دسترسی در سایتِ http://jnrm.srbiau.ac.ir
سال ششم، شماره بیست و هفتم، آذر و دی ۱۳۹۹
شماره شاپا: ۵۸۸-۲۵۸۸X

پژوهش‌های نوین در ریاضی

دانشگاه آزاد اسلامی، واحد علوم و تحقیقات

# جواب تحلیلی معادلات دیفرانسیل مبتنی بر روش فرا ابتکاری ترکیبی الگوریتم‌های ژنتیک و بهینه‌سازی کلونی مورچگان

**ناصر میکائیل وند*[۱]، اکرم جوادی[۲]، حسن حسین‌زاده[۳]**

(۱و۲و۳) گروه ریاضی، واحد اردبیل، دانشگاه آزاد اسلامی، اردبیل، ایران

تاریخ ارسال مقاله: ۹۷/۱۲/۱۳ تاریخ پذیرش مقاله: ۹۸/۰۸/۰۳

**چکیده**

هدف اصلی این تحقیق، بررسی قابلیت استفاده از الگوریتم ترکیبی ژنتیک–کلونی مورچگان با رویکرد تولید جواب‌های آزمایشی و بهبود آنها برای تولید جواب تحلیلی– عددی انواع مختلفی از معادلات دیفرانسیل معمولی و جزئی است. الگوریتم بهینه‌سازی کلونی مورچگان[۱] (ACO) یک الگوریتم مناسب دارای دقت و سرعت همگرایی بالا برای یافتن جوابهای تقریبی برای حل مسائل بهینه‌سازی با استفاده از تابع احتمال وابسته به میزان اثر باقیمانده از حرکت مورچه‌هاست. الگوریتم ژنتیک[۲] نیز یک روش بهینه‌سازی مبتنی بر اپراتورهای جهش و تقاطع است که دارای منطقه جستجوی گسترده‌ای است که مانع از به تله افتادن الگوریتم در جواب محلی می‌شود. ترکیبی از این دو الگوریتم، یک الگوریتم با حداکثر کارایی را ایجاد می‌کند. بررسی مثال‌های گوناگون در بخش پایانی مقاله سرعت و دقت بالای روش پیشنهادی را نمایش می‌دهد.

**واژه‌های کلیدی:** معادلات دیفرانسیل معمولی، معادلات دیفرانسیل با مشتقات جزئی، برنامه‌ریزی ترکیبی مورچگان و ژنتیک، مینیمم‌سازی خطای وابسته.

*. عهده‌دار مکاتبات: nassermikaeilvand@gmail.com

1. Ant Colony Optimization
2. Genetic Algorithm

## مقدمه

بسیاری از مسائل در زمینه‌های مختلف علوم کاربردی نظیر فیزیک، شیمی و اقتصاد که مربوط به بررسی تغییرات یک یا چند متغیر می‌شوند؛ توسط معادلات دیفرانسیل بیان می‌شوند. پیش‌بینی وضع آب و هوا، مکانیک کوانتومی، انتشار موج و دینامیک بازار سهام برخی از این نمونه‌هاست که حل سریع و دقیق آنها تاثیرات شگرفی در زندگی انسان‌ها باقی می‌گذارد و به همین دلیل روش‌های متعددی برای حل معادلات دیفرانسیل پیشنهاد شده است [۷-۱]. همچنین می‌توان به حل با روش‌های دیگر مانند ماتریس عملیاتی چند جمله‌ای‌های زرنیکه شعاعی، وجود جواب‌های یک معادله دیفرانسیل کسری جدید p-لاپلاسین با اثر ضربه‌ای، استفاده از توابع بی-اسپلاین مکعبی و توابع پایه شعاعی با پارامتر شکلی متغیر اشاره کرد [۲۱-۱۷].

با افزایش بیشتر توانایی کامپیوترها در انجام سریع‌تر و دقیق‌تر محاسبات عددی ارائه روشهایی بر مبنای استفاده از این قابلیت به یکی از مهمترین موضوعات مورد بررسی محققان تبدیل شد. روش‌های نیمه-تحلیلی برای حل معادلات دیفرانسیل، رویکرد تبدیل معادلات دیفرانسیل به دستگاه معادلات جبری، روش عناصر متناهی، روش‌های طیفی و شبکه‌های عصبی مصنوعی از جمله این روش‌هاست که در دهه‌های اخیر مورد توجه زیادی قرار گرفته است [۱۰-۸].

یک روش غیرمعمول و غیر سنتی برای حل معادلات دیفرانسیل بر پایه تبدیل این معادلات به مساله مینیمم‌سازی خطا [۱۱] و حل آن با استفاده از الگوریتم‌های بهینه‌سازی به خصوص الگوریتم‌های فرا ابتکاری می‌باشد. برنامه نویسی ژنتیک (GP) یکی از روش‌هایی است که طی آن یک فرآیند بهینه‌سازی مبتنی بر تعداد زیادی از جواب‌های ممکن تولید شده- نسل اولیه- از طریق عملیات‌های ژنتیکی مانند جهش، تقاطع و تکرار ساخته و بهبود می‌یابند. در این تکنیک نسل‌های جواب‌های آزمایشی به فرمی تحلیلی بیان می‌شوند. در بیشتر موارد، جواب ارائه شده دقیق است اما مواردی وجود دارد که جواب بدست آمده به صورت یک جواب تقریبی با میزان دقت معلوم ایجاد می‌شود [۱۲و۱۳]. علاوه بر این، یک روش برای حل معادلات دیفرانسیل براساس الگوریتم کلونی مورچگان با تکنیکی مشابه الگوریتم ژنتیک در [۱۴] ارائه شده و مثال‌های گوناگونی برای بررسی توانایی روش در مسائل گوناگون ارائه شده و نتایج با روش‌های سنتی مورد مقایسه قرار گرفته است.

در این مقاله یک روش جدید برای حل انواع مختلفی از معادلات دیفرانسیل معمولی و جزئی بر پایه برنامه‌نویسی ترکیبی ژنتیک و کلونی مورچگان (GA-ACO) ارائه شده است. در رویکرد ارائه شده نسلی از جواب‌های آزمایشی تولید کرده و سعی کرده‌ایم تا مقدار تابع خطا را کمینه نماییم. مزیت این روش آن است که می‌تواند جواب‌های آزمایشی با فرم تابعی به شدت تطبیق‌پذیر تولید کند.

در ادامه و در بخش‌های بعدی توضیحاتی در زمینه الگوریتم‌های بهینه‌سازی ژنتیک و کلونی مورچگان ارائه می‌گردد و نحوه ترکیب این دو الگوریتم توضیح داده می‌شود و سپس مبانی نظری گرافی که برای تولید جمعیت در این روش استفاده می‌گردد توضیح داده شده؛ تابع برازندگی و شرایط حل مسئله تعیین می‌شوند. در نهایت مثال‌هایی از انواع مختلف معادلات دیفرانسیل توسط الگوریتم پیشنهادی حل شده و نتایج مورد بحث و بررسی قرار می‌گیرد.

## الگوریتم ژنتیک

الگوریتم ژنتیک به عنوان یکی از روش‌های ابتکاری تاکنون قابلیت خود را در حل مسائل پیچیده بهینه‌سازی نشان داده است [۱۶-۱۵]. GA با اغلب روش‌های بهینه سازی و جستجو در موارد زیر دارای اختلاف می‌باشد که باعث برتری آن بر روش‌های دیگر بهینه‌سازی شده است.

- الگوریتم ژنتیک با یک مجموعه از پارامترهای کد شده کار می‌کند.
- الگوریتم ژنتیک جستجو را از یک مجموعه نقاط شروع می‌کند.
- الگوریتم ژنتیک از اطلاعات اصلی (تابع هدف) استفاده می‌نماید نه از اطلاعات کمکی (مشتق، گرادیان، پیوستگی و ...).
- الگوریتم ژنتیک برای جستجو از قوانین احتمال استفاده می‌نماید.

مشکل اصلی الگوریتم ژنتیک علیرغم سادگی پیاده‌سازی، هزینه اجرای آن است. در اغلب موارد حل یک مسئله

نیازمند تولید چندین هزار نسل از کروموزم‌هاست و در نتیجه نیاز به زمان زیادی دارد (خصوصاً اگر تعداد جمعیت اولیه زیاد باشد و یا معادله دیفرانسیل پیچیده باشد.) بطوری‌که گاه برای حل یک مسئله و با یک پردازشگر خانگی بیش از یک هفته برنامه زمان لازم است. طبیعیست که برای حل یک مسئله این زمان زیاد است و همین امر گاهی استفاده از الگوریتم را با مشکل مواجه می‌کند .

در الگوریتم ژنتیک برای تولید جمعیت غالبا از عملگرهای تقاطع، جابجایی و جهش استفاده می‌شود. عملگر تقاطع، عملگر اساسی الگوریتم ژنتیک بوده و در حقیقت به عنوان موتور جستجوی الگوریتم ژنتیک در فضای کاوش عمل می‌نماید. عملگر جابجایی افراد جدیدی تولید می‌کند که از قسمت‌های مختلف ژن والدین خود تشکیل شده‌اند. عملکرد الگوریتم ژنتیک به میزان قابل توجهی به نوع عملگر جابجایی به کار رفته در آن بستگی دارد. عمگلر جهش بعد از تولید نسل جدید با یک احتمال پایین‌تر برای اعمال تغییرات تصادفی در اعضای جمعیت به کار می‌رود. عملگر جهش بر روی ژن‌های کروموزوم‌ها اعمال می‌شود به این ترتیب به ازای هر ژن از کروموزوم یک عدد تصادفی بین ۰ و ۱ انتخاب می‌شود اگر این مقدار کمتر از مقداراز قبل تعیین شده باشد عمل جهش انجام شده و در غیر این‌صورت جهشی صورت نمی‌گیرد .

مراحل اجرای الگوریتم ژنتیک به صورت زیر می‌باشد

۱) ورود اطلاعات مربوط به سیستم و پارامترهای الگوریتم ژنتیک

۲) تولید جمعیت اولیه به صورت تصادفی

۳) محاسبه مقدار تابع برازندگی

۴) انتخاب جمعیت برای انجام تقاطع برای تولید نسل جدید

۵) انتخاب جمعیت برای انجام جهش برای تولید نسل جدید

۶) در صورتی که شرط توقف برقرار نباشد برو به مرحله ۳ در غیر این صورت برو به به مرحله ۷

۷) ارائه بهترین پاسخ ممکن تولیدی توسط الگوریتم ژنتیک

## بهینه‌سازی کلونی مورچه‌ها (ACO)

بهینه‌سازی گروه مورچه‌ها یا ACO یک الگوریتم مناسب برای یافتن جواب‌های تقریبی برای مسائل بهینه‌سازی ترکیباتی است. در این روش، مورچه‌های مصنوعی به وسیله حرکت بر روی نمودار مساله و با باقی گذاشتن نشانه‌هایی بر روی نمودار، همچون مورچه‌های واقعی که در مسیر حرکت خود نشانه‌هایی باقی می‌گذارند؛ باعث می‌شوند که مورچه‌های مصنوعی بعدی بتوانند جواب‌های بهتری را برای مساله فراهم نمایند. همچنین در این روش بر مبنای علم احتمالات می‌توان بهترین مسیر را در یک نمودار یافت.

در دنیای واقعی مورچه‌ها ابتدا به‌طور تصادفی به این سو و آن سو می‌روند تا غذا بیابند. سپس به لانه بر می‌گردند و ردّی از فرومون[1] به جا می‌گذارند. چنین ردی پس از باران به رنگ سفید در می‌آید که قابل رویت است. مورچه‌های دیگر وقتی این مسیر را می‌یابند؛ پرسه زدن را رها کرده و آن را دنبال می‌کنند. اگر به غذا برسند به خانه بر می‌گردند و رد دیگری از خود در کنار رد قبلی می‌گذارند؛ و به عبارتی مسیر قبل را تقویت می‌کنند. فرومون به مرور تبخیر می‌شود که از سه جهت مفید است:

- از آنجا که یک مورچه در یک زمان معین راه‌های کوتاه‌تر را بیشتر پیموده و تقویت می‌کند؛ هر راهی بین خانه و غذا کوتاه‌تر (بهتر) باشد بیشتر تقویت می‌شود و آنکه دورتر است کمتر.
- اگر فرومون اصلاً تبخیر نمی‌شد، مسیرهایی که چند بار طی می‌شدند، چنان بیش از حد جذّاب می‌شدند که جستجوی تصادفی برای غذا را بسیار محدود می‌کردند.
- وقتی غذای انتهای یک مسیر جذاب تمام می‌شود دیگر نباید مورد استفاده قرار گیرد و ردی باقی نماند.

لذا وقتی یک مورچه مسیر کوتاهی (خوبی) را از خانه تا غذا بیابد بقیه مورچه‌ها به احتمال زیادی همان مسیر را دنبال می‌کنند و با تقویت مداوم آن مسیر و تبخیر ردهای دیگر، به مرور همه مورچه‌ها هم مسیرمی‌شوند. هدف الگوریتم کلونی مورچه‌ها تقلید این رفتار توسط مورچه‌های مصنوعی است که روی نمودار در حال حرکت اند. در واقع مساله

1. Pheromone

یافتن کوتاه‌ترین مسیر توسط مورچه‌های مصنوعی حل می‌شود.

برای پیاده‌سازی کلونی مورچه، از مورچه‌های مصنوعی به عنوان عناصری در بهینه‌سازی استفاده می‌شود. البته این عناصر تفاوت‌های اساسی با مورچه‌های واقعی دارند که عبارتند از:

**۱- حافظه:** برای مورچه‌های مصنوعی حافظه‌ایی در نظر گرفته می‌شود که مسیرهای حرکت را در خود نگه می‌دارد.

**۲- موانع ساختگی:** با هدف تغییر دادن جزئیات مسئله برای بررسی الگوریتم و رسیدن به جواب‌های متنوع.

**۳- حیات در محیط گسسته:** مورچه‌های واقعی نمی‌توانند جدا از کلونی به حیات خود ادامه دهند.

مورچه‌ها با اعمال یک سیاست تصمیم‌گیری محلی اتفاقی سفر می‌کنند که از دنباله‌های فرومونی و اطلاعات ابتکاری با هم استفاده می‌کنند. براساس قانون احتمال زیر، هر مورچه از گره $r$ به گره $s$ گراف حرکت می‌کند (شکل ۱):

$$P_{rs}(t) = \frac{\tau_{rs}(t) * [\gamma_s]^{\beta}}{\sum_{i \in J_r^k} [\tau_{ri}(t)] * [\gamma_i]^{\beta}} \qquad (۱)$$

به‌طوری‌که $\gamma_s = \left(\frac{1}{2+\pi_s}\right)^d$ که در آن $\pi_s$ توان نماد $s$ است که یا نماد ترمینال و یا نماد تابع است، $d$ طول جاری عبارت حسابی، $\beta$ پارامتری است که وزن نسبی دنباله فرومون و رویت‌پذیری $(\beta = 1)$ را کنترل می‌کند و $J_k^r$ مجموعه گره‌های ملاقات نشده است.

این مورچه‌ها اطلاعات و داده‌های مربوط به سفرهای خود را در حافظه کاری خود ذخیره می‌کنند. حافظه مورچه در برنامه‌نویسی توسط یک درخت تجزیه نشان داده می‌شود که برای حذف برخی از سفرها که برای جواب همگرا مناسب نیستند استفاده می‌شود. این امر به صرفه‌جویی در زمان محاسبه کمک می‌کند. بعد از تکمیل هر نسل، یک به روز رسانی سراسری دنباله فرمون انجام می‌پذیرد و سطح فرومون به شکل زیر جایگزین می‌شود:

$$\tau_{ij}(t+g) = (1-\rho) * \tau_{ij}(t) + \rho * \frac{1}{L} \qquad (۲)$$

به‌طوری‌که $g$ تعداد نسل‌ها بوده، شاخه‌های $(i.j)$ متعلق به سفر بهینه پیداشده (تا آن لحظه)، $\rho \in [0.1]$ ضریب زوال فرومون، $\tau_y$ مقداردنباله فرومون در شاخه $(i.j)$ و $L$ طول این سفر است. هدف از این عمل، به روز رسانی مقدار فرومون و افزایش مقادیر فرومون در مسیر حل است.

در حالت کلی روند اجرای الگوریتم مورچگان به صورت زیر می‌باشد.

مرحله ۱. تولید یک گراف دارای $l$ گره.

مرحله ۲: به عنوان نقطه آغاز، مقدار یا وزن برابری از فرومون را در هر شاخه از گراف توزیع کن.

مرحله ۳: $k$ مورچه از $k$ نقطه آغازین به گراف وارد شده و براساس قانون احتمال به گره بعدی می‌روند.

مرحله ۴: درخت‌های تجزیه را از روی سفرهای $k$ مورچه بساز.

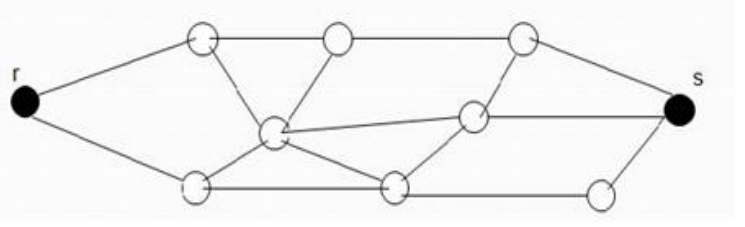
مرحله ۵: عبارات را ارزیابی کن و موارد ناخواسته را رد کن.

مرحله ۶: تنها عباراتی که قابلیت ارزیابی دارند، به سمت تابع برازندگی $E_r$ ارسال خواهند شد.

مرحله ۷: اگر $E_r \to 0$ رفته و شرایط ترمینال ارضاء شود، آن‌گاه متوقف شو. درغیراین‌صورت از قاعده بروز رسانی سراسری استفاده کن.

مرحله ۸: بهترین سفرهای نسل قبلی را شناسایی کن.

مرحله ۱۰: $k$ مورچه مشابه را از طریق بهترین سفرهای نسل قبلی عبور بده و به مرحله ۳ برو.




**شکل ۱.گراف حرکت مورچه**

## الگوریتم ترکیبی GA-ACO

نقطه قوت الگوریتم ژنتیک در گسترده بودن فضای جستجوی آن می‌باشد و از ایرادات آن سرعت کم همگرایی آن به سمت جواب بهینه است. از طرفی الگوریتم ACO سرعت زیادی در همگرایی مسئله به سمت پاسخ بهینه دارد ولی در بعضی مواقع در پاسخ محلی گیر می‌کند. برای حل این مشکل این دو الگوریتم را با هم ترکیب می‌کنیم تا پاسخی با سرعت همگرایی بالاتر و با فضای جستجوی بزرگتر بیابیم. در الگوریتم ترکیبی ژنتیک و کلونی مورچه از عملگرهای هر دو این الگوریتم‌ها به صورت ترکیبی بر اساس فلوچارت (شکل۲) استفاده می‌گردد. (۳)

## حل معادله دیفرانسیل با استفاده از روش برنامه ریزی ترکیبی ژنتیک و کلونی مورچگان

برای حل معادله دیفرانسیل داده شده با شرایط اولیه/مرزی مناسب، الگوریتم دارای مراحل زیر است. (۴)

- مقداردهی اولیه
- ارزیابی برازندگی
- عملیات ژنتیک و کلونی مورچگان
- کنترل خاتمه

## ارزیابی برازندگی

- **حالت ODE**

ODEها را در فرم زیر توصیف می‌کنیم

$$f\left(x, y, y^{(1)}, \dots, y^{(n-1)}, y^{(n)}\right) = 0, \quad x \in [a, b] \quad (2)$$

بطوریکه $y^{(n)}$ نشان‌دهنده مشتق nام $y$ می‌باشد. فرض کنید شرایط مرزی یا اولیه به شکل زیر باشد:

$$g_i\left(x, y, y^{(1)}, \dots, y^{(n-1)}\right)_{x=t_i} = 0, \quad i = 1, \dots, n$$

بطوریکه $t_i$ یکی از دو نقطه پایانی $a$ یا $b$ است. مراحل ارزیابی برازندگی جمعیت به شکل زیر است:

- N نقطه هم فاصله $(x_0 و x_1 و \dots و x_{N-1})$ را در بازه مربوطه انتخاب کنید.


**شکل ۲. فلوچارت حل مسئله توسط الگوریتم ترکیبیGA-ACO**

- برای هر کرومزم $i$

الف) مدل متناظر $M_i(x)$ را بسازید.

ب) کمیت زیر را محاسبه کنید

$$E(M_i) = \sum_{j=0}^{N-1}\left(f\left(x_j, M_i^0(x_j), \dots, M_i^{(n)}(x_j)\right)\right)^2 \quad (۳)$$

ج) جریمه وابسته $P(M_i)$ را به شکلی که در زیر نشان داده شده، محاسبه کنید.

تابع جریمه $P$ وابسته به شرایط اولیه/مرزی بوده و به شکل زیر می‌باشد:

$$P(M_i) = \lambda \sum_{k=1}^{n} g_k^2\left(x, M_i, M_i^{(1)}, \dots, M_i^{(n-1)}\right)_{|x=t_k} \quad (۴)$$

بطوریکه $\lambda$ یک عدد مثبت می‌باشد.

د) مقدار برازندگی کرومزم برابر است با:

$$v_i = E(M_i) + P(M_i) \quad (۵)$$

- **حالت PDE**

ما در اینجا فقط PDEهای بیضوی دو و سه متغیره دارای شرایط مرزی دیریکله را در نظر می‌گیریم. تعمیم این فرایند به انواع شرایط مرزی و ابعاد بالاتر به راحتی قابل انجام است. PDEرا به فرم کلی زیر در نظر می‌گیریم:

$$f\begin{pmatrix} x, y, \psi(x,y), \frac{\partial}{\partial x}\psi(x,y), \frac{\partial}{\partial y}\psi(x,y), \\ \frac{\partial^2}{\partial x^2}\psi(x,y), \frac{\partial^2}{\partial y^2}\psi(x,y) \end{pmatrix} = 0(12) \quad (۶)$$

به ازای $x \in [x_0 و x_1]$ و $y \in [y_0 و y_1]$. شرایط مرزی وابسته دیریکله به صورت زیر توصیف می‌شود:

$$E(M_i) = \sum_{j=1}^{N^2} f\begin{pmatrix} x_j, y_j, M_i(x_j, y_j), \frac{\partial}{\partial x}M_i(x_j, y_j), \\ \frac{\partial}{\partial y}M_i(x_j, y_j), \frac{\partial^2}{\partial x^2}M_i(x_j, y_j), \\ \frac{\partial^2}{\partial y^2}M_i(x_j, y_j) \end{pmatrix}^2$$

$$v_i = E(M_i) + P(M_i) \quad (۷)$$

در رابطه ۱۰ در صورتی که هدف به دست آوردن جواب عمومی باشد و شرایط مرزی در نظر گرفته نشود $P(M_i)$ در نظر گرفته نمی شود و مقداری برابر با ۰ دارد. در تمامی این توابع برای ارزیابی دقیقتر مسئله تابع برازندگی نهایی را به صورت رابطه زیر در نظر می‌گیریم که در این شرایط اگر خروجی تابع خطا صفر گردد حاصل تابع برازندگی ۱ می‌شود و در این حالت هر چه خطا افزایش یابد مقدار تابع برازندگی به صفر میل می نماید که در این شرایط سیستم از پاسخ اصلی دور می‌گردد.

$$Fittnes_i = \frac{1}{1+v_i} \quad (۸)$$

که در این رابطه $v_i$ همان مقدار تابع کل خطاست. این معادله برای تمام معادلات دیفرانسیلی ODE و PDE قابل استفاده می‌باشد.

## گرامر حل مسئله

تکامل گرامری یک الگوریتم تکاملی است که می‌تواند در هر زبانی کد تولید کند. این الگوریتم به عنوان ورودی‌ها نیازمند تعریف گرامری BNF زبان هدف و تابع برازندگی مناسب می‌باشد. برخلاف برنامه‌نویسی ژنتیک کلاسیک کرومزم‌ها در تکامل گرامری به صورت درخت‌های تجزیه توصیف نمی‌شوند بلکه در قالب بردارهای عدد صحیح بیان می‌شوند. هر عدد صحیح نشان‌دهنده یک قاعده تولید از گرامر BNF می‌باشد. در این رویکرد برای حل مسئله کاراکترهای مورد استفاده را به سه بخش نماد، تابع و عملگر تقسیم می‌نماییم که در این شرایط جمعیتهای تولیدی در هر سطح بر اساس نمادها و توابع تعیین می‌گردد. در جدول (۱) نمادها، توابع و عملگرهای مورد استفاده به همراه کد مربوط به هر یک از آنها نمایش داده شده است.

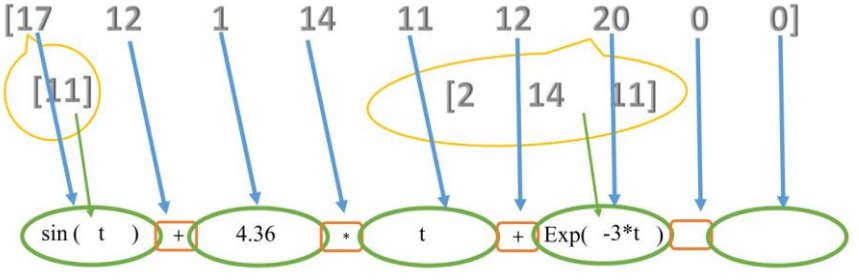
**جدول ۱. نمادهاو توابع و عملگرهای سیستم**

| | نمادها | عملگرها | توابع |
|---|---|---|---|
| کاراکترها | t 0 9 8 7 6 5 4 3 2 1 | ^ / × − + | Sin, Cos, Log, Exp, Sqrt |
| کد مربوط به هر کاراکتر | 11 10 9 8 7 6 5 4 3 2 1 | 16 15 14 13 12 | 17 18 19 20 21 |

در این مقاله با توجه به سناریو در نظر گرفته شده نیازی به کاراکترهای ‘)’ و ‘(‘ نیست و همچنین کاراکتر ^ برای راحت تر شدن محاسبات در نظر گرفته شده است. در این مقاله گراف تولید جمعیت به صورت دو سطحی در نظر گرفته شده که در هر سطح می توان ۹ کاراکتر قرار داد و در صورت استفاده از تابع در هر سطح سطح بعدی ایجاد می گردد. در تولید معادله در صورتی که در سطح اول از توابع استفاده گردد سیستم سطح دوم را برای آن کاراکتر نیز تولید می‌کند و این سطح بندی تا آخرین کاراکتر جلو می‌رود تا به بهترین پاسخ ممکن دست یابیم. در این حالت در بین نمادها و توابع، عملگرها قرار می‌گیرند و در این شرایط سیستم به صورت هوشمند سعی دارد بعد از علامت تقسیم از صفر استفاده نکند.در این شرایط جواب تولیدی می‌تواند با تعداد کاراکترهای مختلف به دست آید و تعداد کاراکترها با توجه به کاراکترهای ایجاد شده تعیین می‌گردد. در این تئوری اعداد به دو صورت تولید می گردند که در حالت اول اگر عدد مورد نظر ۱ باشد یک عدد اعشاری در فاصله [۵و۵−] تولید می گردد و اگر کد برابر ۲ باشد یکی عدد صحیح در محدوده [۱۵و۱۵−] ایجاد می‌شود. از طرفی اگر کد انتخابی ۰ باشد در این حالت آن کاراکتر خالی در نظر گرفته می‌شود.

برای مثال، یک جمعیت به صورت [۰ ۰ ۲۰ ۱۲ ۱۱ ۱۴ ۱۲ ۱۷] که عبارت $\sin(t) + 1 * t + \exp(2t)$ را تولید می‌کند و به خاطر وجود دو تابع یعنی ۱۷ و ۲۰ دو زیر ساخت برای سطح اول ایجاد می‌گردد که در سطح دوم قرار می‌گیرد که برای مثال این زیر ساخت‌ها عبارتند از [۱۱] و [۱۱ ۱۴ ۲] می‌باشند در این حالت معادله $\sin(t) + 1 * t + \exp(2 * t)$ است. در این مرحله با توجه به حضور اعداد ۱ در معادله یک عدد تصادفی اعشاری در محدوده مجاز تولید می‌گردد و برای عدد ۲ نیز یک عدد صحیح در محدوده ایجاد می‌شود و در نهایت معادله به صورت $\sin(t) + 4.36 * t + \exp(-3 * t)$ تولید می‌شود.

در این سیستم در صورتی که تابع در سطح اول قرار گیرد الگوریتم برای آن سیستم یک زیر سیستم دوم درست می‌کند که در صورت استفاده از تابع در آن نیز زیر سیستم سوم تولید می‌گردد. البته شایان توجه است که در سیستم مورد طراحی تعداد زیر سیستم‌ها با توجه به اندازه معادله تعیین می‌گردد یعنی می‌توان سطوح تولید معادله را از ۴ سطح هم بیشتر در نظر گرفت. در زیر نحوه تولید جمعیت ارائه گردیده است.




**شکل ۳. نحوه تولیدجمعیت توسط تئوری گراف ارائه شده**

## مثال‌ها

در این بخش، روش ارائه شده را برای حل انواع مختلفی از معادلات دیفرانسیل معولی و جزئی بکار برده و نتایج را ارائه می‌نماییم. برای حل، پایگاه داده‌ها را بر روی سیستم کامپیوتر خانگی با پردازشگر ۲۲۷۰ مگا هرتز Corei5 و حافظه داخلی 2 گیگا بایت و با نرم افزار MATLAB پیاده‌سازی نموده‌ایم. نتایج به خوبی دقت و سرعت بالای روش را در مقایسه با سایر روش‌ها نشان می‌دهد.

(نمونه ۱ ODE)

$$y''y' = \frac{-4}{t^3}$$
$$y'(1) = 0$$
$$y(1) = 0$$

پاسخ دقیق این معادله در بازه [۲و۱] عبارت است از

$$y = \log(t^2)$$

پاسخ حاصل از اجرای کد برای این مسئله به صورت شکل زیر است:

```
Loop Index = 2
Best Fitness = 1
Best Checked Solution = log(9.3687*(-t))+log(9.3687*(-t))
**********************
Final Solution = log(9.3687*(-t))+log(9.3687*(-t))
Elapsed time is 9.531897 seconds.
```

با ساده‌سازی داریم:

$$\log(9.3687(-t)) + \log(9.3687(-t)) =$$
$$2\log(-9.3687t) = C_1 \log(t^2) + C_2$$

این پاسخ، در حلقه دوم و پس از 9.53 ثانیه بدست آمده است.

(نمونه ۲ ODE)

$$\frac{y''}{y'} = -\tan(t) \text{ و } y(0) = 0 \text{ و } y'(0) = 0$$

پاسخ دقیق این معادله در بازه [۲و۰] عبارت است از

$$y = \sin(t)$$

پاسخ حاصل از اجرای کد برای این مسئله به صورت شکل زیر است:

```
Loop Index = 1
Best Fitness = 1
Best Checked Solution = sin((-t))+(-6.4333)^(-9.4994)
**********************
Final Solution = sin((-t))+(-6.4333)^(-9.4994)
Elapsed time is 2.421145 seconds.
```

ساده‌سازی نشان می‌دهد، عبارت بدست آمده برابر است با

$$\sin((-t)) + (-6.4333)^{-9.4994} =$$
$$\sin(t) + C_1$$

این پاسخ، در همان حلقه اول و پس از 2.42 ثانیه بدست آمده است.

(نمونه ۳ ODE)

$$y'' + y' = 2 + 2t \text{ و } y(0) = 0 \text{ و } y'(0) = 0$$

پاسخ دقیق این معادله در بازه [۱و۰]عبارت است از

$$y = t^2$$

پاسخ حاصل از اجرای کد برای این مسئله به صورت شکل زیر است:

```
Loop Index = 2
Best Fitness = 1
Best Checked Solution = t*t
************************
Final Solution = t*t
Elapsed time is 1.011793 seconds.
```

این پاسخ، در حلقه دوم و پس از یک ثانیه بدست آمده است.

(نمونه ۴ ODE)

$$y' = 2\sin(t)\cos(t) \text{ و } y(0) = 0 ,$$
$$y'(0) = 0$$

پاسخ دقیق این معادله در بازه [۰و۱]عبارت است از
$y = \sin^2(t)$

پاسخ حاصل از اجرای کد برای این مسئله به صورت شکل زیر است:

```
Loop Index = 4
Best Fitness = 1
Best Checked Solution = sin(t)*sin(t)
************************
Final Solution = sin(t)*sin(t)
Elapsed time is 3.124991 seconds.
```

این پاسخ، در حلقه چهارم و پس از 3.12 ثانیه بدست آمده است. شکل ۴، نمودار همگرایی را برای این مسئله نشان می‌دهد. محور افقی گام و محور عمودی معیاری از میزان مقبول بودن پاسخ بدست آمده است.

(نمونه ۵ ODE)

$y'' = y$و $y(0) = 1$ و $y'(0) = 1$

پاسخ دقیق این معادله در بازه [۰و۱] عبارت است از
$y = e^t$

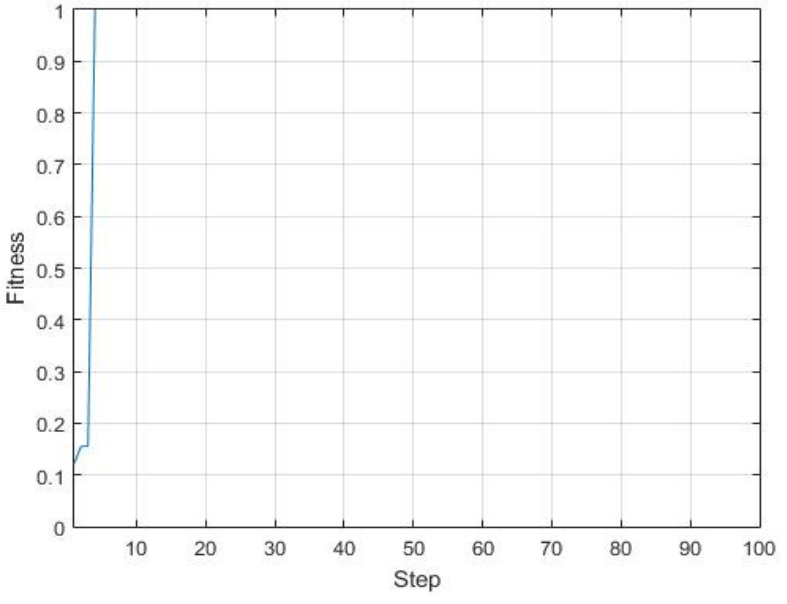
پاسخ حاصل از اجرای کد برای این مسئله به صورت شکل زیر است:

```
Loop Index = 6
Best Fitness = 1
Best Checked Solution = cos(exp((-8.8692)))/exp((-t))
************************
Final Solution = cos(exp((-8.8692)))/exp((-t))
Elapsed time is 2.256184 seconds.
```

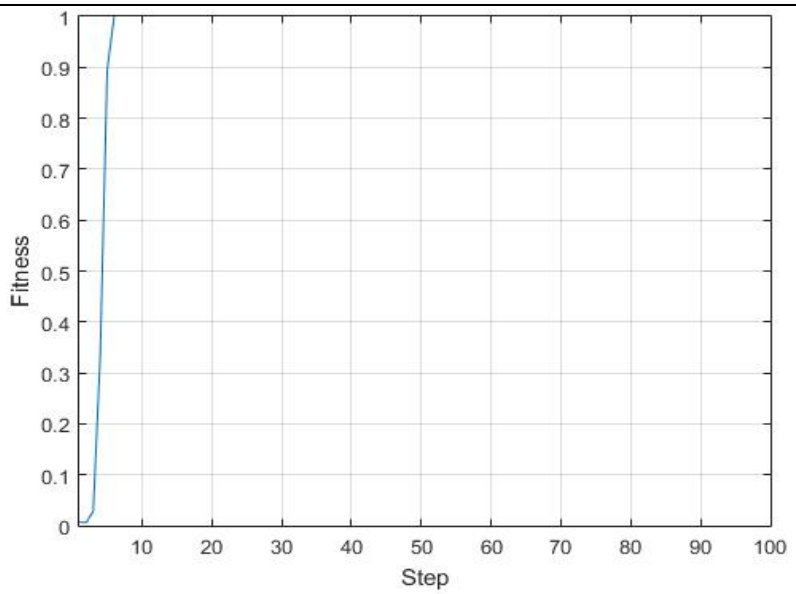
این پاسخ، در حلقه ششم و پس از 2.26 ثانیه بدست آمده است. با ساده‌سازی خواهیم داشت:

$$\frac{\cos(exp(-8.9))}{\exp(-t)} = \frac{1}{e^{-t}} = e^t$$

شکل ۵، نمودار همگرایی را برای این مسئله نشان می‌دهد.




**شکل ۴. مسیرهمگرایی جواب برای ۴ODE**




**شکل ۵. مسیر همگرایی جواب برای۵ODE**

(نمونه ۶ ODE)

$y' = \frac{1}{t}$ و $y(e) = 0$ و $y'(1) = 1$

پاسخ دقیق این معادله در بازه [ e و ۱ ] عبارت است از $y = \ln(t)$

پاسخ حاصل از اجرای کد برای این مسئله به صورت شکل زیر است:

```
Loop Index = 2
Best Fitness = 1
Best Checked Solution = log(t)-exp((-5.6024)^(-4.3527))
************************
Final Solution = log(t)-exp((-5.6024)^(-4.3527))
Elapsed time is 0.929219 seconds.
```

این پاسخ، در حلقه دوم و در کمتر از یک ثانیه بدست آمده است.

(نمونه ۷)

$\frac{y'}{y} = -\frac{1}{t}$ و $y(1) = 1$ و $y'(1) = -1$

پاسخ دقیق این معادله عبارت است از

$y = \frac{1}{t}$

پاسخ حاصل از اجرای کد برای این مسئله به صورت شکل زیر است که این پاسخ، در حلقه ششم و پس از 5.1 ثانیه بدست آمده است.

با توجه به اینکه

$\log(\exp(t)) = t, \sin\big((-3.0676)(-3.6063)\big) = -0.9977$

با ساده‌سازی خواهیم داشت:

$\log(\exp(t))^{\sin((-3.0676)(-3.6063))} = t^{-1} = \frac{1}{t}$

- **معادلات PDE**

مثال ۱) برای معادله با مشتقات جزئی زیر

$\frac{\partial \psi}{\partial x} + \frac{\partial \psi}{\partial y} = 2(x + y)$

پاسخ پیشنهادی برنامه عبارت است از:

```
Loop Index = 3
Best Fitness = 1
Step's Best Solution = x*(y)+x*(y)
************************
Final Solution = x*(y)+x*(y)
```

که معادل $2xy$ بوده و در گام سوم بدست آمده است.

مثال ۲) برای معادله با مشتقات جزئی زیر

$$\frac{\partial^2\psi}{\partial x^2}+\frac{\partial\psi}{\partial y}=-\psi+\cos(x+y)$$

پس از همگرایی به صورت زیر شکل ۶ پاسخ پیشنهادی برنامه عبارت است از:

```
Loop Index = 82
Best Fitness = 0.96576
Step's Best Solution = 1.0151^(x)*sin(x+y)
************************
Final Solution = 1.0151^(x)*sin(x+y)
```

که در گام ۸۲ بدست آمده است. برای محدوده مورد نظر یعنی $[-0.5و0.5]$:

$$1.0151^{x}\times\sin(x+y)\equiv\sin(x+y)$$

مثال ۳) برای معادله با مشتقات جزئی زیر

$$\frac{\partial\psi}{\partial x}\frac{\partial\psi}{\partial y}=\psi$$

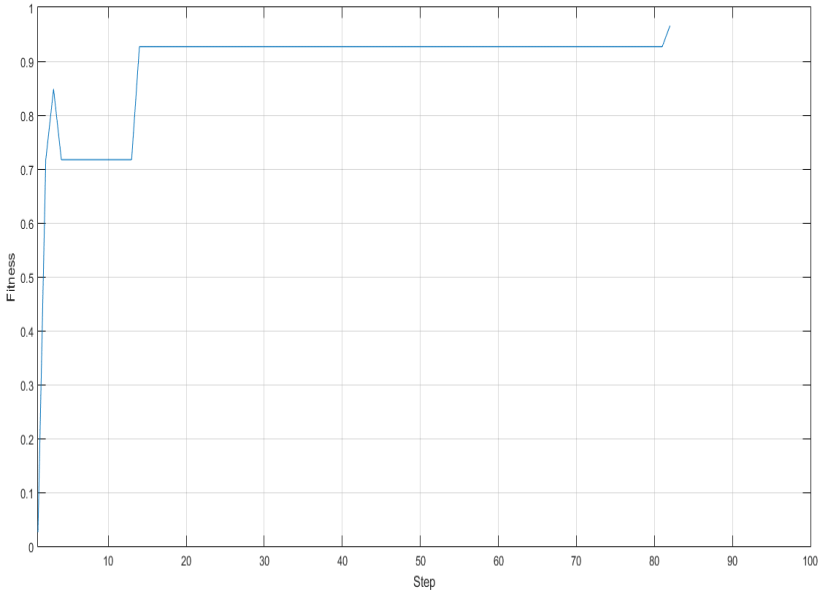
پس از یک همگرایی سریع، پاسخی به صورت:

```
Loop Index = 3
Best Fitness = 0.97551
Step's Best Solution = (-x)*(y)*log(abs(0.049463+0.53576))+(-x)*(y)*log(abs(0.049463+0.53576))
************************
Final Solution = (-x)*(y)*log(abs(0.049463+0.53576))+(-x)*(y)*log(abs(0.049463+0.53576))
```

پیشنهاد می‌کند که در گام سوم بدست آمده است. با ساده سازی

$$-xy\times log\big(abs(0.049463+0.53576)\big)-xy\times\log\big(abs(0.049463+0.53576)\big)=$$
$$-2xy\times log\big(abs(0.049463+0.53576)\big)=C_1xy$$

که با یک بررسی ساده، صحت پاسخ تائید می‌شود.




**شکل ۶. مسیر همگرایی جواب برای۲PDE**

## نتیجه‌گیری

در این مقاله یک روش جدید برای حل انواع مختلفی از معادلات دیفرانسیل معمولی و جزئی ارائه گردید. روش ارائه شده مبتنی بر برنامه‌نویسی ترکیبی ژنتیک- کلونی مورچگان (GA-ACO) است. این رویکرد جواب‌های آزمایشی تولید شده وخطای وابسته را مینیمم می‌نماید. مزیت این روش، تولید جوابهای تحلیلی با دقت و سرعت همگرایی بالابرای انواع گوناگونی از معادلات دیفرانسیل است. گرامر استفاده شده در این روش به تولید جمعیت برای نسلهای مختلف بر اساس تئوری گراف و به صورت دو سطحی پرداخت که باعث افزایش کارایی سرعت و دقت روش شده است.

## فهرست مراجع


[1] Peng, Y. Z. 2003 Exact solutions for some non linear partial differential equations. Physics Letters A 314, 401-408.

[2] Salzner, Y., Otto, P., and Ladik, J. 1990 Numerical solution of a partial differential equation system describing chemical kinetics and diffusion in a cell with the aid of compartmental-ization. Journal of Computational Chemistry, 11, 194-204.

[3] Culshaw, R.V., Ruan, S. 2000 A delay-differential equation model of HIV infection of CD+ 4 T-cells. Mathematical Biosciences 165, 27-39.

[4] Norberg, R. 1995 Differential equations for moments of present values in life, Insurance. Mathematics and Economics 17, 171-180.

[5] Srebrenik, S., Weinstein, H., and Pauncz, R.1973 Analytic calculation of atomic and molecular electrostatic potentials from the Poisson equation. Chem. Phys. Letters 20, 419-423.

[6] Lee, C., Lee, K., Kim, C.Ki., and Moon-Uhn, K. 1997 Variational Formulation of Poisson's Equation in Semiconductor at Quasi-Equilibrium and Its Applications, VOL. 44, NO. 9, September 1997.

[7] Perez, P., Gangnet, M., and, Blake, A. 2003 Poisson Image Editing. ACM Transactions on Graphics, 22(3), 313-318.

[8] Jean-Pierre Richard, 2003 A differential transformation approach for solving functional differential equation with multiple delays, Volume 39, Issue 10, 1667-1694.

[9] Mirzaee F., Hoseini S. F., 2017 A new collocation approach for solving systems of high order integro differential equation with variable, Applied Mathematics and Computation, Volume 311, 15, 272-282.

[10] Rudd K., Ferrari S. 2015 A constrained integration (CINT) approach to solving partial differential equations using artificial neural networks, Neuro computing 155, 277–285.

[11] Badakhshan K. P., Vahidian Kamyad A., 2007 Numerical solution of nonlinear optimal control problems using nonlinear programming, Appl. Math. Comput. 187 (2) 511–1519.

[12] Cao. H, Kang. L, Chen. Y and Yu. J, 2010 Evolutionary Modeling of Systems of Ordinary Differential Equations with Genetic Programming, Genetic Programming and Evolvable Machines, vol.1,309-337.

[13] Iba. H, Sakamoto. E, 2012Inference of Differential Equation Models by Genetic Programming, Proceedings of the Genetic and Evolutionary Computation Conference (GECCO 2012),788-795.

[14] Kamali M.Z.M., Kumaresan N. and Ratnavelu K.2015 Solving differential equations with ant colony programming, Applied Mathematical Modelling 39 3150–3163.

[15] Koza. J. R, 2002 Genetic Programming: On the programming of Computer by Means of Natural Selection. MIT Press: Cambridge, MA.

[16] H. Nojavan; S. Abbasbandy; T. Allahviranloo The use of radial basis functions by variable shape parameter for solving partial differential equations, Journal of new researcher in mathematics, Volume 5, Issue 17, 2019.

[17] N. Nyamoradi; A. Razani Existence solutions for new p-Laplacian fractional boundary value problem with impulsive

effects, Journal of new researcher in mathmatices, Volume 5, Issue 19, 2019.

[18] O'Neill. M and Ryan. C, 2003) Grammatical Evolution: Evolutionary Automatic Programming in arbitrary Language, volume 4 of Genetic programming. Kluwer Academic Publishers.

[19] H. Nojavan; S. Abbasbandy; T. Allahviranloo The use of radial basis functions by variable shape parameter for solving partial differential equations, Journal of new researcher in mathmatices, Volume 5, Issue 17, 2019.

[20] M.A. Ebadi; E.S. Hashemizadeh; A.H. Refahi Sheikhani, Zernike radial polynomials method for solving nonlinear singular boundary value problems arising in physiology, Journal of new researcher in mathmatices, Volume 5, Issue 19, 2019.

[21] I. Hossein Zade Shahbolaghi; R. Pourgholi; H. Dana Mazraeh; S.H. Tabasi Solving random inverse heat conduction problems using PSO and genetic algorithms, Journal of new researcher in mathmatices, Volume 5, Issue 19, 2019.