УДК 336.767

ПРОГРАММНАЯ РЕАЛИЗАЦИЯ РОБАСТНОГО АЛГОРИТМА ОЦЕНКИ ВОЛАТИЛЬНОСТИ ФИНАНСОВЫХ ВРЕМЕННЫХ РЯДОВ В РАМКАХ МОДЕЛИ ОБОБЩЕННОЙ АВТОРЕГРЕССИОННОЙ УСЛОВНОЙ ГЕТЕРОСКЕДАТИЧНОСТИ GARCH (1,1)

Маглеванный Илья Иванович[1], д-р физ.-мат. наук, проф.
Цаплина Мария Георгиевна[2], асп.

[1]Волгоградский государственный социально-педагогический университет, пр-т им. В.И. Ленина, 27, Волгоград, Россия, 400131; e-mail: sianko@list.ru

[2]Волгоградский государственный университет, пр-т Университетский, 100, Волгоград, Россия, 400062; e-mail: mlapenkova@gmail.ru

Цель: статья посвящена вопросам оценивания и прогнозирования волатильности с помощью линейной модели GARCH (1,1) с учетом машинно-зависимых аспектов. *Обсуждение*: различные программные пакеты, обладая различными численными методами оптимизации, могут приводить к отличающимся оценкам параметров при одних и тех же исходных данных. Решение данной проблемы представляется авторами в виде создания универсального открытого программного кода с учетом его машинно-зависимых и проблемно-зависимых аспектов. *Результаты*: предложенный авторами эвристический робастный алгоритм численной оценки параметров линейной модели GARCH (1,1) может быть легко реализован на любом языке программирования, не требует сторонних математических процедур и выбора начальных значений параметров для построения модели волатильности. В качестве примера в работе исследованы статистические характеристики дневной доходности энергетического индекса РСВ и построена эконометрическая модель дневной доходности индекса для модели семейства GARCH (1,1).

Ключевые слова: моделирование волатильности, эконометрические расчеты с учетом машинной арифметики, модель GARCH (1,1).

DOI: 10.17308/meps.2015.1/66

1. Введение

Рост финансовых рынков по всему миру способствует развитию математических моделей, адекватно описывающих финансовые ряды. Традиционные модели временных рядов, такие как модель ARIMA, не могут адекватно учесть все характеристики, которыми обладают финансовые ряды и

требуют расширения. Множество проведенных исследований выявило ряд специфических особенностей временных рядов доходности финансовых активов и их волатильности – отсутствие автокорреляции, длительная память, кластеризацию волатильности, условную гетероскедастичность и другие. Более подробный обзор этих особенностей можно найти в [5].

Волатильность – это статистический показатель, характеризующий тенденцию рыночной цены или дохода изменяться во времени. На цены активов влияет большое количество факторов: новости, макроэкономические данные, отчеты компаний об итогах их деятельности, оценки стоимости компаний от ведущих инвестиционных фирм. Это приводит к изменчивости доходностей активов и характеристик изменчивости доходности – волатильности. Поскольку этот показатель непосредственно не наблюдается, требуется провести оценку волатильности по наблюдаемым данным.

В последнее время инструментом оценивания волатильности стали эконометрические модели условной гетероскедастичности (ARCH) [12], обобщенной авторегрессионной условной гетероскедастичности (GARCH) [3] и их модификации. Оценки параметров моделей GARCH используют метод максимального правдоподобия и реализованы во многих прикладных пакетах статистической обработки данных [16, 6, 8]. Однако вплоть до настоящего времени особенности численной реализации решения задач данного типа практически не рассматривались в литературе. В результате в различных пакетах представлены различные программные реализации, с различными численными методами оптимизации, параметрами, определяемыми по умолчанию, что приводит к различным оценкам параметров модели при одних и тех же исходных данных. Данный факт был отмечен в работах [5, 12], где были протестированы различные доступные проприетарные программные пакеты и сформулированы некоторые соображения относительно причин возможных расхождений результатов.

Сложившаяся в настоящее время практика эконометрических вычислений предполагает использование программ с закрытым кодом. В результате пользователь имеет дело с «черным ящиком», в котором скрыты все детали программирования и даже исходные допуски.

В данной работе мы рассматриваем еще один существенный аспект алгоритмов решения задачи указанного типа – проблемы, связанные с особенностями машинной арифметики. В результате нами сформулирован робастный алгоритм, предназначенный для оценки параметров моделей GARCH (1,1), и представлена его программная реализация на алгоритмическом языке C++. Под робастностью (надежностью) понимается способность программы успешно решать задачи, на которые она рассчитана. Это определяется в конечном счете тем, решает ли она задачи пользователей, а также возможностью управлять процессом выполнения программы.

2. Модель GARCH (1,1)

Пусть наблюдаемая реализации соответствующего стохастического

процесса представлена временным рядом r_t, $t=0,\ldots, T-1, T$ – число наблюдений. Стандартная спецификация модели GARCH (1,1) имеет вид [3]

$$r_t = \mu + \varepsilon_t, \quad \varepsilon_t|\psi_{t-1} = \sigma_t \xi_t \tag{1}$$

$$\sigma_t^2 = \omega + \alpha(r_t - \mu)^2 + \beta\sigma_{t-1}^2 \tag{2}$$

Здесь Ψ_{t-1} – информационное множество, μ – условное математическое ожидание, σ^2_t – условная дисперсия (волатильность), ε_t, ξ_t – нормально распределенные независимые случайные величины с нулевым математическим ожиданием и единичной дисперсией. Таким образом, прогноз волатильности на следующий период t является смесью волатильности и остаточного квадрата за период предыдущий. Параметры модели $\omega>0$, $\alpha>0$, $\beta>0$, $\alpha+\beta<1$ удовлетворяют условиям [3]:

$$\omega>0, \quad \alpha>0, \quad \beta>0, \quad \alpha+\beta<1 \tag{3}$$

2.1 Оценка параметров методом максимального правдоподобия

В предположении о нормальном распределении функция максимального правдоподобия определяется следующим образом:

$$L(\mu,\omega,\alpha,\beta) = \prod_{t=1}^{T} \frac{1}{\sqrt{2\pi\sigma_t^2}} \exp\left(\frac{(r_t-\mu)^2}{2\sigma_t^2}\right). \tag{4}$$

Принцип максимального правдоподобия сводится к задаче минимизации целевой функции:

$$f(\mu,\omega,\alpha,\beta) = \frac{1}{2}\left[T\ln(2T) + \sum_{t=1}^{T}\left(\ln\sigma_t^2 + \frac{(r_t-\mu)}{\sigma_t^2}\right)\right] \to min. \tag{5}$$

В силу условий (3) задача (5) является нелинейной задачей оптимизации с ограничениями и требует применения адекватных численных методов.

Отметим несколько важных особенностей модели (5). Во-первых, максимизируется не функция правдоподобия, а функция условного правдоподобия, что требует информации о предыстории процесса. Во-вторых, в соответствии с (3), задача (5) является задачей условной минимизации, в то время как большинство апробированных и хорошо зарекомендовавших себя на практике алгоритмов минимизации являются безусловными. В-третьих, модель является частично определенной, т.к. начальные значения ε_{-1} и σ^2_1 не определены. На самом деле эти начальные значения, на что часто не обращается внимание, могут существенно повлиять на «решение», генерируемое данной программой. Все программные средства используют начальную инициализацию по умолчанию, однако не во всех указано, какую именно, и не все позволяют пользователю задавать их самостоятельно – это существенный недостаток [12].

3. Учет машинной арифметики и робастный алгоритм

В этом разделе обсуждаются вопросы оценки машинных программ, а не алгоритмов. Отличие состоит в том, что машинная программа может содержать много деталей, которые имеют решающее значение для ее ра-

ботоспособности, но не являются частью «базового алгоритма». Примерами служат рассмотренные ниже критерии останова. Пользователь машинной программы должен сознавать, что используется конкретная программная реализация базового алгоритма, и что две реализации одного и того же базового алгоритма могут работать совершенно по-разному.

Некоторые элементы вычислительных алгоритмов, такие как проверка на сходимость, зависят от того, насколько точно в ЭВМ представляются действительные числа. Уместно отметить, что понимание машинной арифметики влияет на написание вычислительных программ.

В ЭВМ действительные числа представляются в так называемой экспоненциальной форме с использованием знака, основания системы счисления, порядка и мантиссы. Это представление становится единственным, если потребовать, чтобы 1/основание < мантисса < 1, т.е. первый «десятичный» разряд справа от десятичной точки не должен быть нулевым. Длина мантиссы, называемая точностью представления, особенно важна для численных расчетов. Это представление действительных чисел в ЭВМ называется представлением с плавающей точкой. Будем обозначать через $fl(x)$ представление с плавающей точкой числа x.

Хранение действительных чисел только с конечной точностью имеет серьезные последствия, которые можно легко изложить в краткой форме. Во-первых, коль скоро не всякое действительное число может быть точно представлено в машине, то в лучшем случае можно ожидать, что решение будет верным с машинной точностью. Во-вторых, в зависимости от машины и компилятора результат каждой промежуточной арифметической операции либо округляется, либо усекается до машинной точности. Таким образом, погрешность, связанная с конечной точностью, может накапливаться и еще более понизить точность результатов. Такие ошибки называются ошибками округления. Если постоянно помнить об этих ситуациях при написании и использовании вычислительных программ, то можно понять и избежать многих проблем, связанных с использованием арифметики конечной точности, из-за использования которой, и даже в большей степени из-за итеративной природы алгоритмов, мы не получаем точного ответа в большинстве нелинейных задач. Поэтому важно ввести характеристику машинной точности так, чтобы рассуждения и вычислительные программы могли быть довольно независимыми от любой конкретной машины. Обычно используется понятие машинный эпсилон (machine epsilon) или кратко macheps. Для конкретной вычислительной машины оно определяется как наименьшее положительное число τ, такое, что $1+\tau > 1$.

Величина macheps весьма полезна при рассмотрении машинных чисел. Например, можно легко показать, что относительная ошибка машинного представления $fl(x)$ любого действительного ненулевого числа x меньше, чем macheps, и, наоборот, машинное представление любого действительного числа x будет лежать в пределах $(-macheps + x, +macheps + x)$. Анало-

гично, два числа x и y совпадают в крайней в левой половине их разрядов, когда $|x-y|/|x| \leq macheps$.

В качестве примера рассмотрим важный вопрос о критериях останова итерационного алгоритма решения задачи минимизации, в которых используются машинно-зависимые и проблемно-зависимые точности.

Принимать решение о том, когда остановиться, приходится в некоторой степени неформально. Это решение далеко не всегда безупречно, и все же для него требуется четкий критерий. Критерий принятия решения обычно представляется в таком виде: «Получено ли приближенное решение задачи»? или «Находятся ли результаты последних двух (или нескольких) итераций практически в одном и том же месте?» Первый вопрос представляется проверкой типа: имеет ли место:

$$\frac{|f(x_+)-f(x_c)|}{\max\{|f(x_+)|, typf\}} < ftol, \qquad (6)$$

где x_c и x_+ – соответственно начальная и конечная точка на очередном шаге итерации; $typf$ – типичное значение модуля целевой функции в точке минимума, которое определяется пользователем. Точность $ftol$, выбирается так, чтобы отразить представления пользователя о достаточной близости к решению для данной задачи. Например, $ftol$ может быть положено равным macheps. Эта проверка, конечно, очень чувствительна к масштабу, и поэтому важно, чтобы программа давала пользователю инструкции по выбору $ftol$ или масштаба $typf$ таких, что x_+, удовлетворяющее условию (6), было бы приемлемым решением задачи. Уберечься от чрезмерной ограничительности этого условия помогает второй вопрос, который состоит в проверке соотношения вида: имеет ли место:

$$\frac{|x_+ - x_c|}{\max\{|x_+|, typx\}} < steptol, \qquad (7)$$

где задаваемая пользователем величина $typx$ отражает характерные значения переменной x вблизи точки минимума [1]. Разумной точностью здесь служит $steptol = macheps^{\frac{1}{2}}$, которое соответствует останову всякий раз, когда совпадает левая половина разрядов у x_c и x_+. На практике при решении любой задачи с быстрой сходимостью значение $|f(x_+)-f(x_c)|$ обычно становится малым раньше, чем шаг по x (критерий (6)), а для задач, для которых скорость лишь линейна, первым может стать шаг по (критерий (7)). Таким образом, выбор критериев останова достаточно сложен, особенно для плохо масштабированных задач [1].

Другой взгляд на величину macheps служит ключом к трудному вопросу принятия решения о том, когда в определенном контексте машинное число могло бы также выступать только как нулевое. Нередко в процессе вычислений возникают машинные числа x и y, имеющие настолько различные порядки величин, что $fl(x+y) = fl(x)$. Это означает, что в данном контексте y неотличимо от нуля, и иногда полезно контролировать вычисления и, в самом деле, присвоить y нулевое значение.

И наконец, любому пользователю ЭВМ следует знать о явлениях переполнения и антипереполнения, встречающихся, когда в процессе вычислений возникает ненулевое число, порядок которого больше или соответственно меньше крайних значений диапазона, отведенного в ЭВМ под порядки. Более детально данные вопросы, связанные с разработкой машинно-ориентированных алгоритмов, рассмотрены в монографии [11].

Рассмотрим теперь, как особенности машинной арифметики влияют на расчеты волатильности по модели GARCH(1,1). Одним из достоинств модели GARCH (1,1) принято считать т.н. «бесконечную память» – якобы рекуррентное соотношение (2) учитывает всю предысторию процесса вплоть от момента t до момента $t=0$. Формально-теоретически это так, однако, легко видеть, что в действительности при конкретных вычислениях память модели становится конечной и зависит от программной реализации.

Действительно, соотношение (2) эквивалентно следующему:

$$\sigma_t^2 = \sum_{i=0}^{t-1} [\omega + \alpha(r_{t-i-1} - \mu)^2]\beta^i + \beta^i \sigma_{-1}^2. \qquad (8)$$

При $\beta < 1$ величина β^i экспоненциально убывает, и при $\beta^i < macheps$ очередное слагаемое не изменяет уже накопленной суммы. Поэтому при вычислении суммирование в (8) фактически включает только последние i_{min} отсчетов исходного временного ряда,

$$\sigma_t^2 = \sum_{i=0}^{i_{min}} [\omega + \alpha(r_{t-i-1} - \mu)^2]\beta^i + \beta^i \sigma_{-1}^2, \qquad (9)$$

$$i_{min} \sim \frac{\ln macheps}{\ln \beta}. \qquad (10)$$

Типичными значениями для финансовых рядов являются $\beta<0.8$ [6, 7, 8]. Из данного факта следует, по крайней мере, два вывода.

Во-первых, не имеет практического смысла увеличивать объем выборки $\{r_t\}_{t=0}^{T-1}$ якобы для более точной оценки параметров модели. Число отсчетов следует согласовывать с точностью представления действительных чисел в используемой программе. Например, при расчетах с помощью популярного пакета Matlab по умолчанию используется двойная точность представления чисел, тогда:

$$macheps \approx 2{,}22 \cdot 10^{-16},\ i_{min} \le \frac{\ln 2{,}22 \cdot 10^{-16}}{\ln 0{,}8} \approx 162, \qquad (11)$$

т.е. в типичной ситуации достаточно порядка 200 отсчетов, предшествующих значению времени t, остальная предыстория не влияет на результаты расчетов. Ни одна из известных нам доступных программ эконометрических расчетов не дает пользователю рекомендаций подобного рода.

Во-вторых, величина i_{min} недетерминированным образом зависит от текущего значения параметров μ, ω, α, β. Это означает, что, несмотря на формально детерминированный характер задачи, вычисление целевой функции (5) приводит к ее «зашумленности», обусловленной особенностями программной реализации и техническими характеристиками компьютера.

В результате с вычислительной точки зрения целевая функция становится недифференцируемой. Данное обстоятельство делает проблематичным использование для решения задачи минимизации (5) алгоритмов, основанных на конечно-разностной аппроксимации производных.

Таким образом, вследствие вычислительной сложности GARCH моделей различные программные средства для решения задачи (5) могут привести к различиям в численных результатах при одних и тех же входных данных, как отмечено в [12, 5].

Нами были проведены численные эксперименты с использованием апробированных методов, описанных в [1]. Первый – алгоритм Левенберга-Марквардта в реализации Море [13] представлен в MINPACK [14]. Второй – «полный» алгоритм ньютоновского типа, который использует вторые конечно-разностные производные. Программная реализация представлена кодом NL2SOL [7]. В обоих случаях даже при выборе начального приближения, близкого к оптимальной точке, погрешности конечно-разностной аппроксимации градиента целевой функции уже на первой итерации приводили в «выбросу» очередного приближения за пределы допустимой области и остановке алгоритма.

В связи со слабой надежностью вышеуказанных методов было принято решение разработать собственный программный код решения задачи оценки параметров моделей типа GARCH, основанный на концепции прямого поиска, исключающей какие-либо разностные аппроксимации производных и использующий минимальное число вычислений целевой функции. В качестве такового разработан авторский численный метод, основанный на общей схеме покоординатного спуска [2] (циклический метод релаксации). При этом для одномерной минимизации была разработана модификация известного метода золотого сечения [2], однако наш алгоритм не требует выделения области унимодальности минимизируемой функции, как в [15], и за конечное число шагов всегда находит глобальный минимум на сегменте. Программная реализация алгоритма была осуществлена на языке C++, что позволяет получить быстрый и эффективный исполняемый код.

4. Программный код на C++

Ниже приведен код программы, реализующей модифицированный алгоритм одномерного поиска минимума методом золотого сечения.

```
#include <math.h>
//Одномерная минимизация методом золотого сечения
static inline double amax1(double x,double y){if(x>y) return x;else return y;}
static double mach(void)//Возвращает машинный эпсилон
{double eps=1.,eps1=2;while(eps1!=1){eps/=2.;eps1=1.+eps;}return(2.*eps);}
//Одномерная минимизация на сегменте [a,b]
//Функция возвращает минимальное значение в точке xmin
double GoldSectDim1(double a,double b,//границы сегмента
```

```
double (*func)(double),//указатель на минимизируемую функцию
double steptol,//критерий останова по величине шага
double typx,//характерная величина аргумента в точке минимума
double &xmin//точка минимума, выходной параметр
)
{if(steptol<=0.)steptol=sqrt(mach());typx=(typx!=0.)?fabs(typx):1;
double x[4],f[4],ksi=(3-sqrt(5))/2.,ks=ksi*(b-a),fmin;
x[0]=a;x[3]=b;x[1]=x[0]+ks;x[2]=x[3]-ks;for(int i=0;i<=3;i++)f[i]=func(x[i]);
while(fabs(x[2]-x[1])/amax1(fabs(x[2]),typx)>steptol){xmin=x[0];fmin=f[0];
for(i=1;i<=3;i++)if(f[i]<fmin){xmin=x[i];fmin=f[i];}
if(xmin<=x[1]){x[3]=x[2];f[3]=f[2];}else {x[0]=x[1];x[1]=x[2];f[0]=f[1];f[1]=f[2];}
x[2]=x[0]+x[3]-x[1];f[2]=func(x[2]);if(x[1]>x[2]){double a=x[1];x[1]=x[2];x[2]=a;a=f[
1];f[1]=f[2];f[2]=a;}}//while
xmin=(x[1]+x[2])/2;return func(xmin);}
```

Ниже представлен текст программы покоординатного спуска.

```
//Глобальные переменные
int ivar;int NGlobal;double *xGlobal;
double (*funcGlobal)(int Ndim,double x[]);
static double OneDimensional(double y)
{xGlobal[ivar]=y;return funcGlobal(NGlobal,xGlobal);}
static int umstop(int n,double xc[],double xp[],double typx[],double steptol,
double fc,double fp,double typf,double ftol,double it,double itlim)
{if(it>=itlim)return 3;if(fabs(fp-fc)/amax1(typf,fp)<ftol)return 1;
double stepmax=0.;for(int i=0;i<n;i++)stepmax=amax1(stepmax,fabs(xp[i]-xc[i])/amax
1(fabs(xp[i]),typx[i]));if(stepmax<=steptol)return 2;return 0;}
//Покоординатный спуск
//Функция возвращает значение целевой функции в точке xmin[]
double DirectSearch(int Ndim,//Размерность пространства параметров
//Входные параметры программы
double a[],//Нижние границы в пространстве параметров
double b[],//Верхние границы в пространстве параметров
double (*func)(int Ndim,double x[]),//Указатель на целевую функцию
double steptol,//Критерий останова по величине шага
//параметр, определяющий критерий останова по масштабированному расстоянию
//между двумя последовательными приближениями. Если входное значение равно 0,
//то программа использует steptol=macheps**(1/2).
double ftol,//Критерий останова по изменению значений целевой функции
double itlim,//Допустимое число итераций
double typx[],//Типичные значения аргумента
double typf,//Типичные значения функции
//Выходные параметры программы
```

```
double xmin[],//Точка минимума в пространстве параметров
int &termcode,//Код, указывающий причину останова.
double &it//Число итераций
)
{::NGlobal=Ndim;::funcGlobal=func;if(steptol<=0.)steptol=sqrt(mach());
if(ftol<=0.)ftol=steptol*steptol;if(typf!=0.)typf=fabs(typf);else typf=1.;
for(int i=0;i<Ndim;i++)typx[i]=(typx[i]!=0.)?fabs(typx[i]):1;
double *xp=new double[Ndim],*xc=new double[Ndim],xk,fp,fc;
for(i=0;i<Ndim;i++)xc[i]=0.5*(a[i]+b[i]);//Начальное приближение
fc=funcGlobal(Ndim,xc);termcode=0;it=1;xGlobal=xp;for(i=0;i<Ndim;i++)xp[i]=xc[i];
//Итерации
while(1){for(i=0;i<Ndim;i++){::ivar=i;
fp=GoldSectDim1(a[i],b[i],OneDimensional,0.,typx[i],xk);xp[i]=xk; }
//Проверить условия останова
termcode=umstop(Ndim,xc,xp,typx,steptol,fc,fp,typf,ftol,it,itlim);
if(termcode)break;it+=1;fc=fp;for(i=0;i<Ndim;i++)xc[i]=xp[i];}//end while
for(i=0;i<Ndim;i++)xmin[i]=xp[i];delete xp;delete xc;return fp;}
```

Ниже приведены параметры масштабирования, останова и другие, а также указания по выбору для них подходящих значений.

typx – массив размерности *n*, *i*-я компонента которого есть положительный скаляр, указывающий характерную величину. Важно задать значения *typx*, когда ожидается, что величины *x[i]* будут сильно различаться, в этом случае программа может работать лучше, чем при *typx[i] = 1*, которое предлагается по умолчанию.

steptol – положительный скаляр, задающий диапазон, в котором масштабирование расстояние между двумя последовательными приближениями считается достаточно близким к нулю для того, чтобы остановить алгоритм. Алгоритм останавливается, если:

$$max\left\{\frac{|x_+[i]-x_c[i]|}{\max\{|x_+[i]|,typx[i]|\}}\right\} \leq stepol. \qquad (12)$$

Здесь x_c и x_+ – соответственно начальная и конечная точка на очередном шаге итерации. Значение *steptol* должно мало. Предлагаемое значение по умолчанию: $steptol = macheps^{1/2}$.

itlimit – положительное целое, указывающее максимальное число итераций, которые могут быть выполнены.

Выходные параметры программы: *xmin[n]* – оптимальная точка в пространстве параметров, *termcode* – указывает на условие окончания, которое вызвало остановку алгоритма: *termcode=1* – алгоритм остановлен по условию (6), *termcode=2* – алгоритм остановлен по условию (12), *termcode=3* – превышен лимит на число итераций.

Ниже представлен текст программы, реализующей оценку параметров модели GARCH (1,1).

```
//Глобальные параметры для вычисления целевой функции
int T;//Число отсчетов временного ряда
double *R;//Временной ряд
double aminus1_2=0,sigminus1_2=0;
//Целевая функция
double func(int n,double zz[])
{double mu=zz[0],omega=zz[1],beta=zz[3],alpha=zz[2],aminus1_2,sigminus1_2;
if(n==5)aminus1_2=sigminus1_2=zz[4];else aminus1_2=sigminus1_2=::sigminus1_2;
double sum=T*log(2*M_PI);int t;
double sig2t=omega+alpha*aminus1_2+beta*sigminus1_2,at,addend;
for(t=0;t<T;t++){if(t>0){at=R[t-1]-mu;sig2t=omega+alpha*at*at+beta*sig2t;}
at=R[t]-mu;sum+=log(sig2t)+at*at/sig2t;}return 0.5*sum;}
void main(void)
{int n=4;//Число оцениваемых параметров
double lowerBounds[4]=
{0.001,0.001,0.001,0.001};//Нижние границы пространства параметров
double upperBounds[4]=
{10.,10.,0.999,0.999};//Верхние границы пространства параметров
double steptol=0,ftol=0,itlim=1000;//Допуски
double typx[4]={0,0,0,0};//Типичные значения параметров
double typf=0;//Типичное значение функции
double xnew[4];//Выходной параметр, точка минимума
int termcode;//Код, причина останова
double it;//Число итераций
//Найти оценки параметров модели GARCH (1,1)
double fnew=DirectSearch(n,lowerBounds,upperBounds,
func,steptol,ftol,itlim,typx,typf,xnew,termcode,it);
//На выходе fnew – значение целевой функции в точке минимума xnew[] }
```

Для тестирования программы нами использовался набор данных, представленный в [4] и результаты, полученные в [11], основанные на оптимизации логарифма функции правдоподобия с использованием аналитических выражений для градиента и гессиана. Отметим, что разработанный нами алгоритм не требует ни вычисления производных, ни оценки какого-либо начального приближения, достаточно установить ограничения сверху и снизу для оцениваемых параметров.

5. Численный пример апробации авторского алгоритма

На рис. 1 (слева) представлен график показателя суточной доходности за наблюдаемый период с 1 января 2011 г. по 31 марта 2014 г. : $r_t = \ln \frac{P_t}{P_{t-1}}, t = 1, \dots, T = 1183.$ Здесь P_t – цены на электроэнергию на РСВ, r_t – логарифмическая доходность финансового актива.

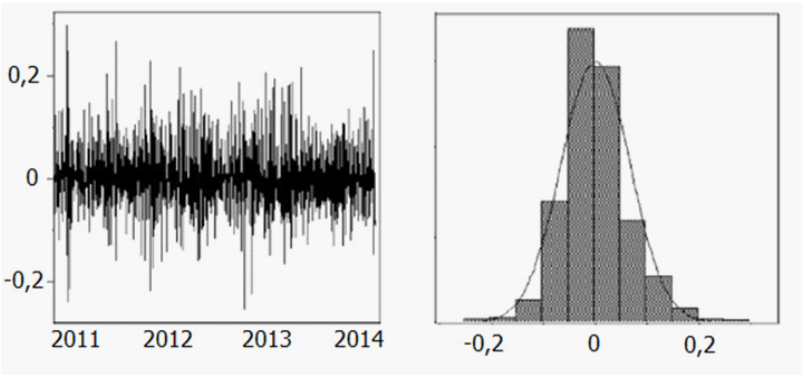

Рис. 1. Логарифмическая доходность цен на РСВ (слева), гистограмма эмпирического распределения и нормальная кривая (справа)

График эмпирической плотности распределения представляется гистограммой. Гистограмма и нормальная кривая с параметрами и изображены на рис. 1 (справа). Некоторые выборочные статистики исследуемого набора данных представлены в табл. 1.

Таблица 1

Статистические характеристики набора данных

Среднее $\overline{r}$	$5.772 \cdot 10^5$
Дисперсия $\overline{\sigma}$	0.0043
Асимметрия	0.4568
Эксцесс	1.5611
Минимум	-0.253756
Максимум	0.296632

6. Численные результаты

Для набора данных, представленных на рис. 1 (слева), получены следующие оценки параметров GARCH (1,1) модели (см. табл. 2).

Таблица 2

Оценки параметров GARCH (1,1) модели.

μ	0.0032015
ω	0.002273
α	0.464667
β	0.080312

7. Заключение

Очевидно, оптимизация является основой эконометрических приложений при изучении реальных временных рядов, при выборе модели и оценке параметров. Различные алгоритмы построения эконометрических моделей и их оптимизации имеют давнюю историю применения и весьма полно исследованы с формально-теоретической точки зрения. Однако анализ современного состояния доступного программного обеспечения в области эконометрических расчетов позволяет характеризовать его как неудо-

влетворительное. Проприетарность, отсутствие открытого кода, отсутствие информации об используемых допущениях и машинно-ориентированных критериях особенно затрудняют возможности отечественных исследователей в области практической эконометрики.

Например, даже одна из самых простых задач оптимизации – задача оценки параметров модели GARCH (1,1) является трудной с вычислительной точки зрения и требует тщательного анализа ее машинно-зависимых и проблемно-зависимых аспектов. Формальное применение стандартных численных алгоритмов аппроксимации может привести к неудаче при расчетах с использованием реальных наборов данных.

Каким же образом следует действовать при решении реальной прикладной задачи в ситуации, когда проблема оценки параметров модели является действительно сложной с вычислительной точки зрения? Одним из подходов может быть игнорирование проблемы и использование доступных стандартных методов, как будто проблемы не существует. Хотя данный подход вряд ли может быть рекомендован для серьезных исследований, в литературе имеется множество прикладных расчетов, адекватность которых не подтверждена каким-либо анализом и даже не обсуждается, либо вследствие незнания о существовании вычислительных проблем оптимизации, либо вследствие априорных представлений о том, что любая публичная программа, особенно проприетарная, дает заслуживающий доверия результат. Однако многими добросовестными исследователями отмечено, что некритичное использование программных продуктов типа «черного ящика» зачастую приводит к неверным результатам и, как следствие, к некорректным статистическим выводам и рекомендациям. Поэтому актуальной является задача разработки доступного компактного программного кода для решения частных задач эконометрики.

В данной работе приведен анализ машинно-ориентированного алгоритма решения задачи максимизации функции условного правдоподобия в рамках модели GARCH (1,1), обусловленных особенностями машинной арифметики. Представлена открытая программная реализация робастного алгоритма моделирования волатильности финансовых временных рядов в модели обобщенной авторегрессионной условной гетероскедастичности GARCH (1,1) при априорном предположении о нормальной форме распределения. Данный алгоритм является в известной степени эвристическим, мы не даем формального теоретического анализа таких его характеристик, как локальная скорость сходимости, свойства глобальной сходимости, работоспособность на специальных классах функций. Текст программы является самодостаточным и позволяет любому квалифицированному пользователю произвести реализацию на удобном для него языке программирования.

Наше обсуждение проиллюстрировано на численном примере. Построена эконометрическая модель дневной доходности энергетического индекса РСВ с использованием представленной программы.

Список источников

1. Дэннис Дж., Шнабель Р. *Численные методы безусловной минимизации и решения нелинейных уравнений*. Москва, Мир, 1988. 440 с.

2. Калиткин Н.Н. *Численные методы*. Москва, Наука, 1978. 512 с.

3. Bollerslev T. Generalized autoregressive conditional heteroskedasticity // J*ournal of Econometrics*, 1986, vol. 31, no 3, pp. 307-327.

4. Bollerslev T., Ghysels E. Periodic Autoregressive Conditional Heteroscedasticity // *Journal of Business and Economic Statistics*, 1996, no. 14, pp. 139-151.

5. Brooks C., Burke S.P., Persand G. Benchmarks and the Accuracy of GARCH Model Estimation // *International Journal of Forecasting*, 2001, no. 17, pp. 45-56.

6. Brooks C. *Introductory Econometrics for Finance*, 2nd ed. Cambridge University Press, 2008. 674 p.

7. Dennis J.E., Gay D.M, Welsch R.E. *An Adaptive Nonlinear Least-Squares Algorithm*. ACM Transactions on Mathematical Software, 1981, vol. 7, pp. 348-368.

8. *Econometrics Toolbox User's Guide.* The MathWorks, Inc, 2009. Доступно: http://cn.mathworks.com/help/pdf_doc/econ/econ.pdf. (дата обращения: 22.09.2014)

9. Engle R. *Autoregressive conditional heteroscedasticity with estimates of variance of United Kingdom inflation*. Econometrica, 1982, vol. 50, pp. 987-1008.

10. Engle R.F., Patton. A. *What good is a volatility model?* Quantitative Finance, 2001, vol. 1, pp. 237-245.

11. Fiorentini G., Calzolari G., Panattoni L. Analytic derivatives and the computation of GARCH estimates // *Journal of Applied Econometrics*, 1996, vol. 1, pp. 399-417.

12. McCullough B.D., Renfro C.G. Benchmarks and Software Standards: A Case Study of GARCH Procedures // *Journal of Economic and Social Measurement*, 1999, vol. 25, pp. 59-71.

13. More J.J. *In Numerical Analysis, Lecture Notes in Mathematics*. G.A.Watson, ed. (Berlin: Springer-Verlag), 1977, vol. 630, pp. 105-116.

14. More J.J., Garbow B.S. and Hillstrom K.E. *User Guide for MINPACK-1*. Argonne National Laboratory Report ANL-80-74, 1980.

15. Press W.H., Flannery B.P. *Numerical Recipes in C: The Art of Scientific Computing, Second Edition*. Cambridge University Press, 1992. 997 p.

16. Tsay R.S. *Analysis of Financial Time Series*, 2nd ed. Cambridge University Press, 2005. 576 p.

SOFTWARE IMPLEMENTATION OF ROBUST ALGORITHM FOR FINANCIAL INDICES VOLATILITY ESTIMATION WITHIN THE SCOPE OF GENERALIZED AUTOREGRESSIVE CONDITIONAL HETEROSKEDASTICITY GARCH (1,1) MODEL

Maglevanny Ilya Ivanovich[1], Dr. Sc. (Phys.-Math.), Prof.
Tsaplina Maria Georgievna[2], graduate student

[1]Volgograd State Pedagogical University, V.I. Lenin ave., 27, Volgograd, Russia, 400131; e-mail: sianko@list.ru, mlapenkova@gmail.ru

[2]Volgograd State University, University ave., 100, Volgograd, Russia, 400062; e-mail: mlapenkova@gmail.ru

Purpose: the article is devoted to problems of estimation and forecasting of volatility using a linear model GARCH (1,1) with the machine-dependent aspects. *Discussion*: various software packages having different numerical optimization methods may lead to different estimates of parameters working with the same source data. The solution to this problem is represented by the authors in the form of the development of a universal open source code based on its machine-dependent and domain-specific aspects. *Results*: authors propose a heuristic robust algorithm of numerical parameters estimation of the linear model GARCH (1,1) which can be easily implemented by any programming language, does not require third-party mathematical procedures, so as selection of the initial values of the parameters for the model volatility. As an example, we have studied the statistical characteristics of daily energy index RSV and built an econometric model of daily yield index for the model family of GARCH (1,1).

Keywords: volatility modeling, econometric estimations based on computer arithmetic, the model GARCH (1,1).

Reference

1. Dennis J., Shnabel R. *Chislennye metody bezuslovnoy minimizazii i resheniya nelineynyh uravneniy* [Numerical methods for unconstrained minimization and solving nonlinear equations]. Moscow, Mir, 1988. 440 p. (In Russ.)

2. Kalitkin N.N. *Chislennye metody* [Numerical methods]. Moscow, Nauka, 1978. 512 p. (In Russ.)

3. Bollerslev T. Generalized autoregressive conditional heteroskedasticity. *Journal of Econometrics*, 1986, vol. 31, no 3, pp. 307-327.

4. Bollerslev T., Ghysels E. Periodic Autoregressive Conditional Heteroscedasticity. *Journal of Business and Economic Statistics*, 1996, no. 14, pp. 139-151.

5. Brooks C., Burke S.P., Persand G. Benchmarks and the Accuracy of GARCH Model Estimation. *International Journal of Forecasting*, 2001, no. 17, pp. 45-56.

6. Brooks C. *Introductory Econometrics*

for Finance, 2nd ed. Cambridge University Press, 2008. 674 p.

7. Dennis J.E., Gay D.M, Welsch R.E. *An Adaptive Nonlinear Least-Squares Algorithm.* ACM Transactions on Mathematical Software, 1981, vol. 7, pp. 348-368.

8. *Econometrics Toolbox User's Guide.* The MathWorks, Inc, 2009. Available at: http://cn.mathworks.com/help/pdf_doc/econ/econ.pdf. (accessed: 22.09.2014)

9. Engle R. *Autoregressive conditional heteroscedasticity with estimates of variance of United Kingdom inflation*. Econometrica, 1982, vol. 50, pp. 987-1008.

10. Engle R.F., Patton. A. *What good is a volatility model?* Quantitative Finance, 2001, vol. 1, pp. 237-245.

11. Fiorentini G., Calzolari G., Panattoni L. Analytic derivatives and the computation of GARCH estimates. *Journal of Applied Econometrics*, 1996, vol. 1, pp. 399-417.

12. McCullough B.D., Renfro, C.G. Benchmarks and Software Standards: A Case Study of GARCH Procedures. *Journal of Economic and Social Measurement*, 1999, vol. 25, pp. 59-71.

13. More J.J. *In Numerical Analysis, Lecture Notes in Mathematics*. G.A. Watson, ed. (Berlin: Springer-Verlag), 1977, vol. 630, pp. 105-116.

14. More J.J., Garbow, B.S. and Hillstrom K.E. *User Guide for MINPACK-1.* Argonne National Laboratory Report ANL-80-74, 1980.

15. Press W.H., Flannery B.P. *Numerical Recipes in C: The Art of Scientific Computing, Second Edition*. Cambridge University Press, 1992. 997 p.

16. Tsay R.S. *Analysis of Financial Time Series, 2nd ed.* Cambridge University Press, 2005. 576 p.