TUNING FUZZY SYSTEM FSSAM WITH GENETIC ALGORITHM

Georgieva Penka V., Burgas Free University, pgeorg@bfu.bg

Dimitrova Adreane, Burgas Free University, adreanedimitrova@gmail.com

Abstract: Forecasting is a key element of the process of decision making. Predicting the behaviour of financial markets is in the center of research of academic, financial and governmental institutions and private investors. The development of computer technologies leads to a new class of intelligent forecasting methods – the methods of soft computing.
This article describes a hybrid model, based on the techniques of fuzzy logic and genetic algorithms, which optimizes the parameters of the fuzzy system FSSAM.

Keywords: genetic algorithm, fuzzy system, hybrid system, optimization problems

НАСТРОЙВАНЕ НА ПАРАМЕТРИТЕ НА СИСТЕМА FSSAM С ГЕНЕТИЧЕН АЛГОРИТЪМ

Пенка В. Георгиева, Бургаски свободен университет, pgeorg@bfu.bg

Адреане Димитрова, Burgas Free University, adreanedimitrova@gmail.com

Абстракт: Прогнозирането е ключов елемент от процеса на вземане на решения. Прогнозирането на финансовите пазари е в центъра на изследвания от академични, финансови и правителствени институции, както и частни инвеститори. Развитието на компютърните технологии води до появата на нов клас интелигентни методи за прогнозиране – методите на софт компютинга.
Настоящата статия описва хибриден модел, основан на техниките на размитата логика и генетичните алгоритми, който оптимизира параметрите на размитата система FSSAM.

Ключови думи: генетичен алгоритъм, размита система, хибридна система, оптимизационни задачи

Генетични алгоритми

През 1975г. Джон Холанд предлага алгоритъм, който той нарича генетичен. Механизмът на генетичните алгоритми е анализиран и развит в последствие от Голдбърг, Де Жонг, Дейвис, Мюленбейн и др. Според [6] генетичните алгоритми имат три основни приложения – интелигентно търсене, оптимизация и машинно обучение.
Генетичните алгоритми дават възможност за намиране на оптимални решения, като те могат да бъдат прилагани и при задачи, при които целевата функция е прекъсната, недиференцируема или стохастична.

Формулировка на задачата на генетичния алгоритъм

Нека X е пространството на търсене, където $f: X \rightarrow R$ е целевата функция.
Да се намери $x_0 \in X$, такова че $(x_0) = sup_{x \in X} f(x)$.

Дефинират се кодираща и декодираща функции $c: X \rightarrow S$ и $\overline{c}: S \rightarrow X$, където S е множество от бинарни стрингове и $\overline{c}$ е инективна и $(c \circ \overline{c}) \equiv id_S$.
Тогава задачата е да се намери $s_0 \in S$, такова че $\overline{f}(s_0) = sup_{x \in S} \overline{f}(s). \overline{f} := f \circ \overline{c}$.

Основни елементи на генетичните алгоритми

Хромозоми

При генетичните алгоритми хромозомите представляват набор от гени, които кодират неизвестните променливи. Всяка хромозома представлява допустимо решение на поставената задача. Гените от своя страна могат да бъдат булеви, целочислени, с плаваща запетая и стрингови променливи (редици от букви и символи), както и всяка тяхна комбинация. Множеството от различни хромозоми (индивиди) съставят текущото поколение. Чрез еволюционни операции (селекция, рекомбинация и мутация) се достига до следващото поколение (потомство) [2].

Селекция

При генетичните алгоритми селекцията на най-добрите индивиди става въз основа на функцията на пригодност, даваща оценка на конкретния индивид. Съществуват различни методи за селекция при генетичните алгоритми. На фигура 4 са представени някои от тях [10]. Разнообразните методи за селекция предразполагат към изграждане на различни генетични алгоритми.

Рекомбинация

При генериране на ново поколение първата операция, която се прилага е рекомбинацията. При нея гените от родителите формират изцяло нова хромозома. Типичната рекомбинация при генетичните алгоритми е операция, изискваща два родителя. Целта на операцията е новата хромозома може да бъде по-добра от двамата родителя, ако вземе най-добрите характеристики от всеки родител.

Мутация

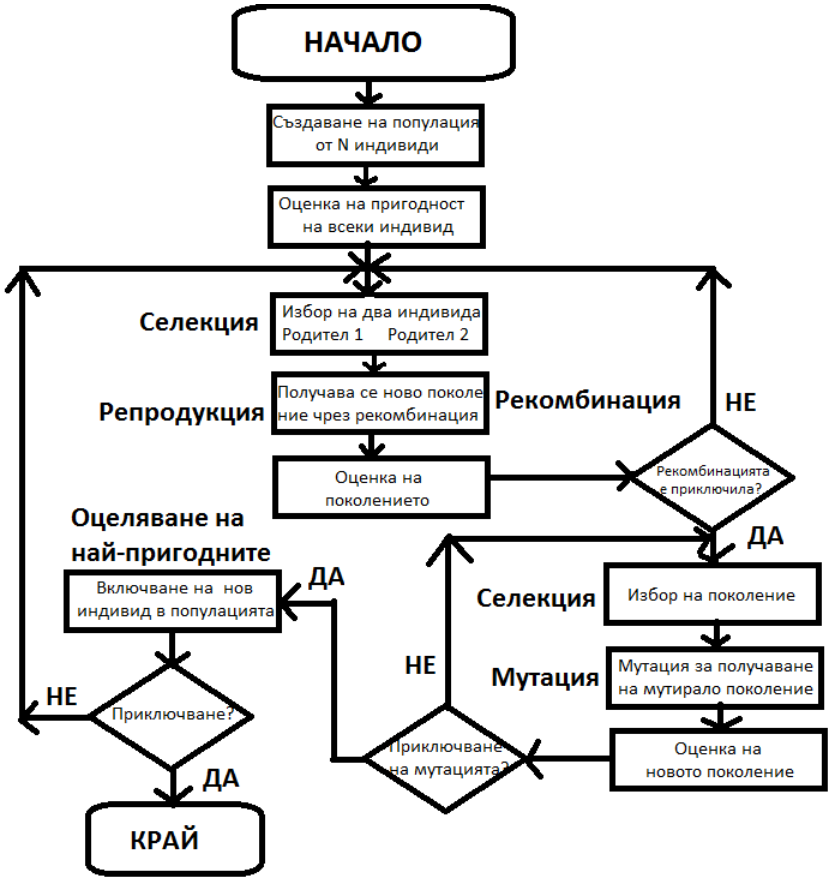
Новосъздаденото чрез селекция и рекомбинация потомство може да бъде подложено на мутация. При биологичните видове мутациите променят елементи от ДНК, като тези промени са породени главно от грешки при копирането на гените от родителите. При генетичните алгоритми мутацията е произволна промяна на стойността на ген в поколението.

Обща схема на генетичен алгоритъм

Стъпките за изпълнение на генетичен алгоритъм са (фиг. 1):

1. [Старт] Създава се първоначална произволна популация от N хромозоми (т.е. възможни решения на задачата)
2. [Фитнес] Изчислява се стойността на функцията на пригодност $f(x)$ за всяка хромозома x от популацията.

3. [Нова популация] Създава се нова популация чрез повтаряне на следните стъпки, докато следващата популация е завършена.
    - (а) [Селекция] Избира двама родители от популацията, според тяхната пригодност (колкото по-голяма е пригодността, толкова по-голям е шансът да бъдат избрани)
    - (б) [Рекомбинация] Рекомбинират се родителите, за да формират ново поколение. Ако не се извърши рекомбинация, поколението повтаря родителите.
    - (в) [Мутация] Поколението мутира в случайно избрана позиция.
    - (г) [Оцеляване] Новото поколение се приема в популацията.
4. [Включване] Новата популация се използва за следващи действия в алгоритъма.
5. [Проверка] Ако крайното условие е изпълнено, спира и връща най-доброто решение от новосъздадната популация.
6. [Цикъл] Връщане в стъпка 2.



Фиг. 1. Блок-схема на генетичен алгоритъм

ХИБРИДНИ СИСТЕМИ, ИЗПОЛЗВАЩИ РАЗМИТА ЛОГИКА И ГЕНЕТИЧНИ АЛГОРИТМИ

Размита логика, подобряваща еволюционни пресмятания

Размит генетичен алгоритъм е генетичен алгоритъм, който използва техниките на размита логика, за да подобри работата на генетичния алгоритъм, изменяйки различните му параметри. Размитата логика се употребява при търсенето в пространството от решения, а също и при локално търсене на генетичния алгоритъм [8], [9]. Управлението на параметрите на генетичния алгоритъм подобрява неговата ефикасност и скоростта на сходимост. В [7] се предлага размита система, настройваща параметрите на генетичния алгоритъм и избора на генетични оператори. Съществен аспект на този подход е намирането на баланс между изчислителните ресурси, разпределени между размитата логика и генетичните алгоритми.

Генетични размити системи

При генетичните размити системи се прилагат еволюционни пресмятания върху размитите системи по няколко начина: генетични алгоритми, генетично програмиране и еволюционни стратегии. Фигура 2 представя обща схема на генетична размита система [5].

Фиг. 2. Генетична размита система

Видове генетични размити системи

За да се проектира генетична размита система, първата стъпка е да се вземе решение кои части от размитата система да бъдат оптимизирани. Следваща стъпка е кодирането им в хромозоми, които да бъдат оптимизирани от генетичния алгоритъм. Подходите са категоризирани в две направления: настройване и учене:

- избор на параметри на генетичния алгоритъм така, че базата с правила остава непроменена, а параметрите на размитата система се настройват. Този процес се прилага, ако предварително съществува база знания.
- генетично учене е по-сложен начин, който оптимизира само базата от правила или цялата база знания. При този процес няма нужда от предварително зададени правила и базата от правила се създава по време на генетичното учене.

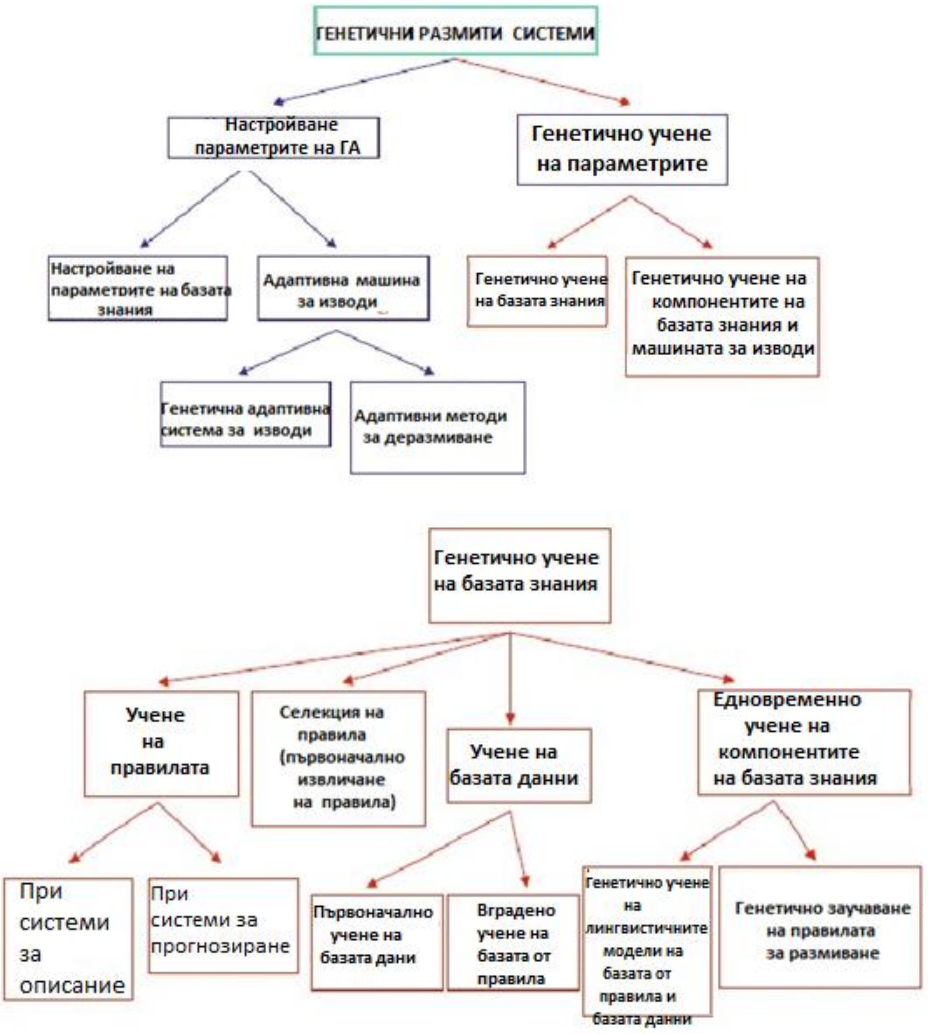
Разграничават се три групи генетични размити системи, в зависимост от компонентите на базата знания, включени при процеса генетично учене.

1) *Настройване на базата данни чрез генетичен алгоритъм* -Настройването на функциите за мащабиране и функциите за принадлежност е важна задача при изграждането на размити системи. Възможно е на функциите на мащабиране или функциите на принадлежност да бъдат зададени параметри и да бъдат

адаптирани чрез генетични алгоритми, за да бъдат техните параметри пригодни на функцията. Предложени са няколко метода за настройване на базата данни. Всяка хромозома представя различни означения на базата данни,т.е., всяка хромозома съдържа кодиран стринг на променливите на функции на принадлежност. Могат да бъдат разгледани две възможности според това дали размитият модел е описателен или апроксимиращ.

2) *Учене на базата от правила чрез генетичен алгоритъм* При този тип учене съществува предварително определена библиотека от функции на принадлежност. Генетичните алгоритми се прилагат, за да се произведе подходяща база от правила, използвайки хромозоми, които кодират правила или цяла база от правила.

3) *Учене на базата знания чрез генетичен алгоритъм* - Съществуват подходи, които представят променлива дължина на хромозомите, други кодират постоянен брой правила и техните функции на принадлежност, а трети работят с хромозомите като кодират по едно правило вместо цялата база знания. Фигура 3 представя видове генетични размити системи [23].



Фиг. 3. Видове генетични размити системи

При проектиране на размита система първоначално се настройват базата данни или параметрите на машината за размити изводи. Три са възможностите за настройване :

1. Настройване на параметрите на базата знания, като се настройват параметрите на функцията на принадлежност. Процесът променя формата на функциите на принадлежност и не променя дължината на хромозомите.
2. Генетични размити системи за изводи – основната цел на този метод е прилагане на параметрите в системите за изводи, които се наричат адаптивни системи за изводи за по-голямо взаимодействие правилата за размиване и по-точни модели за размиване.
3. Адаптивни методи за деразмиване. Един от модулите на системата за изводи е модула за деразмиване, който се оптимизира чрез генетични алгоритми.

Настройване на функциите на принадлежност чрез генетични алгоритми

Генетичен алгоритъм може да настройва и базата с функции на принадлежност. Основната задача е определяне на параметрите на функцията и след това адаптирането им.

Видове функции на принадлежност

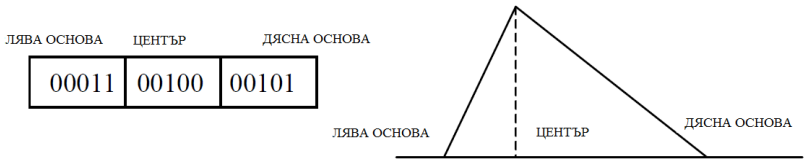
Основните групи функции на принадлежност са: *на части линейни функции и диференцируеми функции*

1) На части линейни функции – триъгълни и трапецовидни

Λ – функция (триъгълна)	$\Lambda(x,a,b,c)=\begin{cases}0, & x<a,\ x>c\\ \dfrac{x-a}{b-a}, & a\le x<b\\ \dfrac{c-x}{c-b}, & b\le x\le c\end{cases}$
Π – функция (трапецовидна)	$\Pi(x,a,b,c,d)=\begin{cases}0, & x<a,\ x>d\\ \dfrac{x-a}{b-a}, & a\le x<b\\ 1, & b\le x<c\\ \dfrac{d-x}{d-c}, & c\le x\le d\end{cases}$

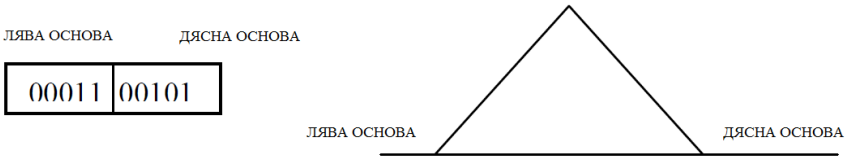
Кодиране на триъгълните функции на принадлежност

1. Триъгълна функция на принадлежност се описва с три параметъра, условно наречени: лява основа, център и дясна основа (фиг. 4).



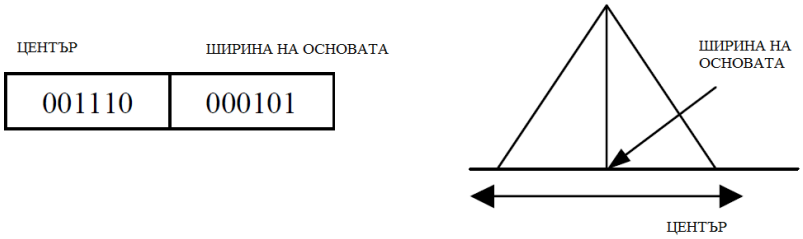
Фиг. 4. Двоично кодирана хромозома на триъгълна функция на принадлежност

2. При симетричните триъгълни функции на принадлежност (равнобедрен триъгълник) параметрите са два: уаловно лява основа и дясна основа (фиг. 5).



Фиг. 5. Двоично кодирана хромозома на симетрична триъгълна функция на принадлежност

3. Възможно е и при симетричните триъгълни функции на принадлежност да се кодират и настроят центъра и ширината на основата. Фигура 6 представя хромозома на симетрична функция на принадлежност с кодиран център и ширина на основата.



Фиг. 6. Генетично представяне на центъра и ширината на основата на симетрична триъгълна функция на принадлежност

Кодиране на на трапецовидни функции на принадлежност

За да се кодират трапецовидни функции на принадлежност, е необходим още един параметър за хромозомата.

2) Диференцируеми функции

Примери за диференцируеми функции са Гаусова, Бел и сигмоидална функция.

Гаусова функция	$gaussian\,(x,\beta,\alpha) = e^{-\frac{1}{2}\left(\frac{x-\alpha}{\beta}\right)^2}$

Бел функция	$bell(x,\alpha,\beta,\gamma) = \frac{1}{1+\left\|\frac{x-\gamma}{\alpha}\right\|^{2\beta}}$
Сигмоидална функция	$sig(x,\alpha,\beta) = \frac{1}{1+e^{-\alpha.(x-\beta)}}$

За да се преведат параметрите на функцията в генетична информация, се използва двоичен код, като при гаусовата и сигмоидалната функции се кодират по два параметъра (α, β), а при Бел функцията – три (α, β, γ).

Кодиране на правилата

При апроксимиращия процес всяка хромозома, образуваща популация, кодира цялата база знания. Всички те кодират базата от правила, правилата, и разликата между тях са функциите на принадлежност, т.е. определянето на базата данни.

Имайки предвид параметричното представяне на триъгълните функции на принадлежност, базирани на 3 тройкит цели числа, всяко правило

$$R_i\text{: Ако } x_1 \text{ е } A_{i1} \text{ и ... и } x_n \text{ е } A_{in}, \text{ то } y \text{ е } B_i,$$

от определена база знания, е кодирано в хромозомата C_{li} :

$$C_{li} = (a_{i1}, b_{i1}, c_{i1}, \dots, a_{in}, b_{in}, c_{in}, a_i, b_i, c_i)$$

където A_{ij}, B_i параметрично се представят (a_{ij}, b_{ij}, c_{ij}),(a_i, b_i, c_i), $i = 1, \dots, m$ (m – броят на правилата), $j = 1, \dots, n$ (n – броят на входните данни).

Тогава цялата база от правила, която се свързва с базата данни, е представена от цялата хромозома C_l:

$$C_l = C_{l1}\, C_{l2} \dots C_{lm}$$

Тази хромозома може да бъде кодирана като двоичен или реален индивид.

При описателния процес всяка хромозома кодира различна база данни. Първоначалното размито деление е представено като масив, състоящо се от $3 \cdot N$ реални стойности, където N е броя на термите, които формират лингвистичната променлива от множеството на терми. Базата данни, в която са включени m лингвистични променливи, се кодира като хромозома с точна дължина, изградена от реални числа C_j, съставена от събиране на дяловото представяне на всяко едно от променливите *размити деления:*

$$C_{ji} = (a_{i1}, b_{i1}, c_{i1}, \dots, a_{i1Ni}, b_{i1Ni}, c_{i1Ni})$$

$$C_j = C_{j1} C_{j2} \dots C_{jm}$$

където C_{ji} представя размитото деление, отговарящ на i- тата променлива.

РЕАЛИЗАЦИЯ НА МОДЕЛА

Хибридната система, се състои от генетичен алгоритъм, които оптимизира стойностите на параметрите на изходящата променлива на размитата система FSSAM. Тази размита система е подробно описана в [1], [3] и [4].

Преди стартирането на алгоритъма първоначално се декларират параметри, които влияят на операторите му. Това са *брой поколения, размер на популацията, брой променливи, битове за променлива, вероятност за мутация P_m, вероятност за рекомбинация P_c, елитност, горна и долна граница на задачата*. Горната и долна граница са интервали, в които могат да се изменят променливите на целевата функция.

Първоначално се създава популация от произволни кандидат-решения на оптимизационната задача. Тя е случайно генерирана с предварително зададен *размер на популацията*. Необходимо е хромозомите да се кодират от двоични в реални стойности. Преобразуването на векторите се осъществява според зададените *размер на популацията, брой променливи, брой битове и долна и горна граници.*
След като вече е образувана популация, се изчислява пригодността й към дадената целева функция.

Целевата функция на оптимизационната задача е:

$$Y = (Q(1) - Q(2))^2 + (Q(1) - Q(3))^2 + (Q(2) - Q(3))^2,$$

където $Q(k)$, $k=1, 2, 3$ са деразмитите стойности на изходната променлива, получени през различни интервали от време.
От стойностите на целевата функция се избират най-пригодните индивиди и върху тях се изпълнява операторът селекция. Подходящо е селекцията да се осъществи по рулетковия метод. Вероятността за селекция на всеки индивид се определя като отношение на стойността на целевата му функция към сумата от стойностите на целевата функция на всички индивиди. За да работи за задачи за минимизация се преизчисляват стойностите на целевата функция на всеки индивид, така че стойността на най- пригодния да бъде максимална. При този метод окръжността се разделя на N на брой сектори, пропорционални на вероятността за селекция на индивидите. Всеки път когато се завърти рулетката, се избира един индивид. При програмната реализация на рулетковия метод се използва линейно представяне на колелото. Използва се генератор на случайни числа в интервала (0,1). Това число се умножава по сумата от всички стойности на целевата функция, като се натрупва сума до достигане на случайното число. Взима се номерът на индивида с последната прибавена стойност на целева функция. По този начин индивиди с по-големи стойности на целевата функция получават пропорционално по-големи вероятности да бъдат избрани.
След като са селектирани родители, започва репродукцията за получаване на ново поколение. Изпълнява се рекомбинация. Избрана е рекомбинация в една точка, при която родителите разменят генната си информация в една точка с вероятност за рекомбинация P_c, зададена в началото на алгоритъма.
При операторът мутация за вход се използва рекомбинираното поколение, което мутира с вероятност за мутация P_m. За генетичния алгоритъм е избрана мутация с промяна на бита. Създава се и матрица, която показва в кой ген е извършена мутацията.
За новосъздаденото поколение отново се пресмятат стойностите на целевата функция и за най-пригодните се извършва проверка за елитност. След генерирането на всяко поколение, по-пригодните индивиди и техния генетичен материал може да бъде загубен, заради селекция (при него не се избират пригодни индивиди), рекомбинация и мутация могат да доведат до влошаване на пригодните индивиди. За да се запазят техните добри белези, генетичният им материал трябва да се запази в алгоритъма. Този процес се нарича елитност.

РЕЗУЛТАТИ ОТ ИЗСЛЕДВАНЕТО

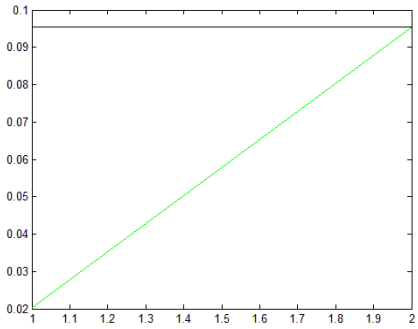
След успешното реализиране на модела в средата MatLab са проведени експерименти с реални данни от БфБ-София АД.
Преди генетичната оптимизация, размитата система има зададени стойности на параметрите PAR1=0; PAR2=0,25; PAR3=0,5; PAR4=0,75 и PAR5=1.
Таблица 1 показва промяната на PAR1 и Y при промяна на броя на поколенията, а останалите фактори запазят своите стойности. PAR1 е входната лингвистична

променлива X във функциите на принадлежност. Y е оптимална за функцията на пригодност за алгоритъма и представя оптимална стойност на Q-мярката.

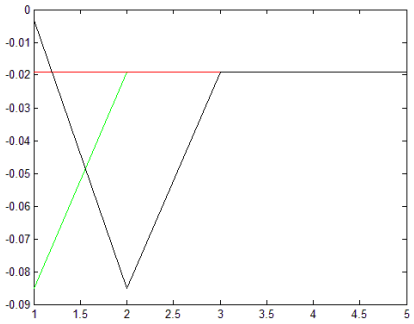
nGen	nPop	P_m	P_c	PAR1	Y
2	6	0,1	0,75	0,0005	0,0955
5	6	0,1	0,75	0,0006	-0,0192
10	6	0,1	0,75	0,0004	0,0292
50	6	0,1	0,75	0,0004	-0,0295

Табл. 1. Промяна на PAR1 и Y при промяна на nGen

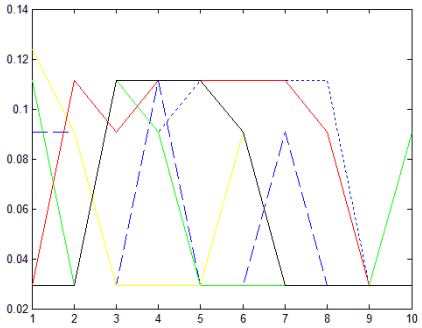
На фигура 7 е показано изменението на векторите на PAR1 при промяна на броя на поколенията. При nGen=5 стойността на Q е най-близка до 0, а също така и сходимостта се достига на третото поколение, за разлика от останалите.



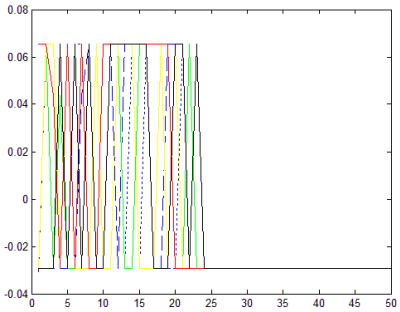
nGen=2; nPop=6; Pm=0,1;Pc=0,75



nGen=5; nPop=6; Pm=0,1; Pc=0,75



nGen=10; nPop=6; Pm=0,1; Pc=0,75



nGen=50; nPop=6; Pm=0,1; Pc=0,75

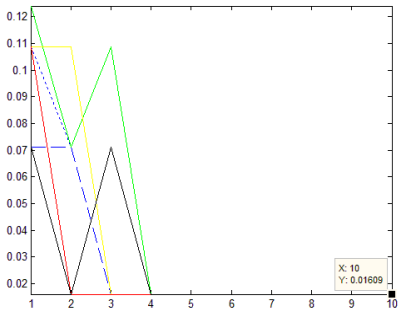
Фиг. 7. Изменение на векторите на PAR1 при промяна на броя на поколенията

В таблица 2 е показана промяната на PAR1 и Y при промяна на размера на популациите, а останалите фактори запазят своите стойности. При nPop 10 и 20 стойностите на Q са еднакви и най-оптимални, но сходимостта се достига по-бързо при стойност 10 за размер на популациите.

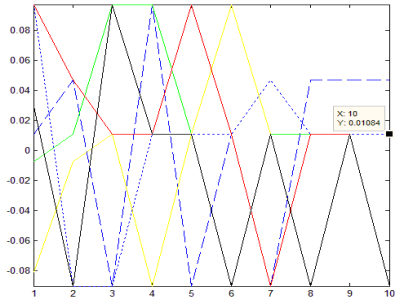
nGen	nPop	Pm	Pc	PAR1	Y
10	5	0,1	0,75	/	/
10	10	0,1	0,75	0,161	0,003
10	20	0,1	0,75	0,0467	0,003
10	30	0,1	0,75	-0,0344	0,006

Табл. 2. Промяна на PAR1 и Y при промяна на nPop

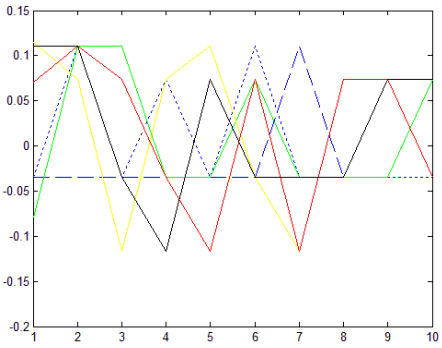
На фигура 8 е показано изменението на сходимостта на векторите на PAR1 при промяна на размера на популациите.
При nGen=10; nPop=5; Pm=0,1; Pc=0,75 – грешка при рекомбинацията - невъзможно да бъде изпълненп. Направен е и експеримент при други нечетни стойности на nPop, като 7, 9 и 19, но отново не може да се изпълни операторът рекомбинация.



nGen=10; nPop=10; Pm=0,1; Pc=0,75



nGen=10; nPop=20; Pm=0,1; Pc=0,75



nGen=10; nPop=30; Pm=0,1; Pc=0,75

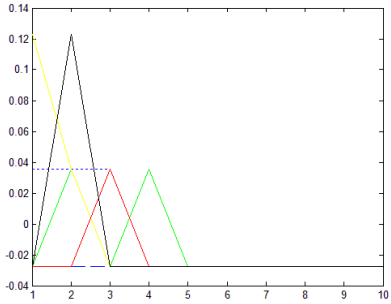
Фиг. 8. Изменение на векторите на PAR1 при промяна на размера на полулацията

Таблица 3 показва промяната на PAR1 и Y при промяна на вероятността за мутация, а останалите фактори запазят своите стойности.

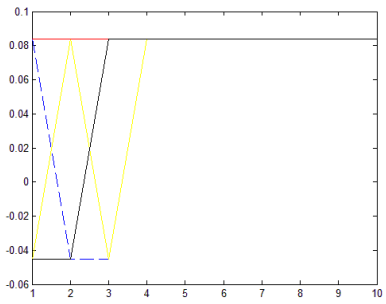
nGen	nPop	Pm	Pc	PAR1	Y
10	6	0,05	0,75	0,003	-0,0275
10	6	0,2	0,75	0,0837	0,0003
10	6	0,5	0,75	-0,0765	0,003
10	6	0,8	0,75	0,0079	0,0005

Табл. 3. Промяна на PAR1 и Y при промяна на Pm

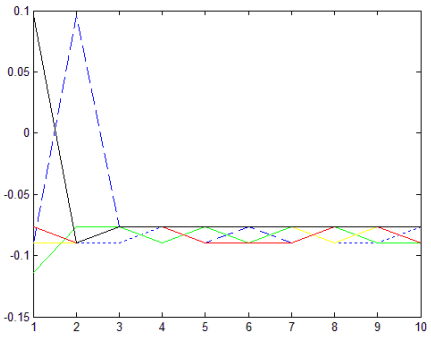
На фигура 9 е показано показват изменението на сходимостта на векторите на PAR1 при промяна на вероятността за мутация.



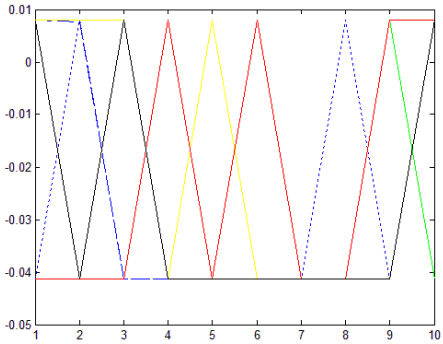
nGen=10; nPop=6; Pm=0,05; Pc=0,75



nGen=10; nPop=6; Pm=0,2; Pc=0,75



nGen=10; nPop=6; Pm=0,5; Pc=0,75



nGen=10; nPop=6; Pm=0,8; Pc=0,75

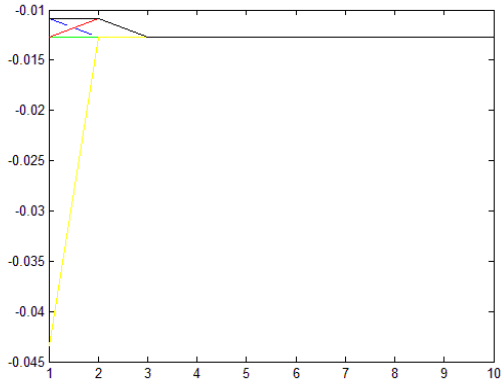
Фиг. 9. Изменение на векторите на PAR1 при промяна на вероятността за мутация

Таблица 4 показва промяната на PAR1 и Y при промяна на вероятността за рекомбинация, а останалите фактори запазят своите стойности. При по-малка стойност на P_c се достигат по-оптимални стойности на PAR1 и Q, а също така и по-бърза сходимост.

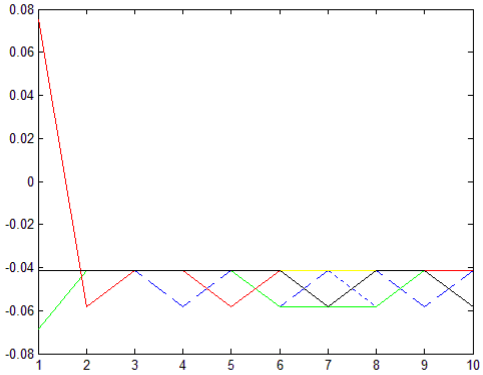
nGen	nPop	Pm	Pc	PAR1	Y
10	6	0,1	0,25	-0,0127	0,0002
10	6	0,1	0,5	-0,0582	0,0004
10	6	0,1	0,75	-0,1121	0,0004
10	6	0,1	0,9	-0,1	0,0006

Табл. 4. Промяна на PAR1 и Y при промяна на Pc

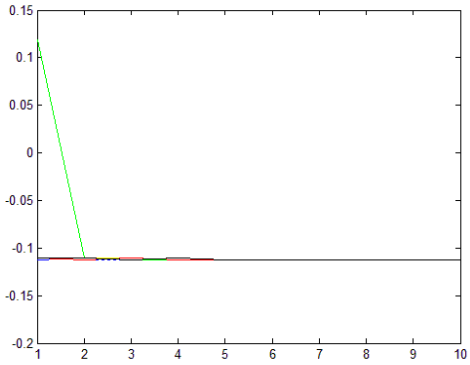
На фигура 10 е показано изменението на сходимостта на векторите на PAR1 при промяна на вероятността за рекомбинация.



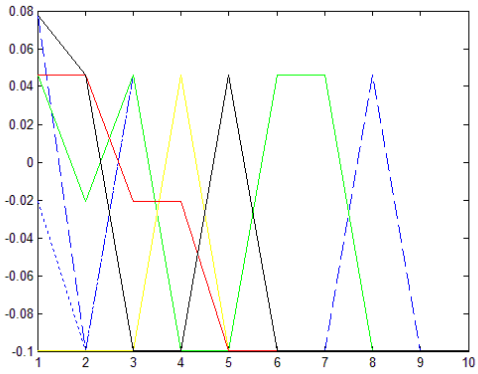
nGen=10; nPop=6; Pm=0,1; Pc=0,25



nGen=10; nPop=6; Pm=0,1; Pc=0,5



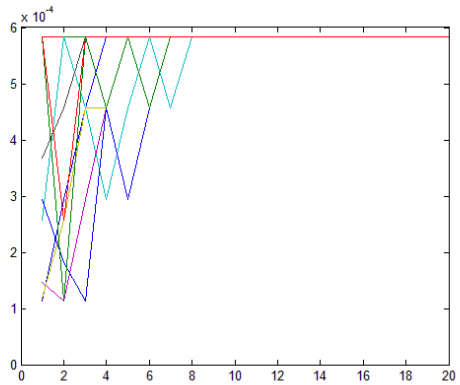
nGen=10; nPop=6; Pm=0,1; Pc=0,75



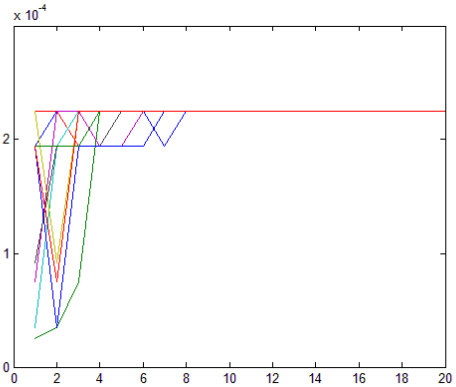
nGen=10; nPop=6; Pm=0,1; Pc=0,9

Фиг. 10. Изменение на векторите на PAR1 при промяна на вероятността за рекомбинация

Фигура 11 и фигура 12 показват промяната на функцията на пригодност при промяна на размера на популациите. Забелязва се, че при по-голям размер на популацията по-бързо се достига сходимост на функцията.



Фиг. 11. Стойност на функцията на пригодност при nGen=10; nPop=6; Pm=0,2; Pc=0,75



Фиг. 12. Стойност на функцията на пригодност при nGen=10; nPop=20; Pm=0,2; Pc=0,75

Заключение

Генетичните алгоритми успешно се прилагат върху задачи от различни сфери. Те работят върху голямо пространство от решения, сравнително бързо и надеждно решават сложни задачи и е възможно взаимодействие със съществуващи симулации и модели. Генетичните алгоритми предлагат гъвкавост, която ги прави подходящи при оптимизирането на размити системи и разработване на системи за взимане на решения, свързани с диагностика, наблюдение и управление.

Направените експерименти показват, че разликата на търсената стойност на целевата функция преди и след оптимизацията е с точност до 10^{-4}, което доказва, че създаденият генетичен алгоритъм оптимизира параметрите на размитата система. Предстоят още изследвания върху поведението на генетичния алгоритъм и неговите оператори – селекция, рекомбинация и мутация, които да оптимизират компонентите на размитата система.

References

[1] ГЕОРГИЕВА П., Изследване на модели на софт компютинг за управление в реално време., Академик Дринов, София, 2013г.

[2] BONISSONE P., T. CHEN, K. GOEBEL, Hybrid Soft Computing Systems: Industrial and Commercial Applications., IEEE Proceedings, Special Issue on Computational Intelligence, 1999, vol. 87

[3] GEORGIEVA P., I. POPCHEV, Application of Q-measure in Real Time Fuzzy System for Managing Financial Assets, International Journal on Soft Computing (IJSC) Vol.3, No.4, November 2012

[4] GEORGIEVA P., IVAN POPCHEV, Fuzzy Logic Q-measure Model for Managing Financial Investments, COMPTES RENDUS DE L ACADEMIE BULGARE DES SCIENCES, vol. 66(5), 2013

[5] KENDEL A., Fuzzy Expert Systems, Florida, USA: CRC Press, 1991

[6] KONAR A., Artificial Intelligence and Soft Computing Behavioral and Cognitive Modeling of the Human brain, CRC Press, 2000

[7] LEE M.A., H. TAGAKI, Dynamic control of genetic algorithm using fuzzy logic techniques, in Proc.Fifth Int. Conf. on Genetic Algorithms, pp. 76-83. Morgan Kaufmann, CA. 1993.

[8] SURMANN H., A. KANSTEIN, AND K. GOSER, Self-Organizing and Genetic Algorithms for an Automatic Design of Fuzzy Control and Decision Systems, in Proc. EUFIT'93, pp. 1097-1104, Aachen, Germany, 1993

[9] YE Z., L.GU, A Fuzzy System for Trading the Shanghai Stock Market in Trading on the Edge, Neural, Genetic, and Fuzzy Systems for Chaotic Financial Markets, G.J Deboeck, Ed. New York: Wiley, 1994

[10] HERRERA F., Ten Lectures on Genetic Fuzzy Systems, URL: http://www.researchgate.net/profile/Ulrich_Bodenhofer/publication/2453876_Ten_Lectures_on_Genetic_Fuzzy_Systems/links/09e4150bde4e4a190f000000.pdf