DURAĞAN ZAMAN SERİLERİNDE UYGUN ARMA MODELİNİN GENETİK ALGORİTMALAR İLE BULUNMASI VE İMKB VERİLERİ ÜZERİNE BİR UYGULAMA

Ar. Gör. Mehmet Hakan SATMAN *

Özet

Genetik ve doğal seçilim ilkesini büyük ölçüde taklit eden genetik algoritmalar, optimizasyon süreçlerinde ve hesaplama yükünün çok fazla olduğu durumlarda başarıyla uygulanma potansiyeline sahiptir. Bu çalışmada uygun ARMA modelinin seçiminde ancak tüm mümkün regresyonların taratılmasıyla elde edilebilecek sonuçlara yakın bir sonucun genetik algoritmalar ile bulunması olanakları sunulmuştur. Yöntemi uygulamak için Eviews 5.0 'da bir program yazılmış ve İMKB endeks verilerine uygulanmıştır.

Anahtar Kelimeler: Genel algoritmalar, değişken seçme, ARMA modelleri.

Abstract

Genetic algorithms that mimic the genetics and the principles of natural selection can be applied for the optimization processes and for the computationally intensive problems. In this paper we show that the genetic algorithms can be used to select the best ARMA model. It would be expected that the result of the proposed method can be similar to that of the process of all-possible-regressions which inherently requires too much computation time.

Key Words: Genel algorithms, variable selection, ARMA models.

* İstanbul Üniversitesi İktisat Fakültesi Ekonometri Bölümü.

1. Giriş

Genetik algoritma, genetik ve doğal seçilim ilkelerine dayalı bir optimizasyon tekniğidir (Haupt R.,L., Haupt S.,E., 2004: 22). Genetik algoritmalar, Darwin 'in Türlerin Kökeni (The Origin of Species) adlı eserinde en iyi uyum sağlayanın yaşaması ifadesiyle ünlenen genetik süreçlere, özellikle de doğal seçilim ilkesine dayanmaktadır (Dawid, 1999: 41). Genetik algoritmalar, özetle, bir optimizasyon probleminin çözümü için rasgele belirlenen aday çözümlerin genetik operatörlere tabi tutulması ve doğal seçilim ilkesine göre amaç fonksiyonuna iyi uyum sağlayanların bir sonraki iterasyonda yeni aday çözümler oluşturması şeklinde tanımlanır.

Durağan bir zaman serisinde en uygun *ARMA(p, q)* sürecinin tespit edilmesi bir optimizasyon sorunu olarak görülebilir. Her ne kadar zaman serisine ait korelogramın incelenmesi uygun AR ve MA terimlerinin seçilmesinde bir belirleyici olsa da, AR ve MA terimlerinin birlikte yer aldığı modellerde bu uygulama yeterli olamayabilmektedir (Ong, Huang, Tzeng, 2005). Balcombe (2005) ABD gelir-tüketim ilişkisini araştırmak için kurduğu otoregresif gecikmesi dağıtılmış modelde uygun gecikme terimlerini bulmak için, Ong ve ark. (2005) uygun ARIMA modelinin genetik algoritmalar ile bulunması problemini bilgisayar bellek fiyatlarının tahmini için, Hasheminia ve ark. (2006) en uygun ekonometrik modelin belirlenmesinde ve bu modelin içereceği değişkenlerin seçiminde genetik algoritma kullanmışlardır.

Bu çalışmanın ikinci bölümünde genetik algoritmaların temel unsurları anlatılacak, genetik algoritmaların nasıl çalıştığı bir örnek ile gösterilecektir. Üçüncü bölümde genetik algoritmaların değişken seçme işleminde nasıl kullanılacağı, dördüncü bölümde, yazılan EViews 5.0 programı ve nasıl çalıştırılacağı, beşinci bölümde ise İMKB endeks verilerinden hesaplanan getiri oranları için en uygun ARMA modellerinin belirlenmesi incelenecektir.

2. Genetik Algoritmalar

Optimizasyon problemlerinin çözümünde kullanılan tekniklerden biri olan genetik algoritmalar, canlılardaki doğal seçilim sürecinin taklit edilmesiyle oluşturulmuş yöntemlerdir. Genetik algoritmaların diğer optimizasyon

tekniklerinden farkı, bir topluluğa (population) dayanması, amaç fonksiyonunun (goal function) türevlenebilir olmasını gerektirmemesi ve yerel optimum çözümlere takılmadan global optimum çözümü bulabilmesidir.

Teknik olarak genetik algoritmalar, bir problemin aday çözümlerinin kromozomlarla (chromosome) ifade edilip, bu kromozomların amaç fonksiyonunu sağlamadaki başarısına göre (fitness) sahip olduğu genlerin (gene) bir sonraki nesle aktarılması şeklinde kurulur. Ortamı belirleyen amaç fonksiyonuna iyi uyum sağlayan genler, bir sonraki topluluğu oluşturma sürecinde rol oynama hakkına sahip olurlar.

Amaç fonksiyonuna iyi uyum sağlayan aday çözümlere ait kromozomlar çaprazlanarak (cross-over) yeni aday çözümler oluşturulur. Bu yeni aday çözümlere döl (offspring) adı verilir. Dölün kromozomlarına ait bir veya birkaç gende yapılan değişiklikler de mutasyon (mutation) adını alır.

Genetik algoritmalarda aday çözümlerin kromozom şeklinde ifadesi, aday çözümlerin ikili sayı sisteminde bir temsili ile gösterilir. Tablo 2.1 'de iki aday çözüm ve bu aday çözümlerin kromozom gösterimleri verilmiştir. Her ne kadar genetik algoritmalarda genel olarak ikili kodlama yaygın olsa da semboller ve reel sayılarla kodlama da geniş uygulama alanına sahiptir (Mitchell, 1999: 117).

Tablo 2.1: Aday Çözümlerin Kromozomlarla İfade Edilmesi

Aday Çözüm	Kromozom
16	00010000
21	00010101

Tablo 2.1 'de onlu sayı sistemiyle gösterilen aday çözümler, ikili sayı sistemine dönüştürülerek kromozomlar ile ifade edilmiştir. Problemin doğasına göre aday çözümler başka bir sisteme göre de kromozomlara dönüştürülebilir.

Çaprazlama işlemi, iki kromozomun bir noktada kesilip; birinci kromozomun ilk parçası ile ikinci kromozomun ikinci parçasının birleştirilerek yeni bir kromozom oluşturulması şeklinde tanımlanmıştır. Benzer şekilde ikinci kromozomun ilk parçası ile birinci kromozomun ikinci parçası birleştirilerek

ikinci bir kromozom oluşturulur. Tablo 2.2 'de, Tablo 2.1 'deki kromozomların çaprazlama süreci gösterilmiştir.

Tablo 2.2: Tek Noktadan Çaprazlama İşlemi

Aday Çözüm	Kromozom	Çaprazlama Noktası	Döl	Yeni Aday Çözüm
16	00010000	000100 | 00	00010001	17
21	00010101	000101 | 01	00010100	20

Çaprazlama işlemi rasgele bir çaprazlama noktasından yapılabileceği gibi, problemin doğasına göre başka bir sistematiğe göre de yapılabilir. Örneğin, iki çaprazlama noktası belirlenip Tablo 2.3 'teki gibi bir çaprazlama yapılabilir.

Tablo 2.3: İki noktadan çaprazlama işlemi

Aday Çözüm	Kromozom	Çaprazlama Noktası	Döl	Yeni Aday Çözüm
16	00010000	0001 | 00 | 00	00010100	20
21	00010101	0001 | 01 | 01	00010001	17

Mutasyon işlemi ise kromozoma ait herhangi bir genin değerinin ters çevrilmesi işlemidir. Yani rasgele seçilen bir gen 1 ise 0 'a, 0 ise 1 'e çevrilirse mutasyon işlemi gerçekleştirilmiş olur. Tablo 2.4 'te Tablo 2.3 'teki kromozomlar üzerinde uygulanan mutasyon işlemi gösterilmiştir.

Tablo 2.4: Mutasyon İşlemi

Aday Çözüm	Kromozom	Mutasyon Noktası	Mutasyon	Döl
20	00010100	0̲0010100	10010100	148
17	00010001	0001̲0001	00000001	1

Tablo 2.3 ve 2.4 üzerinden konuşmak gerekirse; mutasyon işlemi, mutasyon işleminin uygulandığı noktaya bağlı olarak döl kromozomda büyük değişikliklere sebep olmuştur. Böylece tek bir genin değerinin değişmesi çözüm

uzayında bir bölgeden diğer bir bölgeye geçişi sağlayabilmektedir. İki kromozomun çaprazlanması sonucu oluşan döller ebeveyn kromozomlar civarında değer almışlardır. Bu bağlamda, çaprazlama işlemi ebeveyn kromozomların aldığı değerler civarında çözüm uzayını tararken, mutasyon işlemi döl kromozomları ebeveyn kromozomların değerlerinden uzaklaştırarak yerel optimum noktadan kurtulup, global optimum noktasının bulunmasına katkıda bulunur[1]. Kodlamanın yapıldığı sistematiğe göre çaprazlama ve mutasyonun etkileri değişebilir.

Seçilim (selection) işlemi ise amaç fonksiyonuna daha iyi uyum sağlayan çözümlerin yeni toplulukta soyunu idame ettirmede yüksek olasılığa sahip olma ilkesine dayanır. Örneğin, amaç fonksiyonu $y = -x^2$, $x \geq 0$ fonksiyonunu en büyükleyen değeri bulmak olsun. Tablo 2.5'te n = 4 elemanlı rasgele bir topluluk ve bu topluluğu oluşturan kromozomların amaç fonksiyonuna uygunluğu verilmiştir. Burada uygunluk için, f (Aday) değerlerine 81 değeri eklenerek, işaretlerin pozitif olması sağlanmıştır. Uygunluk problemin doğasına göre farklı şekillerde de oluşturulabilir.

Tablo 2.5: Uygunluk Değerlerinin Hesaplanması

x = Aday	f(Aday)	U = 81 + f(Aday)	Olasılık	Birikimli Olasılık
2	- 4	77	0,336	0,336
1	- 1	80	0,349	0,685
3	- 9	72	0,314	1
9	- 81	0	0	1

Tablo 2.5 'te görüldüğü gibi fonksiyonunu en büyüklemede başarısız olan x = 9 değerinin, seçilecek yeni toplulukta hayatını sürdürme olasılığı sıfır olarak hesaplanmıştır. x = 1 değeri ise en iyi uyum sağlayan aday çözüm olduğu için soyunu sürdürme olasılığı 0,349 olarak bulunmuştur.

Doğal seçilim işlemi, topluluktaki en kötü aday çözümün yeni toplulukta yaşatılmaması ve en iyi aday çözümün yeni topluluğa doğrudan kopyalanması

[1] Kromozom kodlamaları, çaprazlama ve mutasyon işlemlerinin şema teoremi altında daha geniş açıklamaları için John Holland 'ın 1975 tarihli kitabı incelenebilir.

ve yeni topluluğun nüfusu n olana kadar aday çözümlerin çaprazlanıp mutasyonlanması ile yapılabilir.

3. Değişkenlerin Seçimi

Teorinin ekonometrik model hakkında belirleyici olmadığı durumda; deneme yanılma yoluyla uygun model aranabilir. Durağan bir zaman serisinde uygun AR ve MA terimlerinin belirlenebilmesi için sırasıyla kısmi otokorelasyon (PACF) ve otokorelasyon (ACF) fonksiyonlarına bakılabilir. AR ve MA terimlerinin birlikte yer aldığı modellerde korelogram yeterli bilgi veremeyebilmekte ve uygun terimlerin seçimi araştırmacının kişisel yargılarına dayanabilmektedir (Ong, Huang, Tzeng, 2005).

Araştırmacının uygun olduğunu düşündüğü modeller içinden AIC (Akaike Information Criterian) veya BIC (Schwarz Criterian) kriterlerini en küçükleyen model nihai model olarak seçilebilir. Bu nedenle değişken seçme aşaması bir optimizasyon sorunu olarak düşünülebilir.

Durağan bir zaman serisi için bir *ARMA(p, q)* süreci Eşitlik 3.1'deki gibi gösterilebilir.

$$y_t = \alpha + \beta_1 y_{t-1} + \beta_2 y_{t-2} + ... + \beta_p y_{t-p} + \gamma_1 \varepsilon_{t-1} + \gamma_2 \varepsilon_{t-2} + ... + \gamma_q \varepsilon_{t-q} + \varepsilon_t \qquad (3.i)$$

$p_1 < p$ ve $q_1 < q$ için her $ARMA(p_1, q_1)$ modeli, Eşitlik 3.1 'deki model tarafından yuvalanır. Seçilebilecek her AR ve MA terimi için toplam $2^{p+q} - 1$ adet alt model elde edilebilecektir. Örneğin AR gecikmesi en fazla 10, MA gecikmesi ise en fazla 5 olan toplam $2^{10+5} - 1 = 32.767$ model kurulabilir. Böyle bir sınırlama ile kurulabilecek tüm mümkün modellerin her birinin hesaplanması 3 'er saniye sürdüğü durumda, toplam işlem süresi 27,3 saat olacaktır. Küçük bir sınırlama artışı ile işlem süreleri üstel olarak artacağından AR gecikmesi 11, MA gecikmesi ise 5 ile sınırlandırılmış olan tüm mümkün modellerin sayısı $2^{11+5} - 1 = 65.535$ olacak, yine her bir modelin hesaplanması 3 'er saniye sürdüğü durumda toplam hesap süresi 54,6 saat sürecektir. Benzer olarak da 15 'er gecikme kısıtlaması yapılmış bir ARMA modelinin tüm

mümkün alt kümelerinin sayısı 1.073.741.823 yapar ki, bu modellerin çözümü aynı hesapla 102 yıl sürecektir. İşlem hızını on katına çıkartacak etkin bir algoritmanın bulunmasının dahi böyle bir kombinasyonal problemin çözümü için yeterince uygun olmadığı açıktır.

Bu çalışmanın asıl konusu bir yapay zeka optimizasyon tekniği olan genetik algoritmaların böyle bir kombinasyonal sorunda nasıl kullanılacağı olacaktır. Genetik algoritma tasarımı aday çözümlerin kromozomlarla nasıl ifade edileceğinin belirlenmesiyle başlar. Bu çalışmada kromozom kodlaması Tablo 3.1 'deki gibi bir sistemle yapılacaktır (Ong, Huaıig, Tzeng, 2005).

Tablo 3.1: ARMA Modelinin Kromozomlarla İfadesi

Model	Kodiama	Kromozom
$y_t = \alpha + \beta_1 y_{t-1} + \beta_2 y_{t-3} + \beta_3 y_{t-5} + \gamma_1 \varepsilon_{t-2} + \gamma_2 \varepsilon_{t-3} + \varepsilon_t$	ARMA ((1,3,5) , (2,3))	1010101100
$y_t = \alpha + \beta_1 y_{t-1} + \beta_2 y_{t-4} + \gamma_1 \varepsilon_{t-4} + \varepsilon_t$	ARMA ((1,4) , (4))	1001000010

Tablo 3.1 'deki modellerde hem AR hem de MA terimleri için 5 kısıtlaması getirilmiştir. Tablodaki kromozomların her ikisinin de ilk geninin 1 değerini almış olması, AR(1) teriminin her iki modelde de yer alacağını göstermektedir. Benzer şekilde ilk modelin 9. geninin 0 değerini almış olması bu modelde MA(4) teriminin yer almayacağını gösterirken, ikinci modelin 9. geninin 1 değerini almış olması bu modelde MA(4) teriminin yer alacağım göstermektedir. Bu tür arama problemlerinde 1 ve 0 'lar nümerik değerden çok bir değişkenin varlığı veya yokluğuna ilişkin bir kodlamadır (Back, Fogei, Michalewicz, 2000: 9). Tablo 3.1'deki iki aday modelin kromozomlarının rasgele bir noktadan çaprazlanması ve oluşan döl modeller Tablo 3.2'de gösterilmiştir. Oluşan yeni modellere ait kromozomlara herhangi bir noktadan uygulanan mutasyon işlemi de Tablo 3.3 'te gösterilmiştir.

Tablo 3.2: ARMA Modellerine Ait Kromozomların Çaprazlanması

Model	Kromozom	Döl	Yeni Model
ARMA((1,3,5) , (2,3))	101010 | 1100	1010100010	ARMA((1,3,5) , (4))
ARMA ((1,4) , (4))	100100 | 0010	1001001100	ARMA((1,4) , (2,3))

Tablo 3.3: ARMA Modellerine Ait Kromozomların Mutasyona Uğratılması

Model	Kromozom	Döl	Yeni Model
ARMA((1,3,5) , (4))	1010100010	1010100110	ARMA((1,3,5) , (3,4))
ARMA ((1,4) , (2,3))	1001001100	0001001100	ARMA((4) , (2,3))

4. Uygulamada Kullanılacak EViews 5.0 Kodu

Bu bölümde tanıtılan programda, çaprazlanmış ve mutasyona uğratılmış kromozomlara ait ARMA modellerinden hesaplanan AIC değerlerine göre seçilim işlemi uygulanacak, AIC değerini en küçükleyen aday modeller yüksek olasılıklarla bir sonraki topluluklarda soyunu idame etme şansına sahip olacaktır.

Uygun ARMA modellerinin genetik algoritmayla tahmini için EViews 5.0 'da bir program hazırlanmıştır. Kurulacak modellerde AR ve MA terimleri en fazla 10 olacak şekilde ayarlanmıştır. Programda ***scalar maxARMA=20*** olan satır değeri değiştirilerek, bu sınır daraltılıp genişletebilir. Örneğin AR ve MA terimlerinin en fazla 15 olması isteniyorsa ilgili satır ***scalar maxARMA=30*** olarak değiştirilmelidir. Genetik algoritmanın topluluk nüfusu uygulamada 50 olarak belirlenmiştir. Bazı çalışmalarda, problemin doğasına bağlı olarak bu değer 30, 50, 100 veya 150 olarak değiştirilebilir. Yine de bu sayının ne olabileceği konusunda formel bir kural yoktur. Programda topluluk büyüklüğünü değiştirmek için ***scalar populationSize=50*** satırında gerekli düzeltmeler yapılabilir. Bir önceki topluluktan bir sonraki topluluğa en iyi kaç aday çözümün kopyalanacağı bilgisini değiştirmek için ***scalar crossOverCount=30*** satırı değiştirilmelidir. Bu sayının 30 olması demek, bir önceki topluluktaki 20 en iyi aday çözümün bir sonraki topluluğa kopyalanması, 30 aday çözümün ise çaprazlama sonucu oluşturulacağı anlamına gelir. Oluşan yeni topluluktaki kaç genin mutasyona uğratılacağı ise ***scalar mutationCount=6*** satırında verilmiştir. Açılan çalışma dosyasında ARMA yapısı bulunacak olan

zaman serisi *y* olarak adlandırılıp, program çalıştırıldığında ***reporttable*** adında bir tablo oluşturulmaktadır. Bu tablo içinde AIC kriterini en küçükleyen ARMA modelini barındırır. Model kopyala-yapıştır ile ***Quick-Estimate Equation*** menüstinden tahmin edilebilir. Eğer genetik algoritmanın bulduğu model bazı yönlerden tatmin edici değilse ***population*** adlı matris incelenerek topluluğun içerdiği diğer modeller de denenebilir. Program, AIC kriterini en küçükleyen AR ve MA terimlerini aramaktadır. **fitness(!i,1)=genetic.@aic** olan satır **fitness(!i,1)=genetic.@schwarz** şeklinde değiştirilirse program BIC kriterini en küçükleyen AR ve MA terimlerini arayacaktır.

5. Uygulama

Bu bölümde İMKB Ulusal-Tüm Endeksi, Ulusal-100 Endeksi, Ulusal-50 Endeksi, Ulusal- 30 Endeksi, İkinci Ulusal Pazar Endeksi, Ulusal Sınai Endeksi, Ulusal Hizmetler Endeksi, Ulusal Mali Endeksi, Ulusal Teknoloji Endeksi ve Menkul Kıymetler Yatırım Ortaklıkları Endeksi günlük kapanış verileri kullanılarak, bu verilerden hesaplanan getiri oranlarının bir *ARMA* sürecine uygun olup olmadıkları araştırılacaktır. *ARMA* süreçlerindeki gecikme terimlerinin belirlenmesi sübjektif yargılara bağlı olarak değil, genetik algoritma ile AIC değerinin en küçüklenmesi yöntemiyle gerçekleştirilecektir. Uygulamada kullanılan veriler ve gözlem aralıkları Tablo 5.1'de gösterilmiştir.

Tablo 5.1. Uygulamada Kullanılan Veriler ve Gözlem Aralıkları

Kod	Endeks	Gözlem Aralığı
U-TUM	İMKB Ulusal-Tüm Endeksi	06.01.1997 – 26.04.2006
U-100	İMKB Ulusal-100 Endeksi	06.01.1997 – 26.04.2006
U-50	İMKB Ulusal-50 Endeksi	21.04.2004 – 26.04.2006
U-30	İMKB Ulusal-30 Endeksi	06.01.1997 – 26.04.2006
İkiu	İMKB İkinci Ulusal Pazar Endeks	21.04.2004 – 26.04.2006
Sind	İMKB Ulusal Sınai Endeks	06.01.1997 – 26.04.2006
Hiz	İMKB Ulusal Hizmetler Endeksi	06.01.1997 – 26.04.2006
Mald	İMKB Ulusal Mali Endeks	06.01.1997 – 26.04.2006
Utek	İMKB Ulusal Teknoloji Endeksi	21.04.2004 – 26.04.2006
Yort	İMKB Menkul Kıymetler Yatırım Ortaklıkları Endeksi	06.01.1997 – 26.04.2006

anlamlıdır ve AIC kriteri de bir önceki modelden daha küçük hesaplanmıştır. Benzer problemlerde aynı sorunun yaşanmaması için birkaç deneme-yanılma ile amaç fonksiyonunun global optimumu bulunabilir. Bu uygulamada algoritma bir kaç kez daha çalıştırılmış, fakat Tablo 5.4 'teki modelin AIC değerinden daha küçük bir değer bulunmamıştır.

Tablo 5.4: U-50 Modelinin Yeniden Tahmin Edilmesi

Yöntem: EKK
Gözlem (düzeltilmiş): 10 507
Dahil edilmiş gözlem: 498 (düzeltme sonrası)

Değişken	Parametre tahmini	Standart Hata	*t*-istatistik	P-değeri
C	0.001932	0.000583	3.314680	0.0010
AR(2)	-0.090871	0.021768	-4.174568	0.0000
AR(3)	-1.079732	0.022929	-47.08939	0.0000
AR(9)	0.407194	0.024409	16.68235	0.0000
MA(1)	0.038452	0.014949	2.572286	0.0104
MA(3)	1.182091	0.015242	77.55561	0.0000
MA(5)	-0.121154	0.024557	-4.933651	0.0000
MA(7)	-0.060828	0.012834	-4.739496	0.0000
MA(8)	-0.050684	0.022264	-2.276546	0.0232
MA(9)	-0.507271	0.018440	-27.50888	0.0000

R-kare	0.101571	Bağımlı değişkenin ortalaması	0.001801
Düzeltilmiş R-kare	0.085001	Bağımlı değişkenin standart sapması	0.016150
Regresyon standart hatası	0.015448	Akaike bilgi kriteri (AIC)	-5.482773
Kalıntı kare toplamı	0.116460	Schwarz kriteri (BIC)	-5.398222
Log likelihood	1375.210	F-istatistiği	6.130031
Durbin-Watson	1.954332	F-istatistiğinin P-değeri	0.000000

Bir diğer yöntem de, algoritmanın bir kez çalıştırılıp AIC değerini en küçükleyen modeli bulmak ve sonra anlamsız değişkenleri modelden çıkarmak olabilir. Çünkü tek başına AIC değerinin en küçüklenmesi model kurma

açısından yeterli değildir. Uygun bir modelin bulunması, bazı istatistiksel özelliklerin ve teorinin belirlediği unsurların sağlanması ile olmaktadır. Bu yüzden birden fazla amaç fonksiyonunun sağlandığı bir model, uygun model olarak seçilebilir. Çok amaç fonksiyonlu genetik algoritmalar (Multi-objective genetic algorithms – MOGA) ile değişken seçme işlemi sonraki çalışmaların konusu olabilir.

6. Sonuç

Beşinci bölümde, bazı İMKB endekslerinden hesaplanan getiri oranları için uygun bir ARMA modelinin olup olmadığı genetik algoritmalar ile araştırılmıştır. İMKB Ulusal-50 Endeksi hariç diğer tüm endeksler için hesaplanan getiri oranları normal dağılmadığından, bunların doğrusal birer kombinasyonu olan tahmin edilmiş modelin kalıntıları da normal dağılıma uymamaktadır. Bu bağlamda hesaplanan t ve F istatistikleri artık t ve F dağılımına uymazlar. Bu nedenle bu modeller için sadece AIC kriterini en küçükleyen AR ve MA terimleri raporlanmıştır. Klasik olmayan başka bir yöntemle uygun gecikmeler kullanılarak tahmin yapılabilir. İMKB Ulusal-50 Endeksi getiri oranları için kurulan modelin t ve F istatistiklerine güvenilebilir. Bu bağlamda genetik algoritma ile AIC kriterinin en küçüklenmesiyle bulunmuş model uygun modeldir. Ek-1 'de verilen E-Views 5.0 kodu kullanılarak bu yöntem farklı zaman serileri için de uygulanabilir. Model kurma işlemi her zaman tek bir amaç fonksiyonunun en küçüklenmesi yoluyla mekanik hale getirilememektedir. Bağımsız değişkenlerden bağımlı değişkene doğru tek yönlü bir ilişki olması, bağımsız değişkenlerin ilişkisiz olması, kalıntıların normal dağılması ve tahmin için gereken klasik bazı varsayımların gerçekleşiyor olması, model kurma aşamasında sağlanması gereken amaç fonksiyonları gibi düşünülebilir. Dolayısıyla çok amaç fonksiyonlu genetik algoritmalarla (Multi-objective genetic algorithms) model kurma sonraki çalışmaların konusu olabilir. Öte yandan salt istatistiksel bir takım özelliklerin sağlanıyor olmasının yanında kurulan modelin teori ile uyumu gözden geçirilmelidir. İMKB Ulusal-50 Endeksi getiri oranlarını açıklayan uygun AR ve MA terimlerinin bulunması konusunda teorinin koyduğu kesin sınırlamalar olmadığından, mekanik bir arama sürecine başvurmak mümkün olabilmiştir.

KAYNAKÇA

Back, T., Fogel, D.B., Michalewicz, Z. (2000): Evolutionary Computation 2 - Advanced algorithms and Operators, Institute of Physics Publishing, Bristol and Philadelphia.

Balcombe, K.G. (2005): "Model Selection Using Information Criteria and Genetic Algorithms", Computational Economics, 25, p. 207-228.

Dawid, H. (1999): Adaptive Learning by Genetic Algorithms – Analytical Results and Applications to Economic Models, Springer, New York.

Hasheminia, H., Niaki, S.T.A. (2006): "A genetic algorithm approach to find the best regression/econometric model among the candidates", Applied Mathematics and Computation, 183, p. 337-349.

Haupt, R.L., Haupt, S.E. (2004): Practical Genetic Algorithms, Second Edition, John Wiley & Sons Inc., Canada.

Holland, J.H. (1975): Adaptation in Natural and Artificial Systems, University of Michigan Press. (Second Edition: MIT Press, 1992).

Mitchell, M. (1999), An Introduction to Genetic Algorithms, MIT Press. 1999. England.

Ong, C.S., Huang, J.J., Tzeng, G.H. (2005): "Model identification of ARIMA family using genetic algorithms", Applied Mathematics and Computation, 164, p. 885-912.

Ek-1. Eviews 5.0 Kodu

```
'Genetic Algorithm for ARMA Model Selection
'Defining Main Variables
scalar populationSize=50
scalar maxARMA=20
scalar crossOverCount=30
scalar mutationCount=6
matrix(populationSize,maxArma) population
matrix(populationSize,maxArma) newpopulation
rowvector(maxArma) Chromosome1
rowvector(maxArma) Chromosome2
rowvector(maxArma) Chromosome3
rowvector(maxArma) Chromosome4
matrix(populationSize,1) fitness
table ReportTable
'Random Population Creation
subroutine RandomizePopulation
for !i=1 to populationSize
for !j=1 to maxARMA
population(!i,!j)=@round(@rnd)
population(!i,!j)=0
next
next
endsub
'Calculating Fitness Values
subroutine CalculateFitness
for !i=1 to populationSize
%s="y c "
for !j=1 to maxARMA/2
if population(!i,!j)=1 then
%s=%s+" AR("+@str(!j)+") "
endif
next
```

```
!m=1
for !j=(maxARMA/2)+1 to maxARMA
if population(!i,!j)=1 then
%s=%s+" MA("+@str(!m)+") "
endif
!m=!m+1
next
equation genetic.ls {%s}
fitness(!i,1)=genetic.@aic
next
endsub
'Sort Population by Fitness
subroutine SortPopulation
for !i=1 to populationSize
for !j=!i to populationSize
if fitness(!i,1)>fitness(!j,1) then
!c=fitness(!i,1)
fitness(!i,1)=fitness(!j,1)
fitness(!j,1)=!c
chromosome1=@rowextract(population,!i)
chromosome2=@rowextract(population,!j)
rowplace(population,chromosome1,!j)
rowplace(population,chromosome2,!i)
endif
next
next
endsub
'Selection Operator
subroutine EliteSelection
for !i=1 to populationsize-crossovercount
chromosome1=@rowextract(population,!i)
rowplace(newpopulation,chromosome1,!i)
next
for !i=(populationsize-crossovercount)+1 to populationsize step 2
```

```
!r1=@round(@rnd*(populationsize-1))+1
!r2=@round(@rnd*(populationsize-1))+1
chromosome1=@rowextract(population,!r1)
chromosome2=@rowextract(population,!r2)
call crossover(chromosome1,chromosome2)
rowplace(newpopulation,chromosome3,!i)
rowplace(newpopulation,chromosome4,!i-1)
next
endsub
subroutine crossover (rowvector v1,rowvector v2)
!rr1=@round(@rnd*(maxarma-1)+1)
for !ij=1 to !rr1
chromosome3(!ij)=v1(!ij)
chromosome4(!ij)=v2(!ij)
next
for !ij=!rr1+1 to maxARMA
chromosome3(!ij)=v2(!ij)
chromosome4(!ij)=v1(!ij)
next
endsub
subroutine mutation
for !mutiter=1 to mutationCount
!rr1=@round(@rnd*(populationsize-2))+2
!cc1=@round(@rnd*(maxarma-1)+1)
if newpopulation(!rr1,!cc1)=0 then
        newpopulation(!rr1,!cc1)=1
        else
        newpopulation(!rr1,!ce1)=0
endif
next
endsub
'Report
subroutine report
call SortPopulation
```

```
%s="y c "
for !j=1 to maxARMA/2
if population(1,!j)=1 then
%s=%s+" AR("+@str(!j)+") "
endif
next
!m=1
for !j=(maxARMA/2)+1 to maxARMA
if population(1,!j)=1 then
%s=%s+" MA("+@str(!m)+") "
endif
!m=!m+1
next
reporttable(1,1)=%s
endsub

'Main Algorithm
subroutine Main
        call RandomizePopulation
        for !iter=1 to 50
        call Calculatefitness
        call SortPopulation
        call EliteSelection
        call Mutation
        population=newpopulation
        next
        call report
endsub

call Main
```