الگوریتم

برای پیاده سازی این amplifire در برنامه genetic algorithms در مطلب ابتدا تابع تبدیل کل را محاسبه می‌کنیم



تصویر ۱۱

همان‌جور که در تصویر 12مشاهده می‌نمایید برای محاسبه ابتدا باید تابع تبدیل داخلی‌ترین لوپ را محاسبه کنیم برای محاسبه فیدبک منفی C1 از فرمول $\frac{G1}{1+G1G2}$ بدست می‌آید

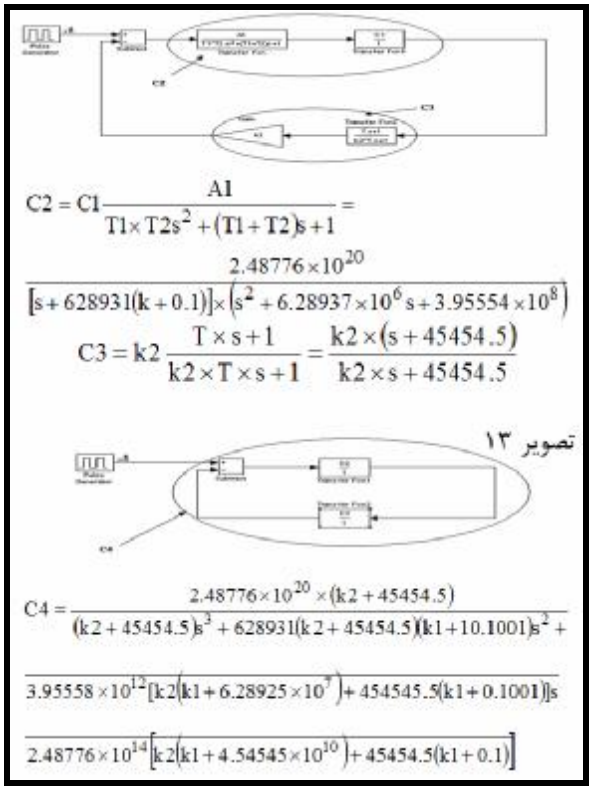


تصویر ۱۲

$$c=\frac{\frac{A2}{T3\times s+1}}{1+k1\frac{A2}{T3\times s+1}}=\frac{628931}{s+628931(k+0.1)}$$

که در آن A1=10 , T3=1.59*10-5 می‌باشد و k1 مقدار گین ماست که مجهول است و میخواهیم از طریق Genetic_Algorithms مقدار بهینه آن را بدست آوریم.

مطابق تصویر 13 مقدار تابع تبدیل‌های C2 و C3 را که در آن $A1=1*10^{6}$, T1= 0.0159 , T2 = $1.59*10^{-7}$, $T=2.2*10^{-5}$ می‌باشد را محاسبه می‌کنیم:



$$C2=C1\frac{A1}{T1\times T2s^{2}+(T1+T2)s+1}=$$

$$\frac{2.48776\times10^{20}}{[s+628931(k+0.1)]\times(s^{2}+6.28937\times10^{6}s+3.95554\times10^{8})}$$

$$C3=k2\frac{T\times s+1}{k2\times T\times s+1}=\frac{k2\times(s+45454.5)}{k2\times s+45454.5}$$

تصویر ۱۳

$$C4=\frac{2.48776\times10^{20}\times(k2+45454.5)}{(k2+45454.5)s^{3}+628931(k2+45454.5)(k1+10.1001)s^{2}+}$$

$$\frac{}{3.95558\times10^{12}[k2(k1+6.28925\times10^{7})+454545.5(k1+0.1001)]s}$$

$$\frac{}{2.48776\times10^{14}[k2(k1+4.54545\times10^{10})+45454.5(k1+0.1)]}$$

C4 تابع تبدیل کلی است صورت و خرج آن را به صورت ماتریس لاپلاس با نامهای num و den تعرف کرده و در peacksfcn.m در Genetic Algorithm استفاده می‌کنیم در این جا ما دو مجهول k1 و K2 داریم که میخواهیم از طریق برنامه Genetic Algorithm بهترین مقدار برای این گین‌ها بدست آوریم peachsfcn.m را به این‌شکل تعریف می‌کنیم :

```
k1 = input(1); k2 = input(2);
z=0;
t=0:0.01:2;
num=[2.48776e20*(k2+45454.5)];
den=[k2+45454.5,628931*(k2+45454.5)*(k
1+10.1001),3.95558e12*(k2*(k1+6.28925e
7)+45454.5*(k1+0.1001)),2.48776e14*

(k2*(k1+4.54545e10)+45454.5*(k1+0.1))];
y=step(num,den,t);
sy=size(y);
for i=1:sy
   z=z+(y(i)-(100))^-2;
end
```

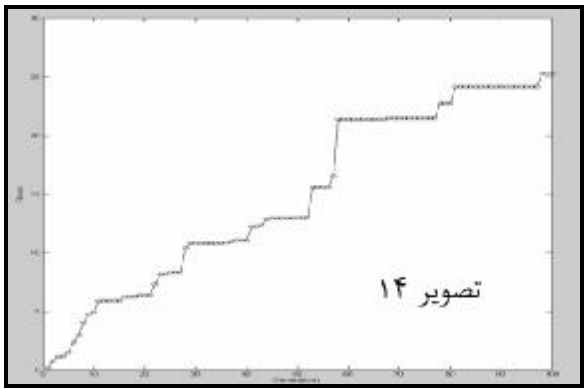
برای بدست آوردن مقدار بهینه باید خطای ما کمترین مقدار را داشته باشد برای بدست آوردن خطا از فرمول

$\sum_{i}\left(y_{out}[i]\_y_e\right)^2$ استفاده میکنیم که همان Fitness Function ما میباشد ، y_e مقدار بهینه‌ای است که خروجی باید به آن برسد.

تعداد نسل‌ها و جمعیت و ژن‌ها را به‌ترتیب 100،100 و 32 قرار میدهیم تا Gnetic Algorithm در بهترین حالت مقدار بهینه گین‌های k1 و k2 را بدست آورد وهمچنین تعداد متغیرهایمان را با مقدار دهی var_n=2 در Gnetic Algorithm تعیین می‌کنیم

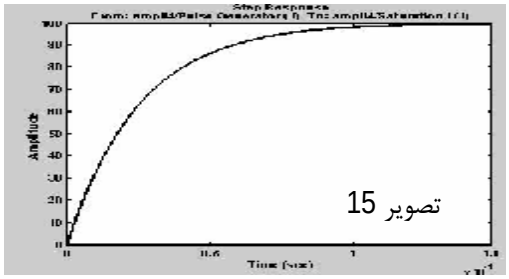
وهمچنین در clab.m محدوده متغیرهای خودمان را تعریف میکنیم‌که در اینجا ما هردو متغیر را از 0,0001 تا 2 تعریف می‌کنیم

حال در clab با زدن کلید F5 برنامه را اجرا میکنیم (Run) و مقدار بهینه k1 و k2 را از طریق این برنامه بدست می‌آوریم که نتیجه را در تصویر 14 می‌توانید مشاهده نمایید.



در نسل صدم مقدار K1=1.994361 و K2=0.009998بدست‌آمد.

ازمنوی..Tools/ControlDesign/LinearAnalysis را انتخابکرده و اجرا می‌کنیم همان‌طورکه در شکل زیر می‌بینید خروجی به مقدار بهینه خو یعنی 100 در مدت 1.5*10-4sec رسیده است پس از طریق Gnetic_Algorithm با بدست آوردن مقدار بهینه گین فیدبک ها در کمترین مدت به بهترین جواب رسیده همچنین گین کلی مدار هم برابر 100 گردیده است.



5- نتیجه‌گیری

از محاسن ذاتی ژنتیک الگوریتم موازی بودن آن می‌باشد . اکثر الگوریتم های دیگر موازی نیستند و فقط می توانند فضای مسئله مورد نظر را در یک جهت و یک لحظه جستجو کنند و جواب پیدا شده ممکن است جواب بهینه محلی باشد و یا زیر مجموعه ای از جواب اصلی باشد. حسن دیگر ژنتیک الگوریتم که در اینجا از آن سود جستیم امکان تغییر چندین پارامتر بشکل همزمان بود. در مسائل واقعی نمی‌توان محدود به یک ویژگی شد تا آن ویژگی ماکسیمم شود یا مینیمم و باید چند جنبه در نظر گرفته شود و به همین خاطر از طریق ژنتیک الگوریتم توانستیم به پاسخ مطلوب دست پیدا کرده علاوه رسیدن به پاسخ پله مناسب که نمودار آنرا در تصویر 15 مشاهده می‌نمایید به گین 100 که مطلوبمان بود دست پیدا نماییم. ناگفته نماند ژنتیک الگوریتم معایبی هم دارد یک مشکل چگونگی نوشتن عملگر Fitness است که منجر به بهترین راه حل برای مسئله شود. مشکل دیگر ،که آن را نارس می نامیم این است که اگر یک ژنوم که فاصله اش با سایر ژنوم های نسل اش زیاد باشد(خیلی بهتر از بقیه باشد) و خیلی زود ایجاد ممکن است محدودیت ایجاد کند و راه حل را به سوی جواب بهینه محلی سوق دهد. متاسفانه در اینصورت تمام زمانی که برای محاسبات صرف شده بیهوده بوده و محاسبات می‌بایستی از ابتدا شروع گردد.

6- مراجع

[1] مجله علم و کامپیوتر www.ccwmagazine.com

[2] www.wikipedia.com

[3] www.talkorigins.org

[4] دانشکده کامپیوتر دانشگاه McGill کانادا

university of Nottingham www.cs.nott.ac.ukIT

www.gpwiki.org

ENGINEERING APPLICATION OF MATLAB AND SIMULINK – MICHEL MARIE - MOHAMMAD MOKHARI

www.itna.com

www.smi.stanford.edu/people/koza

www.cgm.cs.mcgill.ca
www.sharifthinktank.com
Guided operators for a hipper-heuristic Genetic Algorithm
en.wikipedia.org/wiki/Genetic_algorithm
www.rennard.org/alife/english/gavintrgb.html
www.aaai.org/AITopics/html/genalg.html
www.aimlearning.com
www.**genetic**programming.com
www.talkorigins.org/faqs/genalg/genalg.html