Министерство образования и науки Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего профессионального образования
«Южно-Уральский государственный университет»
(Национальный исследовательский университет)
Факультет вычислительной математики и информатики
Кафедра экономико-математических методов и статистики

Реализация технического индикатора: индекс денежного потока

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА К КУРСОВОЙ РАБОТЕ
по дисциплине «Современные компьютерные технологии»
ЮУрГУ-010400.68.2017.049.001 КР

Руководитель,
_____ А.К. Богушев
« » _____2016 г.

Автор проекта
студент группы ВМИ-113
_____ В.А. Безбородов
« » _____2016 г.

Проект защищен
с оценкой
_____
« » _____2016 г.

Челябинск, 2016

Министерство образования и науки Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего профессионального образования
«Южно-Уральский государственный университет»
(Национальный исследовательский университет)
Факультет вычислительной математики и информатики
Кафедра экономико-математических методов и статистики

УТВЕРЖДАЮ
Заведующий кафедрой, д.ф.-м.н., профессор
_____ Панюков А.В.
« » _____ 2016 г.

З А Д А Н И Е

на курсовую работу студента

Безбородова Вячеслава Александровича

Группа ВМИ-113

1. Тема работы: Реализация технического индикатора: индекс денежного потока
2. Срок сдачи студентом законченной работы « » _____ 2016 г.
3. Исходные данные к работе
   3.1. Проект методического пособия по СОУ в формате MS Word;
   3.2. Издательская система компьютерной верстки LaTeX.
4. Перечень вопросов, подлежащих разработке
   4.1. Изучение языка и принципов работы в системе компьютерной верстки LaTeX;
   4.2. Проверка корректности исходных данных методического пособия;
   4.3. Трансляция методического пособия в формат LaTeX;

4.4. Разработка методических указаний к задачам по СОУ.

5. Перечень графического материала

6. Календарный план

Наименование этапов дипломной работы	Срок выполнения этапов работы	Отметка о выполнении
1. Сбор материалов и литературы по теме курсовой работы	10.09.2015 г.	
2. Изучение принципов работы с системой LaTeX	25.09.2015 г.	
3. Проверка корректности исходных данных методического пособия	03.10.2015 г.	
4. Трансляция методического пособия в формат LaTeX	18.10.2015 г.	
5. Разработка методических указаний по СОУ	27.10.2015 г.	
6. Подготовка пояснительной записки курсовой работы	09.11.2015 г.	
Написание главы 1	13.11.2015 г.	
Написание главы 2	17.11.2015 г.	
Написание главы 3	22.11.2015 г.	
7. Оформление пояснительной записки	02.12.2015 г.	
8. Получение отзыва руководителя	05.12.2015 г.	
9. Проверка работы руководителем, исправление замечаний	11.12.2015 г.	
10. Подготовка графического материала и доклада	16.12.2015 г.	
11. Защита курсовой работы	08.06.2016 г.	

7. Дата выдачи задания «___»_____2016 г.

Заведующий кафедрой_____/Панюков А.В./

Руководитель работы_____/А.К. Богушев/

Студент_____/В.А. Безбородов/

Министерство образования и науки Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего профессионального образования
«Южно-Уральский государственный университет»
(Национальный исследовательский университет)
Факультет вычислительной математики и информатики
Кафедра экономико-математических методов и статистики

АННОТАЦИЯ

Безбородов, В.А. Реализация технического индикатора: индекс денежного потока / В.А. Безбородов – Челябинск: ЮУрГУ, Факультет вычислительной математики и информатики, 2016 – 23 с., 1 прил., библиогр. список – 3 названий.

В курсовой работе дается краткое введение в системы одновременных уравнений, косвенный МНК и 2МНК. Производится сравнительный анализ наиболее популярных форматов, используемых в процессе электронного документооборота – MS Word и PDF. По результатам проведенного анализа для перевода проекта методических указаний выбирается наиболее надежный, безопасный и гибкий из них.

В приложениях приведены результаты верстки заданий и методических указаний к ним в системе LaTeX.

ОГЛАВЛЕНИЕ

Введение **6**

1 Технические индикаторы рынка **8**

2 Индекс денежного потока **9**

3 Реализация **11**

Заключение **17**

ПРИЛОЖЕНИЕ А. Исходный код приложения **19**

БИБЛИОГРАФИЧЕСКИЙ СПИСОК **23**

Введение

Самые успешные трейдеры и инвесторы во всем мире используют технические индикаторы рынка. Эта малая часть наиболее проницательных участников рынка добивается результатов, значительно превосходящих среднерыночные, и с каждым годом увеличивает размер принадлежащих им активов. Самым надежным способом присоединиться к этой категории финансистов является использование наиболее трезвого и реалистического подхода к инвестированию, основанного на объективном анализе поведения рынка, зафиксированного в исторических данных о торгах.

Индикаторы были разработаны рыночными профессионалами за несколько последних десятилетий путем тщательного ежедневного анализа поведения рыночных цен. Технические индикаторы предназначены для того, чтобы сделать сложный процесс принятия инвестиционных решений относительно простым, прозрачным и эффективным.

Целями работы являются: изучить язык и принципы работы в системе компьютерной верстки LaTeX; проверить корректность исходных данных методического пособия; перевести методическое пособие в формат LaTeX; разработать методические указания к задачам по СОУ.

В соответствии с поставленными целями в работе решаются следующие **задачи**: краткое ознакомление с системами одновременных уравнений, КМНК и 2МНК; выбор системы подготовки печати и способов верстки текста; разработка и набор методических указаний к задачам по СОУ.

Работа состоит из введения, 3 глав, заключения, 1 приложений и списка литературы. Объем работы составляет 23 страниц. Список литературы содержит 3 наименования.

В первой главе рассматриваются системы одновременных уравнений, структурная и приведенная формы модели, корреляция с ошибкой, проверка на идентифицируемость, а также КМНК и 2МНК.

Во второй главе производится сравнительный анализ наиболее распространенных форматов, используемых в электронном документообороте – MS Word и PDF.

В третьей главе приводятся технические особенности реализации перевода проекта методических указаний в формат LaTeX, а также обсуждаются наиболее сложные случаи набора.

В заключении перечислены основные результаты работы.

1 Технические индикаторы рынка

Доводы в пользу технических индикаторов рынка (ТИР).

1) Принимая к использованию или отклоняя конкретные ТИР, инвестор руководствуется логическими рассуждениями, доводами здравого смысла, а также данными относительно практической действенности индикатора, основанными на результатах его эффективности в прошлом.

2 Индекс денежного потока

Индекс денежного потока (MFI от англ. money flow index) — технический индикатор, призванный показать интенсивность, с которой деньги вкладываются в ценную бумагу и выводятся из неё, анализируя объёмы торгов и соотношения типичных цен периодов [1].

В качестве ключевого ценового показателя для индекса денежного потока используется типичная цена (англ. typical price), которая вычисляется по следующей формуле [3]:

$$\text{TypicalPrice}_t = \frac{\text{high}_t + \text{low}_t + \text{close}_t}{3},$$

где TypicalPrice_t – типичная цена, high_t – наибольшая цена, low_t – наименьшая цена, close_t – цена закрытия рассматриваемого периода t.

Денежный поток (англ. money flow) в каждом периоде вычисляется как произведение типичной цены на объём торгов в этом периоде:

$$\text{MoneyFlow}_t = \text{TypicalPrice}_t \cdot \text{volume}_t,$$

где MoneyFlow_t – денежный поток, TypicalPrice_t – типичная цена, volume_t – объем торгов.

На основе денежного потока вычисляются положительный и отрицательный денежные потоки:

$$\text{Positive}\textit{M}\text{oneyFlow}_t = \textit{M}\text{oneyFlow}_t, \text{ if } \text{TypicalPrice}_t > \text{TypicalPrice}_{t-1},$$

$$\text{Negative}\textit{M}\text{oneyFlow}_t = \textit{M}\text{oneyFlow}_t, \text{ if } \text{TypicalPrice}_t < \text{TypicalPrice}_{t-1},$$

где $\text{Positive}\textit{M}\text{oneyFlow}_t$ и $\text{Negative}\textit{M}\text{oneyFlow}_t$ – положительный и отрицательный денежные потоки.

Денежное отношение (англ. money ratio) в приложении к индексу MFI равно отношению сумм положительных и отрицательных денежных потоков

за выбранный промежуток времени:

$$\text{MoneyRatio}_{t,n} = \frac{\sum_{i=0}^{n-1} \text{PositiveMoneyFlow}_{t-i}}{\sum_{i=0}^{n-1} \text{NegativeMoneyFlow}_{t-i}},$$

где $\text{MoneyRatio}_{t,n}$ – денежное отношение в периоде t, построенное по n предыдущим периодам.

Индекс денежного потока приводит денежное отношение к интервалу $[0; 100]$:

$$\text{MFI}_{t,n} = 100 - \frac{100}{1 + \text{MoneyRatio}_{t,n}} =$$

$$= 100 \cdot \frac{\sum_{i=0}^{n-1} \text{PositiveMoneyFlow}_{t-i}}{\sum_{i=0}^{n-1} \text{PositiveMoneyFlow}_{t-i} + \sum_{i=0}^{n-1} \text{NegativeMoneyFlow}_{t-i}},$$

где $\text{MFI}_{t,n}$ – значение индекса денежного потока в периоде t, построенное по n предыдущим периодам.

Индекс денежного потока является осциллятором в интервале $[0; 100]$. Нижние его значения указывают на перепроданность рынка, верхние – на перекупленность. Все торговые стратегии, применимые к осцилляторам, могут быть использованы и в отношении MFI. Например [1]:

- Купить, когда MFI опускается ниже 20;
- Продать, когда MFI превышает 80.

3 Реализация

Рассмотрим программную реализацию технического индикатора. В качестве языка программирования был выбран Python.

Python – высокоуровневый язык программирования общего назначения, ориентированный на повышение производительности разработчика и читаемости кода. Синтаксис ядра Python минималистичен. В то же время стандартная библиотека включает большой объём полезных функций.

Python поддерживает несколько парадигм программирования, в том числе структурное, объектно-ориентированное, функциональное, императивное и аспектно-ориентированное. Основные архитектурные черты – динамическая типизация, автоматическое управление памятью, полная интроспекция, механизм обработки исключений, поддержка многопоточных вычислений и удобные высокоуровневые структуры данных. Код в Python организовывается в функции и классы, которые могут объединяться в модули (они в свою очередь могут быть объединены в пакеты).

Эталонной реализацией Python является интерпретатор CPython, поддерживающий большинство активно используемых платформ. Он распространяется под свободной лицензией Python Software Foundation License, позволяющей использовать его без ограничений в любых приложениях, включая проприетарные. Есть реализации интерпретаторов для JVM (с возможностью компиляции), MSIL (с возможностью компиляции), LLVM и других. Проект PyPy предлагает реализацию Python на самом Python, что уменьшает затраты на изменения языка и постановку экспериментов над новыми возможностями.

Python – активно развивающийся язык программирования, новые версии (с добавлением/изменением языковых свойств) выходят примерно раз в два с половиной года. Вследствие этого и некоторых других причин на Python отсутствуют стандарт ANSI, ISO или другие официальные стандарты, их роль выполняет CPython.

Как и в большинстве случаев, управляющий поток начинает выполнение со стандартной конструкции с выводом строки приветствия.

```
90 if __name__ == '__main__':
```

```
91      print('Data processor 0.2')
```

Далее, подключаются необходимые для работы модули:

- **sys** – модуль предоставляет доступ к некоторым переменным и функциям, используемым интерпретатором;
- **argparse** – используется для написания дружественных интерфейсов командной строки;
- **logging** – модуль определяет функции и классы, реализующие расширяемую систему логирования для приложений и библиотек.

```
93      import sys
94      import argparse
95      import logging
```

Производится настройка аргументов командной строки для конфигурирования параметров приложения.

```
97      argparser = argparse.ArgumentParser()
98      argparser.add_argument("--symbol", help="stock symbol")
99      argparser.add_argument("--file", help="file to process by program")
100     argparser.add_argument("--out", help="output file")
101     argparser.add_argument("--logfile", help="log file")
102     argparser.add_argument("--log", help="log level")
103     argparser.add_argument("--year", help="year")
104     argparser.add_argument("--mfi", help="money flow count")
105     argparser.add_argument("--level", help="mfi level")
106     args = argparser.parse_args()
```

Также настраивается система логирования для вывода сообщений в файл.

```
111         if args.log:
112             numeric_level = getattr(logging, args.log.upper(), None)
113 
114             if not isinstance(numeric_level, int):
115                 raise ValueError('Invalid log level: %s' % args.log)
116 
117             logging.basicConfig(filename=args.logfile, level=numeric_level)
```

При необходимости, вывод перенаправляется либо в файл, либо в стандартный поток вывода.

```
119         if args.out:
120             ostream = open(args.out, 'w')
121         else:
122             ostream = sys.stdout
```

Происходит конфигурирование периода, за который необходимо получить информацию.

```
124         if args.year:
125             args.year = int(args.year)
126             from datetime import MAXYEAR
127             if args.year > MAXYEAR:
128                 args.year = MAXYEAR
129         else:
130             args.year = 1900
```

На этом этапе логика работы приложения делится. Либо данные читаются из файла, если в параметрах командной строки был задан путь до файла. Либо данные скачиваются из сети Интернет, если был задан адрес.

```
134         if args.file:
135             from os import path
136             uri = 'file://' + path.abspath(args.file)
137             dateFormat = '%Y-%m-%d'
138         elif args.symbol:
139             from datetime import date
140             from urllib2 import quote
141             start = date(args.year, 1, 1)
142             end = date(args.year, 12, 31)
143             uri = 'http://www.google.com/finance/historical?' +
144                 'q={0}&startdate={1}&enddate={2}&output=csv'
145             uri = uri.format(args.symbol.upper(),
146                 quote(start.strftime('%b %d, %Y')),
147                 quote(end.strftime('%b %d, %Y')))
148             dateFormat = '%d-%b-%y'
```

Происходит загрузка данных из источника, вывод результатов обработки в приемник и выдача стратегии MFI по требованию.

```
150         if uri:
151             data = load_data(uri, dateFormat)
152 
153             from json import dump
154             dump(process_data(data, args.year), ostream)
155 
156             if args.mfi:
157                 dump(get_mfi_strategy(data), ostream)
```

Все ошибки приложение обратывает с помощью механизма исключений с выводом причины в файл.

```
159     except IOError as e:
```

```
160             logging.error("Can't open source: {0}".format(e))
161         except:
162             logging.error("Bad thing happened")
```

Рассмотрим подробнее механизм загрузки данных. Функция принимает на вход адрес с данными и выводит информационное сообщение.

```
3  def load_data(uri, dateFormat):
4      logging.info('loading data; uri: {0}'.format(uri))
```

Производится подключение небходимых для работы функций и классов.

- **urllib2** – модуль определяет функции и классы, которые предназначены для окрытия веб страниц;
- **csv** – модуль реализует классы для чтения и записи табличных данных в формате CSV;
- **time** – модуль для работы с датой и временем.

```
6      from urllib2 import urlopen
7      from csv import DictReader

18     from time import strptime
```

Происходит окрытие указанного адреса и чтение данных.

```
9      reader = DictReader(urlopen(uri).readlines())
```

Данные читаются построчно, формируя заданную структуру ответа, которая возвращается как результат работы функции.

```
20     for row in reader:
21         data.append({
22             'date': strptime(row['Date'], dateFormat),
23             'open': float(row['Open']),
24             'close': float(row['Close']),
25             'high': float(row['High']),
26             'low': float(row['Low']),
27             'volume': float(row['Volume'])
28         })
29
30     return data
```

Функция обработки данных принимает на вход сформированные данные и период обработки.

```
73 def process_data(data, year):
74     logging.info('processing data; year: {0}'.format(year))
```

Происходит подсчет средних значений показателей за рассмотриваемый период, и результат обработки возвращается.

```
75      res = {'open': 0, 'close': 0, 'high': 0, 'low': 0}
76
77      for row in data:
78          if row['date'].tm_year == year:
79              res['open'] += row['open']
80              res['close'] += row['close']
81              res['high'] += row['high']
82              res['low'] += row['low']
83
84      if len(data) != 0:
85          for (key, value) in res.items():
86              res[key] /= len(data)
87
88      return res
```

Функция получения MFI стратегии использует показание технического индикатора для формирования своего результата.

```
66  def get_mfi_strategy(data):
67      mfi = get_mfi(data, len(data) - 1, len(data))
68      if mfi < 20:
69          return {'mfi': mfi, 'strategy': 'buy'}
70      elif mfi > 80:
71          return {'mfi': mfi, 'strategy': 'sell'}
```

Функция расчета индикатора нормирует значение денежного отношения.

```
63  def get_mfi(data, t, n):
64      return 100 - 100 / (1 + get_money_ratio(data, t, n))
```

Денежное отношение – это отношение положительного денежного потока за период к отрицательному денежному потоку за тот же период.

```
60  def get_money_ratio(data, t, n):
61      return get_positive_money_flow_total(data, t, n) /
62          get_negative_money_flow_total(data, t, n)
```

Функции получения полного положительного/отрицательного денежного потока используют методы получения денежного потока за период.

```
48  def get_positive_money_flow_total(data, t, n):
49      total = 0
50      for i in range(n):
51          total += get_positive_money_flow(data, t, i)
52      return total
```

```
53
54 def get_negative_money_flow_total(data, t, n):
55     total = 0
56     for i in range(n):
57         total += get_negative_money_flow(data, t, i)
58     return total
```

Методы получения знакового денежного потока оперируют понятием типичной цены.

```
38 def get_positive_money_flow(data, t, i):
39     if get_typical_price(data, t) > get_typical_price(data, t - i):
40         return get_money_flow(data, t)
41     return 0
42
43 def get_negative_money_flow(data, t, i):
44     if get_typical_price(data, t) < get_typical_price(data, t - i):
45         return get_money_flow(data, t)
46     return 0
```

Денежный поток – это произведение типичной цены на объем продаж.

```
35 def get_money_flow(data, t):
36     return get_typical_price(data, t) * data[t]['volume']
```

Типичная цена, в свою очередь, рассчитывается как среднее среди максимальной, минимальной и ценой закрытия.

```
32 def get_typical_price(data, t):
33     return (data[t]['high'] + data[t]['low'] + data[t]['close']) / 3.0
```

Таким образом, мы рассмотрели программные особенности реализации технического индикатора "денежный поток".

Заключение

В работе представлено методическое пособие, переведенное в формат LaTeX, а также методические указания к задачам по системам одновременных уравнений.

В работе решены следующие **задачи**:

- изучен язык и принципы работы в системе компьютерной верстки LaTeX;
- проверена корректность исходных данных методического пособия;
- методическое пособие переведено в формат LaTeX;
- разработаны методические указания к задачам по СОУ.

По результатам работы можно сделать следующие выводы:

- в электронном документообороте предпочтительнее использовать формат PDF ввиду его надежности, гибкости и безопасности;
- разработанные методические указания помогают подойти к изучению СОУ комплексно, предоставляя возможность студентам самостоятельно лучше усваивать материал.

Методическое пособие, переведенное в формат PDF, может быть в дальнейшем использовано в Электронном ЮУрГУ – системе, предназначенной для организации учебного процесса с применением информационных технологий в Южно-Уральском государственном университете.

ПРИЛОЖЕНИЯ

ПРИЛОЖЕНИЕ А

ИСХОДНЫЙ КОД ПРИЛОЖЕНИЯ

```
#!/usr/bin/python

def load_data(uri, dateFormat):
    logging.info('loading data; uri: {0}'.format(uri))

    from urllib2 import urlopen
    from csv import DictReader

    reader = DictReader(urlopen(uri).readlines())

    encodedFieldNames = []
    for fieldname in reader.fieldnames:
        encodedFieldNames.append(fieldname.decode("utf-8-sig").encode("utf-8"))
    reader.fieldnames = encodedFieldNames

    data = []

    from time import strptime

    for row in reader:
        data.append({
            'date': strptime(row['Date'], dateFormat),
            'open': float(row['Open']),
            'close': float(row['Close']),
            'high': float(row['High']),
            'low': float(row['Low']),
            'volume': float(row['Volume'])
        })

    return data

def get_typical_price(data, t):
    return (data[t]['high'] + data[t]['low'] + data[t]['close']) / 3.0

def get_money_flow(data, t):
    return get_typical_price(data, t) * data[t]['volume']

def get_positive_money_flow(data, t, i):
    if get_typical_price(data, t) > get_typical_price(data, t - i):
        return get_money_flow(data, t)
    return 0

```

```
43 def get_negative_money_flow(data, t, i):
44     if get_typical_price(data, t) < get_typical_price(data, t - i):
45         return get_money_flow(data, t)
46     return 0
47 
48 def get_positive_money_flow_total(data, t, n):
49     total = 0
50     for i in range(n):
51         total += get_positive_money_flow(data, t, i)
52     return total
53 
54 def get_negative_money_flow_total(data, t, n):
55     total = 0
56     for i in range(n):
57         total += get_negative_money_flow(data, t, i)
58     return total
59 
60 def get_money_ratio(data, t, n):
61     return get_positive_money_flow_total(data, t, n) /
62         get_negative_money_flow_total(data, t, n)
63 def get_mfi(data, t, n):
64     return 100 - 100 / (1 + get_money_ratio(data, t, n))
65 
66 def get_mfi_strategy(data):
67     mfi = get_mfi(data, len(data) - 1, len(data))
68     if mfi < 20:
69         return {'mfi': mfi, 'strategy': 'buy'}
70     elif mfi > 80:
71         return {'mfi': mfi, 'strategy': 'sell'}
72 
73 def process_data(data, year):
74     logging.info('processing data; year: {0}'.format(year))
75     res = {'open': 0, 'close': 0, 'high': 0, 'low': 0}
76 
77     for row in data:
78         if row['date'].tm_year == year:
79             res['open'] += row['open']
80             res['close'] += row['close']
81             res['high'] += row['high']
82             res['low'] += row['low']
83 
84     if len(data) != 0:
85         for (key, value) in res.items():
86             res[key] /= len(data)
87 
88     return res
```

```
89
90  if __name__ == '__main__':
91      print('Data processor 0.2')
92
93      import sys
94      import argparse
95      import logging
96
97      argparser = argparse.ArgumentParser()
98      argparser.add_argument("--symbol", help="stock symbol")
99      argparser.add_argument("--file", help="file to process by program")
100     argparser.add_argument("--out", help="output file")
101     argparser.add_argument("--logfile", help="log file")
102     argparser.add_argument("--log", help="log level")
103     argparser.add_argument("--year", help="year")
104     argparser.add_argument("--mfi", help="money flow count")
105     argparser.add_argument("--level", help="mfi level")
106     args = argparser.parse_args()
107
108     ostream = sys.stdout
109
110     try:
111         if args.log:
112             numeric_level = getattr(logging, args.log.upper(), None)
113
114             if not isinstance(numeric_level, int):
115                 raise ValueError('Invalid log level: %s' % args.log)
116
117             logging.basicConfig(filename=args.logfile, level=numeric_level)
118
119         if args.out:
120             ostream = open(args.out, 'w')
121         else:
122             ostream = sys.stdout
123
124         if args.year:
125             args.year = int(args.year)
126             from datetime import MAXYEAR
127             if args.year > MAXYEAR:
128                 args.year = MAXYEAR
129         else:
130             args.year = 1900
131
132         uri = None
133
134         if args.file:
```

```
135             from os import path
136             uri = 'file://' + path.abspath(args.file)
137             dateFormat = '%Y-%m-%d'
138         elif args.symbol:
139             from datetime import date
140             from urllib2 import quote
141             start = date(args.year, 1, 1)
142             end = date(args.year, 12, 31)
143             uri = 'http://www.google.com/finance/historical?' +
144                 'q={0}&startdate={1}&enddate={2}&output=csv'
145             uri = uri.format(args.symbol.upper(),
146                 quote(start.strftime('%b %d, %Y')),
147                 quote(end.strftime('%b %d, %Y')))
148             dateFormat = '%d-%b-%y'
149 
150         if uri:
151             data = load_data(uri, dateFormat)
152 
153             from json import dump
154             dump(process_data(data, args.year), ostream)
155 
156             if args.mfi:
157                 dump(get_mfi_strategy(data), ostream)
158 
159     except IOError as e:
160         logging.error("Can't open source: {0}".format(e))
161     except:
162         logging.error("Bad thing happened")
163 
164     ostream.close()
```

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. Акелис, С. Б. Денежных потоков индекс (Money Flow Index) // Технический анализ от А до Я. Полный набор инструментов торговли... от "Абсолютного индекса ширины"до "Японских свечей"/ Пер. с англ. М. Волкова, А. Лебедева. / Стивен Б. Акелис. — М.: Диаграмма, 1999. — 376 с.
2. Львовский, С. М. Набор и верстка в системе LATEX / С. М. Львовский. — 2003.
3. Роберт, К. Энциклопедия технических индикаторов рынка. / Колби Роберт. — М.: "Альпина Бизнес Букс 2006. — 837 с.