# Package 'TickExec'

May 20, 2015

**Type** Package

**Title** Execution Functions for Tick Data Back Test

**Version** 1.1

**Date** 2015-04-20

**Author** HKUST

**Maintainer** SONG Yang <ysongad@connect.ust.hk>

**Description** Functions to execute orders in backtesting using tick data. A testing platform was established by the four major execution functions, namely 'LimitBuy', 'LimitSell', 'MarketBuy' and 'MarketSell', which enclosed all tedious aspects (such as queueing for order executions and calculate actual executed volumes) for order execution using tick data. Such that one can focus on the logic of strategies, rather than its execution.

**License** GPL-3

**NeedsCompilation** no

**Repository** CRAN

**Date/Publication** 2015-05-20 07:54:30

## R topics documented:

---

`TickExec-package` *Execution Functions for Tick Data Back Test*

---

## Description

Functions to execute orders in backtesting using tick data. A testing platform was established by the four major execution functions, namely 'LimitBuy', 'LimitSell', 'MarketBuy' and 'MarketSell', which enclosed all tedious aspects (such as queueing for order executions and calculate actual executed volumes) for order execution using tick data. Such that one can focus on the logic of strategies, rather than its execution.

## Details

| | |
|---|---|
| Package: | TickExec |
| Type: | Package |
| Version: | 1.1 |
| Date: | 2015-04-20 |
| License: | PGL-3 |

## Author(s)

HKUST. Maintainer: SONG Yang <ysongad@connect.ust.hk>

---

`.LoadTickDataHK` *Load Tick Data for Hong Kong Stock Market*

---

## Description

NOT to be called by user!!! Only for 'LoadTickData' to call.

### Usage

```
.LoadTickDataHK(dir, ticker, date, CALL = 'BUY')
```

### Arguments

| | |
|---|---|
| dir | The directory containing the Tick data. |
| date | date of the tick data. |
| ticker | ticker of the tick data. |
| CALL | 'BUY', 'SELL' or 'TRADE', depends on the chunk of the data wanted. |

### Value

A dataframe, or NA if no entries satisfy the given conditions.

---

## .LoadTickDataSHSZ *Load Tick Data for SHSZ Stock Market*

---

### Description

NOT to be called by user!!! Only for 'LoadTickData' to call.

### Usage

```
.LoadTickDataSHSZ(dir, ticker, date, CALL = 'BUY')
```

### Arguments

| | |
|---|---|
| dir | The directory containing the Tick data. |
| date | date of the tick data. |
| ticker | ticker of the tick data. |
| CALL | 'BUY', 'SELL' or 'TRADE', depends on the chunk of the data wanted. |

### Value

A dataframe, or NA if no entries satisfy the given conditions.

## DataSlice *Truncate Given Dataframe According to Given Time Window*

### Description

The input dataframe should have a column named 'Time'. If both 'time1', 'time2' and 'last' were given, then the actual window size should the smaller of (time2 - time1) and 'last'.

### Usage

```
DataSlice(df, time1 = 000001, time2 = 235959, last = 24 * 3600)
```

### Arguments

| | |
|---|---|
| df | dataframe to be truncated. |
| time1 | lower bound of the time window. |
| time2 | upper bound of the time window. |
| last | width od the time window. |

### Value

A dataframe, or NA if no entries satisfy the given time window.

## DrawDown *Calculate Maximum Draw Down of Series*

### Description

Input should be a non-negative series.

### Usage

```
DrawDown(x)
```

### Arguments

| | |
|---|---|
| x | Cumulative wealth (or return) process, non-negative, NA's will be ignored. |

### Value

A percentage between 0 and 1.

## Examples

```
## construct a series ##
x <- rnorm(100) + 10

## calculate drawdown ##
DrawDown(x)
```

---

`GetLastPrice` *Retrieve Last Trade Price of Given Instrument*

---

## Description

Seek the last trade price happened before given timestamp.

## Usage

```
GetLastPrice(dir = dir, date, time, ticker, market = 'SHSZ')
```

## Arguments

| | |
|---|---|
| `dir` | The directory containing the Tick data. |
| `date` | date of the tick data. |
| `ticker` | ticker of the tick data. |
| `time` | timestamp of the wanted price. |
| `market` | specifying the sub-function to call depending on the market. |

## Value

A number, or NA if no entry satisfy the given conditions.

---

`GetQueueLength` *Retrieve Length of Quening Orders at Given Price*

---

## Description

Buy order should queue at bid prices, while sell order should queue at ask prices.

## Usage

```
GetQueueLength(dir = dir, date, orderTime, ticker, limitPrice, CALL,
               position = 1, market = 'SHSZ')
```

### Arguments

| | |
|---|---|
| dir | The directory containing the Tick data. |
| date | date of the tick data. |
| orderTime | timestamp of the order queue. |
| ticker | ticker of the tick data. |
| limitPrice | the price level of the queue. |
| CALL | sell order or buy order. |
| position | position is between 0 and 1, indicating the relative position of the current order in the queue. |
| market | specifying the sub-function to call depending on the market. |

### Value

An integer indicating the queue length, or 0 if no entry satisfy the given conditions.

---

## InitLogEntry *Initialize Log for Each Trade*

---

### Description

This function should only be called buy 'MarketBuy', 'MarketSell', 'LimitBuy' or 'LimitSell', not by users.

### Usage

```
InitLogEntry(dateIn, ticker, capital, timeIn = NA, execVol = 0,
             execQuant = 0, avgPrice = NA, depthIn = NA)
```

### Arguments

| | |
|---|---|
| dateIn | date of the trade. |
| ticker | ticker for traget instrument, characters or numbers are both acceptable. |
| capital | amount of money wanted to fully invested on the instrument. |
| timeIn | The time when the trade was first executed. |
| execVol | executed volume, in shares. |
| execQuant | executed quantities, in dollars. |
| avgPrice | average price achieved the the trade. |
| depthIn | the level of ask prices that trade had reached. 0 for limit orders. |

### Value

A dataframe.

LimitBuy *Execute Limit Buy Order*

## Description

(Try) to buy a given instrument at given date and time slot, by limit order. If limit price was not given, then use the last trading price as limit price. If both 'orderTo' and 'orderLast' was given, then the smaller one will be adopted.

## Usage

```
LimitBuy(dir = dir, date, ticker, capital, limitPrice = NA, orderFrom,
         orderTo = 150000, orderLast = 7 * 3600, costIn = 0.001,
         market = 'SHSZ')
```

## Arguments

| | |
|---|---|
| dir | The directory containing the Tick data. |
| date | the date for placing the order. |
| ticker | ticker for traget instrument, characters or numbers are both acceptable. |
| capital | amount of money wanted to fully invested on the instrument. |
| limitPrice | the limit price to sell. |
| orderFrom | time of the order being placed. |
| orderTo | time of the order being withdrawed. |
| orderLast | duration of the order, in seconds. |
| costIn | transaction cost for buying. |
| market | specifying the sub-function to call depending on the market. |

## Value

A dataframe, with corresponding summary statistics.

## Examples

```
## locate tick data directory ##
dir <- system.file("extdata", '', package = "TickExec")

## Execute order, given duration ##
dfLog1 = LimitBuy(dir = dir, date = 20141013, ticker = 000001, capital = 1e6,
                  limitPrice = NA, orderFrom = 94545, orderLast = 600,
                  costIn = 0.001, market = 'SHSZ')

## Execute order, given ending time ##
dfLog2 = LimitBuy(dir = dir, date = 20141013, ticker = 000001, capital = 1e6,
                  limitPrice = NA, orderFrom = 94545, orderTo = 100001,
```

```
                costIn = 0.001, market = 'SHSZ')

## see result ##
dfLog1
dfLog2
```

---

`LimitSell` *Execute Limit Sell Order*

---

## Description

(Try) to sell a given instrument at given date and time slot, by limit order. If limit price was not given, then use the last trading price as limit price. If both 'orderTo' and 'orderLast' was given, then the smaller one will be adopted.

## Usage

```
LimitSell(dir = dir, date, dfLog, limitPrice = NA, orderFrom, orderTo = 150000,
          orderLast = 7 * 3600, costOut = 0.001, market = 'SHSZ')
```

## Arguments

| | |
|---|---|
| `dir` | The directory containing the Tick data. |
| `date` | the date for placing the order. |
| `dfLog` | The dataframe generated by buy-orders. |
| `limitPrice` | the limit price to sell. |
| `orderFrom` | time of the order being placed. |
| `orderTo` | time of the order being withdrawed. |
| `orderLast` | duration of the order, in seconds. |
| `costOut` | transaction cost for selling. |
| `market` | specifying the sub-function to call depending on the market. |

## Value

The same dataframe dfLog, with corresponding entries updated.

## Examples

```
## locate tick data directory ##
dir <- system.file("extdata", '', package = "TickExec")

## establish a posiyion to sell ##
dfLog = LimitBuy(dir = dir, date = 20141013, ticker = 000001, capital = 1e6,
                 limitPrice = NA, orderFrom = 94545, orderLast = 600,
                 costIn = 0.001, , market = 'SHSZ')
```

```
## sell ##
dfLogSold = LimitSell(dir = dir, date = 20141013, dfLog = dfLog, limitPrice = 10.1,
                      orderFrom = 142020, orderTo = 150000, costOut = 0.001,
                      market = 'SHSZ')

## see result ##
dfLogSold
```

---

`LoadTickData` *Locate and Load Tick Data of Given Instrument at Given Date*

---

## Description

Since the format of the data may vary, this funcation is expecting to vary among different datasets and data structures.

## Usage

```
LoadTickData(dir, ticker, date, CALL = 'BUY', market = 'SHSZ')
```

## Arguments

| | |
|---|---|
| `dir` | The directory containing the Tick data. |
| `date` | date of the tick data. |
| `ticker` | ticker of the tick data. |
| `CALL` | 'BUY', 'SELL' or 'TRADE', depends on the chunk of the data wanted. |
| `market` | specifying the sub-function to call depending on the market. |

## Value

A dataframe, or NA if no entries satisfy the given conditions.

---

`MarketBuy` *Execute Market Buy Order*

---

## Description

(Try) to buy a given instrument at given date and time, by market order.The function can at most penetrate 5 levels of ask prices. For large capital, the average executed price WILL be elevated.

## Usage

```
MarketBuy(dir = dir, date, ticker, capital, orderTime, costIn = 0.001,
          market = 'SHSZ')
```

### Arguments

| | |
|---|---|
| dir | The directory containing the Tick data. |
| date | the date for placing the order. |
| ticker | ticker for traget instrument, characters or numbers are both acceptable. |
| capital | amount of money wanted to fully invested on the instrument. |
| orderTime | Time of the day to place the market order. |
| costIn | transaction cost for buying. |
| market | specifying the sub-function to call depending on the market. |

### Value

A dataframe, with corresponding summary statistics.

### Examples

```
## locate tick data directory ##
dir <- system.file("extdata", '', package = "TickExec")

## Execute order ##
dfLog = MarketBuy(dir = dir, date = 20141010, ticker = 000001, capital = 1e5,
                  orderTime = 94545, costIn = 0.001, market = 'SHSZ')

## see result ##
dfLog
```

---

## MarketSell *Execute Market Sell Order*

---

### Description

(Try) to sell a given instrument at given date and time, by market order. The function can at most penetrate 5 levels of bid prices. For large capital, the average executed price WILL be depressed. There is a possibility that the order will not be fully executed and some holding volumes will remain.

### Usage

```
MarketSell(dir = dir, date, orderTime, dfLog, costOut = 0.001, market = 'SHSZ')
```

### Arguments

| | |
|---|---|
| dir | The directory containing the Tick data. |
| date | the date for placing the order. |
| orderTime | Time of the day to place the market order. |
| dfLog | The dataframe generated by buy-orders. |
| costOut | transaction cost for selling. |
| market | specifying the sub-function to call depending on the market. |

### Value

The same dataframe dfLog, with corresponding entries updated.

### Examples

```
## locate tick data directory ##
dir <- system.file("extdata", '', package = "TickExec")

## establish a posiyion to sell ##
dfLog = LimitBuy(dir = dir, date = 20141013, ticker = 000001, capital = 1e6,
                 limitPrice = NA, orderFrom = 94545, orderLast = 600,
                 costIn = 0.001, market = 'SHSZ')

## sell ##
dfLogSold = MarketSell(dir = dir, date = 20141014, orderTime = 140001,
                       dfLog = dfLog, costOut = 0.001, market = 'SHSZ')

## see result ##
dfLogSold
```

---

## PerformanceReport *Summarize Back Test Performance*

---

### Description

Give 13 basic indicators base on the simple arithmetic investments.

### Usage

```
PerformanceReport(df, cumPnL, initCap = NA)
```

### Arguments

| | |
|---|---|
| df | The dataframe containing the portfolio. |
| cumPnL | the daily pnl series, including those days with no trades. |
| initCap | initial capital, if given 'NA', then use the total capital recorded on the first trading day in the trade log 'df'. |

### Value

A dataframe with 13 basic indicators.

| | |
|---|---|
| DAYS | number of total trading days. |
| FIRSTTRD | the day when first trade happened, normally the first trading day. |
| LASTTRD | the day when last trade happened, normally the last trading day. |
| NONTRDPERC | percentage for non-trading days. |
| DAILYTRD | average number of trades daily. |

| | |
|---|---|
| TOTALPNL | total pnl. |
| RETPERTRD | average return per trade. |
| TRDHITRAT | trade-wise hit rate. |
| DLYHITRAT | daily hit rate. |
| ANNRET | annual return. |
| SHARPE | annual sharpe ratio. |
| DRAWDOWN | maximum draw down. |
| INRETURN | intrinsic return. |

## Examples

```
## locate tick data directory ##
dir <- system.file("extdata", '', package = "TickExec")
ticker = 000001

df <- c()
pnl <- c()

for (d in 20141012:20141017) {

dfLog = LimitBuy(dir = dir, date = d, ticker = ticker, capital = 1e6,
                 limitPrice = NA, orderFrom = 94545, orderLast = 600,
                 costIn = 0.001, market = 'SHSZ')

dfLogSold = MarketSell(dir = dir, date = d, orderTime = 140001,
                       dfLog = dfLog, costOut = 0.001, market = 'SHSZ')


df <- rbind(df, dfLogSold)
pnl <- c(pnl, TotalPnL(dir = dir, df = df, date = d))
}

PerformanceReport(df = df, cumPnL = pnl, initCap = 1e6)
```

---

# PortfolioWorth *Evaluate Market Worth of Given Protfolio*

---

## Description

The evaluation is done at time of a given date.

## Usage

```
PortfolioWorth(dir = dir, df, date, time = 145900, market = 'SHSZ')
```

### Arguments

| | |
|---|---|
| dir | The directory containing the Tick data. |
| date | the date of evaluation. |
| time | the time of evaluation. |
| df | The dataframe containing the portfolio. |
| market | specifying the sub-function to call depending on the market. |

### Value

A number indicating the markrt worth.

### Examples

```
## locate tick data directory ##
dir <- system.file("extdata", '', package = "TickExec")

## establish a posiyion to sell ##
dfLog = LimitBuy(dir = dir, date = 20141013, ticker = 000001, capital = 1e6,
                 limitPrice = NA, orderFrom = 94545, orderLast = 600,
                 costIn = 0.001, market = 'SHSZ')

## sell ##
dfLogSold = MarketSell(dir = dir, date = 20141014, orderTime = 140001,
                       dfLog = dfLog, costOut = 0.001, market = 'SHSZ')

## market worth ##
PortfolioWorth(dir = dir, df = dfLogSold, date = 20141014, time = 145900,
               market = 'SHSZ')
```

---

## PriceToNA *Set 0 In Price To NA*

---

### Description

This is not to be called by user, for the lack of data-type checking mechanism.

### Usage

```
PriceToNA(df)
```

### Arguments

| | |
|---|---|
| df | object whose 0s to be replaced NAs. |

### Value

A dataframe, or matrix depending on the class of the argument.

## SecondsToTime *Calculate Timestamp*

### Description

Number of seconds since midnight must be given, only work with intraday time without date.

### Usage

```
SecondsToTime(seconds)
```

### Arguments

| | |
|---|---|
| seconds | number of seconds since midnight. |

### Value

A timestamp, in the form of a 5 (or 6) digits integer.

## SimpleReturn *Calculate Simple Price to Price Return*

### Description

Used to calculate close-to-close, open-to-close, etc, returns from given price matrices. Duration can be 0 or any positive integers.

### Usage

```
SimpleReturn(priceFrom, priceTo, diff = 0)
```

### Arguments

| | |
|---|---|
| priceFrom | A matrix with the beginning prices. |
| priceTo | A matrix with the ending prices. |
| diff | Number of rows for the two matrices' dislocation. |

### Value

A matrix, of corresponding returns.

## TimeAdd *Calculate Endpoint Timestamp*

### Description

Beginning timestamp and duration must be given, only work with intraday time without date.

### Usage

```
TimeAdd(time1, increase)
```

### Arguments

| | |
|---|---|
| time1 | first timestamp, 5 or 6 digits. |
| increase | duration in senconds. |

### Value

A timestamp, in the form of a 5 (or 6) digits integer.

## TimeDiff *Find Difference Between Timestamps*

### Description

Only work with intraday time without date. origin time NOT necessarily less than ending time. Differnece must be given in seconds.

### Usage

```
TimeDiff(time1, time2)
```

### Arguments

| | |
|---|---|
| time1 | first timestamp, 5 or 6 digits. |
| time2 | second timestamp, 5 or 6 digits. |

### Value

An integer, possibly negative.

TotalPnL *Calculate Total PnL for Given Protfolio*

## Description

The evaluation is done at time of a given date. Length of 'TotalPnL' should be the same as number of trading days, not as 'dfLog'.

## Usage

```
TotalPnL(dir = dir, df, date, time = 160000, market = 'SHSZ')
```

## Arguments

| | |
|---|---|
| dir | The directory containing the Tick data. |
| df | The dataframe containing the portfolio. |
| date | the date of evaluation. |
| time | the time of evaluation. |
| market | specifying the sub-function to call depending on the market. |

## Value

A number indicating the total pnl.

## Examples

```
## locate tick data directory ##
dir <- system.file("extdata", '', package = "TickExec")

## establish a posiyion to sell ##
dfLog = LimitBuy(dir = dir, date = 20141013, ticker = 000001, capital = 1e6,
                 limitPrice = NA, orderFrom = 94545, orderLast = 600,
                 costIn = 0.001, market = 'SHSZ')

## sell ##
dfLogSold = MarketSell(dir = dir, date = 20141014, orderTime = 140001,
                       dfLog = dfLog, costOut = 0.001, market = 'SHSZ')

## market worth ##
TotalPnL(dir = dir, df = dfLogSold, date = 20141014, time = 145900,
               market = 'SHSZ')
```

`VolumeToZero` *Set NA in Volume to 0*

## Description

This is not to be called by user, for the lack of data-type checking mechanism.

## Usage

```
VolumeToZero(df)
```

## Arguments

| | |
|---|---|
| `df` | object whose NAs to be replaced 0s. |

## Value

A dataframe, or matrix depending on the class of the argument.

# Index