

# jhTAlib

Joost Hoeks

2019-08-21

## Contents

<b>jhTAlib</b>	<b>2</b>
Depends only on	2
Docs	2
Install	3
Update	3
Examples	3
Example 1	3
Example 2	4
Example 3	4
Example 4	4
Example 5	4
Example 6	4
Example 7	5
Example 8	5
Example 9	5
Example 10	5
Example 11	6
Test	6
Reference	6
Behavioral Techniques	6
Candlestick	9
Cycle Indicators	11
Data	12
Event Driven	13
Experimental	13
General	15
Information	16
Math Functions	17
Momentum Indicators	21
Overlap Studies	25
Pattern Recognition	28
Price Transform	32

Statistic Functions . . . . .	33
Uncategorised . . . . .	36
Volatility Indicators . . . . .	37
Volume Indicators . . . . .	38
Notebooks . . . . .	39
Recession Probability . . . . .	39
Donation and Funding . . . . .	39

## jhTAlib

Technical Analysis Library Time-Series

You can use and import it for your:

- Technical Analysis Software
- Charting Software
- Backtest Software
- Trading Robot Software
- Trading Software in general

Work in progress...

---

## Depends only on

- The Python Standard Library

---

## Docs

- .html
- .epub
- .json
- .odt
- .pdf
- .rst
- .rtf

- .xml
- 

## Install

From PyPI:

```
$ [sudo] pip3 install jhtalib
```

From source:

```
$ git clone https://github.com/joosthoeks/jhTAlib.git
$ cd jhTAlib
$ [sudo] pip3 install -e .
```

---

## Update

From PyPI:

```
$ [sudo] pip3 install --upgrade jhtalib
```

From source:

```
$ cd jhTAlib
$ git pull [upstream master]
```

---

## Examples

```
$ cd example/
```

### Example 1

```
$ python3 example-1-plot.py
```

or

<https://colab.research.google.com/github/joosthoeks/jhTAlib/blob/master/example/example-1-plot.ipynb>

---

### **Example 2**

```
$ python3 example-2-plot.py
```

or

<https://colab.research.google.com/github/joosthoeks/jhTAlib/blob/master/example/example-2-plot.ipynb>

---

### **Example 3**

```
$ python3 example-3-plot.py
```

or

<https://colab.research.google.com/github/joosthoeks/jhTAlib/blob/master/example/example-3-plot.ipynb>

---

### **Example 4**

```
$ python3 example-4-plot-quandl.py
```

or

<https://colab.research.google.com/github/joosthoeks/jhTAlib/blob/master/example/example-4-plot-quandl.ipynb>

---

### **Example 5**

```
$ python3 example-5-plot-quandl.py
```

or

<https://colab.research.google.com/github/joosthoeks/jhTAlib/blob/master/example/example-5-plot-quandl.ipynb>

---

### **Example 6**

```
$ python3 example-6-plot-quandl.py
```

or

<https://colab.research.google.com/github/joosthoeks/jhTAlib/blob/master/example/example-6-plot-quandl.ipynb>

---

### **Example 7**

`$ python3 example-7-quandl-2-df.py`

or

<https://colab.research.google.com/github/joosthoeks/jhTAlib/blob/master/example/example-7-quandl-2-df.ipynb>

---

### **Example 8**

`$ python3 example-8-alphavantage-2-df.py`

or

<https://colab.research.google.com/github/joosthoeks/jhTAlib/blob/master/example/example-8-alphavantage-2-df.ipynb>

---

### **Example 9**

`$ python3 example-9-cryptocompare-2-df.py`

or

<https://colab.research.google.com/github/joosthoeks/jhTAlib/blob/master/example/example-9-cryptocompare-2-df.ipynb>

---

### **Example 10**

DF NumPy Pandas

<https://colab.research.google.com/github/joosthoeks/jhTAlib/blob/master/example/example-10-df-numpy-pandas.ipynb>

---

## Example 11

Basic Usage

<https://colab.research.google.com/github/joosthoeks/jhTAlib/blob/master/example/example-11-basic-usage.ipynb>

---

## Test

```
$ cd test/  
$ python3 test.py
```

---

## Reference

```
import jhtalib as jhta
```

## Behavioral Techniques

### ATH | All Time High | DONE

- dict of lists of floats = `jhta.ATH(df, price='High')`
- 

### LMC | Last Major Correction | DONE

- dict of lists of floats = `jhta.LMC(df, price='Low', price_high='High')`
- 

### PP | Pivot Point | DONE

- dict of lists of floats = `jhta.PP(df, high='High', low='Low', close='Close')`
  - [https://en.wikipedia.org/wiki/Pivot\\_point\\_\(technical\\_analysis\)](https://en.wikipedia.org/wiki/Pivot_point_(technical_analysis))
-

### **FIBOPR | Fibonacci Price Retracements | DONE**

- dict of lists of floats = `jhta.FIBOPR(df, price='Close')`
- 

### **FIBTR | Fibonacci Time Retracements |**

- 
- 

### **GANNPR | W. D. Gann Price Retracements | DONE**

- dict of lists of floats = `jhta.GANNPR(df, price='Close')`
- 

### **GANNTR | W. D. Gann Time Retracements |**

- 
- 

### **JDN | Julian Day Number | DONE**

- `jdn = jhta.JDN(utc_year, utc_month, utc_day)`
  - [https://en.wikipedia.org/wiki/Julian\\_day](https://en.wikipedia.org/wiki/Julian_day)
- 

### **JD | Julian Date | DONE**

- `jd = jhta.JD(utc_year, utc_month, utc_day, utc_hour, utc_minute, utc_second)`
  - [https://en.wikipedia.org/wiki/Julian\\_day](https://en.wikipedia.org/wiki/Julian_day)
- 

### **SUNC | Sun Cycle |**

- 
-

MERCURYC | Mercury Cycle |

•

---

VENUSC | Venus Cycle |

•

---

EARTH C | Earth Cycle |

•

---

MARSC | Mars Cycle |

•

---

JUPITERC | Jupiter Cycle |

•

---

SATURN C | Saturn Cycle |

•

---

URANUS C | Uranus Cycle |

•

---

NEPTUNE C | Neptune Cycle |

•

---



## PLUTOC | Pluto Cycle |

- 
- 

## MOONC | Moon Cycle |

- 
- 

## Candlestick

### CDLBODYYS | Candle Body Size | DONE

- `list of floats = jhta.CDLBODYYS(df, open='Open', close='Close')`
  - <https://www.tradeciety.com/understand-candlesticks-patterns/>
- 

### CDLWICKS | Candle Wick Size | DONE

- `list of floats = jhta.CDLWICKS(df, high='High', low='Low')`
  - <https://www.tradeciety.com/understand-candlesticks-patterns/>
- 

### CDLUPPSHAS | Candle Upper Shadow Size | DONE

- `list of floats = jhta.CDLUPPSHAS(df, open='Open', high='High', close='Close')`
  - <https://www.tradeciety.com/understand-candlesticks-patterns/>
- 

### CDLLOWSHAS | Candle Lower Shadow Size | DONE

- `list of floats = jhta.CDLLOWSHAS(df, open='Open', low='Low', close='Close')`
  - <https://www.tradeciety.com/understand-candlesticks-patterns/>
-

### **CDLBODYP | Candle Body Percent | DONE**

- `list of floats = jhta.CDLBODYP(df, open='Open', close='Close')`
- 

### **CDLBODYM | Candle Body Momentum | DONE**

- `list of floats = jhta.CDLBODYM(df, n, open='Open', close='Close')`
  - book: Trading Systems and Methods
- 

### **GAP | Gap | DONE**

- `list of floats = jhta.GAP(df, high='High', low='Low', close='Close')`
- 

### **QSTICK | Qstick | DONE**

- `list of floats = jhta.QSTICK(df, n, open='Open', close='Close')`
  - <https://www.fmlabs.com/reference/default.htm?url=Qstick.htm>
- 

### **SHADOWT | Shadow Trends | DONE**

- `dict of lists of floats = jhta.SHADOWT(df, n, open='Open', high='High', low='Low', close='Close')`
  - book: The New Technical Trader
- 

### **IMI | Intraday Momentum Index | DONE**

- `list of floats = jhta.IMI(df, open='Open', close='Close')`
  - <https://www.fmlabs.com/reference/default.htm?url=IMI.htm>
-

## Cycle Indicators

**HT\_DCPERIOD | Hilbert Transform - Dominant Cycle Period |**

- 

---

**HT\_DCPHASE | Hilbert Transform - Dominant Cycle Phase |**

- 

---

**HT\_PHASOR | Hilbert Transform - Phasor Components |**

- 

---

**HT\_SINE | Hilbert Transform - SineWave |**

- 

---

**HT\_TRENDLINE | Hilbert Transform - Instantaneous Trendline |**

- 

---

**HT\_TRENDMODE | Hilbert Transform - Trend vs Cycle Mode |**

- 

---

**TS | Trend Score | DONE**

- list of floats = jhta.TS(df, n, price='Close')
- <https://www.fmlabs.com/reference/default.htm?url=TrendScore.htm>

---

## Data

### CSV2DF | CSV file 2 DataFeed | DONE

- dict of tuples of floats = `jhta.CSV2DF(csv_file_path, datetime='datetime', Open='Open', high='High', low='Low', close='Close', volume='Volume')`
- 

### CSVURL2DF | CSV file url 2 DataFeed | DONE

- dict of tuples of floats = `jhta.CSVURL2DF(csv_file_url, datetime='datetime', open='Open', high='High', low='Low', close='Close', volume='Volume')`
- 

### DF2CSV | DataFeed 2 CSV file | DONE

- csv file = `jhta.DF2CSV(df, csv_file_path, datetime='datetime', Open='Open', high='High', low='Low', close='Close', volume='Volume')`
- 

### DF2DFREV | DataFeed 2 DataFeed Reversed | DONE

- dict of tuples of floats = `jhta.DF2DFREV(df, datetime='datetime', open='Open', high='High', low='Low', close='Close', volume='Volume')`
- 

### DF2DFWIN | DataFeed 2 DataFeed Window | DONE

- dict of tuples of floats = `jhta.DF2DFWIN(df, start=0, end=10, datetime='datetime', open='Open', high='High', low='Low', close='Close', volume='Volume')`
- 

### DF\_HEAD | DataFeed HEAD | DONE

- dict of tuples of floats = `jhta.DF_HEAD(df, n=5, datetime='datetime', open='Open', high='High', low='Low', close='Close', volume='Volume')`
-

#### **DF\_TAIL | DataFeed TAIL | DONE**

- dict of tuples of floats = `jhta.DF_TAIL(df, n=5, datetime='datetime', open='Open', high='High', low='Low', close='Close', volume='Volume')`
- 

#### **DF2HEIKIN\_ASHI | DataFeed 2 Heikin-Ashi DataFeed | DONE**

- dict of tuples of floats = `jhta.DF2HEIKIN_ASHI(df, datetime='datetime', open='Open', high='High', low='Low', close='Close', volume='Volume')`
- 

#### **Event Driven**

##### **ASI | Accumulation Swing Index (J. Welles Wilder) | DONE**

- list of floats = `jhta.ASI(df, L, open='Open', high='High', low='Low', close='Close')`
  - book: New Concepts in Technical Trading Systems
- 

##### **SI | Swing Index (J. Welles Wilder) | DONE**

- list of floats = `jhta.SI(df, L, open='Open', high='High', low='Low', close='Close')`
  - book: New Concepts in Technical Trading Systems
- 

#### **Experimental**

##### **JH\_SAVGP | Swing Average Price - previous Average Price | DONE**

- list of floats = `jhta.JH_SAVGP(df, open='Open', high='High', low='Low', close='Close')`
- 

##### **JH\_SAVGPS | Swing Average Price - previous Average Price Summation | DONE**

- list of floats = `jhta.JH_SAVGPS(df, open='Open', high='High', low='Low', close='Close')`

---

**JH\_SCO | Swing Close - Open | DONE**

- list of floats = `jhta.JH_SCO(df, open='Open', close='Close')`
- 

**JH\_SCOS | Swing Close - Open Summation | DONE**

- list of floats = `jhta.JH_SCOS(df, open='Open', close='Close')`
- 

**JH\_SMEDP | Swing Median Price - previous Median Price | DONE**

- list of floats = `jhta.JH_SMEDP(df, high='High', low='Low')`
- 

**jh\_SMEDPS | Swing Median Price - previous Median Price Summation | DONE**

- list of floats = `jhta.JH_SMEDPS(df, high='High', low='Low')`
- 

**JH\_SPP | Swing Price - previous Price | DONE**

- list of floats = `jhta.JH_SPP(df, price='Close')`
- 

**JH\_SPPS | Swing Price - previous Price Summation | DONE**

- list of floats = `jhta.JH_SPPS(df, price='Close')`
- 

**JH\_STYPP | Swing Typical Price - previous Typical Price | DONE**

- list of floats = `jhta.JH_STYPP(df, high='High', low='Low', close='Close')`
-

**JH\_STYPPS | Swing Typical Price - previous Typical Price Summation | DONE**

- `list of floats = jhta.JH_STYPPS(df, high='High', low='Low', close='Close')`
- 

**JH\_SWCLP | Swing Weighted Close Price - previous Weighted Close Price | DONE**

- `list of floats = jhta.JH_SWCLP(df, high='High', low='Low', close='Close')`
- 

**JH\_SWCLPS | Swing Weighted Close Price - previous Weighted Close Price Summation | DONE**

- `list of floats = jhta.JH_SWCLPS(df, high='High', low='Low', close='Close')`
- 

**General**

**NORMALIZE | Normalize | DONE**

- `list of floats = jhta.NORMALIZE(df, price_max='High', price_min='Low', price='Close')`
  - <https://machinelearningmastery.com/normalize-standardize-time-series-data-python/>
- 

**STANDARDIZE | Standardize | DONE**

- `list of floats = jhta.STANDARDIZE(df, price='Close')`
  - <https://machinelearningmastery.com/normalize-standardize-time-series-data-python/>
-

### **RATIO | Ratio | DONE**

- `list of floats = jhta.RATIO(df1, df2, price1='Close', price2='Close')`
  - <https://www.fmlabs.com/reference/default.htm?url=Ratio.htm>
- 

### **SPREAD | Spread | DONE**

- `list of floats = jhta.SPREAD(df1, df2, price1='Close', price2='Close')`
- 

### **CP | Comparative Performance | DONE**

- `list of floats = jhta.CP(df1, df2, price1='Close', price2='Close')`
  - <https://www.fmlabs.com/reference/default.htm?url=CompPerformance.htm>
- 

### **CRSI | Comparative Relative Strength Index | DONE**

- `list of floats = jhta.CRSI(df1, df2, n, price1='Close', price2='Close')`
  - <https://www.fmlabs.com/reference/default.htm?url=RSIC.htm>
- 

### **CS | Comparative Strength | DONE**

- `list of floats = jhta.CS(df1, df2, price1='Close', price2='Close')`
  - <https://www.fmlabs.com/reference/default.htm?url=CompStrength.htm>
- 

### **Information**

#### **INFO | Print df Information | DONE**

- `print = jhta.INFO(df, price='Close')`
-



### INFO\_TRADES | Print Trades Information | DONE

- `print = jhta.INFO_TRADES(profit_trades_list, loss_trades_list)`
- 

### Math Functions

#### EXP | Exponential | DONE

- `list of floats = jhta.EXP(df, price='Close')`
- 

#### LOG | Logarithm | DONE

- `list of floats = jhta.LOG(df, price='Close')`
- 

#### LOG10 | Base-10 Logarithm | DONE

- `list of floats = jhta.LOG10(df, price='Close')`
- 

#### SQRT | Square Root | DONE

- `list of floats = jhta.SQRT(df, price='Close')`
- 

#### ACOS | Arc Cosine | DONE

- `list of floats = jhta.ACOS(df, price='Close')`
- 

#### ASIN | Arc Sine | DONE

- `list of floats = jhta.ASIN(df, price='Close')`
- 

#### ATAN | Arc Tangent | DONE

- `list of floats = jhta.ATAN(df, price='Close')`
-

#### **COS | Cosine | DONE**

- `list of floats = jhta.COS(df, price='Close')`
- 

#### **SIN | Sine | DONE**

- `list of floats = jhta.SIN(df, price='Close')`
- 

#### **TAN | Tangent | DONE**

- `list of floats = jhta.TAN(df, price='Close')`
- 

#### **ACOSH | Inverse Hyperbolic Cosine | DONE**

- `list of floats = jhta.ACOSH(df, price='Close')`
- 

#### **ASINH | Inverse Hyperbolic Sine | DONE**

- `list of floats = jhta.ASINH(df, price='Close')`
- 

#### **ATANH | Inverse Hyperbolic Tangent | DONE**

- `list of floats = jhta.ATANH(df, price='Close')`
- 

#### **COSH | Hyperbolic Cosine | DONE**

- `list of floats = jhta.COSH(df, price='Close')`
- 

#### **SINH | Hyperbolic Sine | DONE**

- `list of floats = jhta.SINH(df, price='Close')`
-

#### **TANH | Hyperbolic Tangent | DONE**

- `list of floats = jhta.TANH(df, price='Close')`
- 

#### **PI | Mathematical constant PI | DONE**

- `float = jhta.PI()`
- 

#### **E | Mathematical constant E | DONE**

- `float = jhta.E()`
- 

#### **TAU | Mathematical constant TAU | DONE**

- `float = jhta.TAU()`
- 

#### **PHI | Mathematical constant PHI | DONE**

- `float = jhta.PHI()`
- 

#### **FIB | Fibonacci series up to n | DONE**

- `list of ints = jhta.FIB(n)`
- 

#### **CEIL | Ceiling | DONE**

- `list of floats = jhta.CEIL(df, price='Close')`
- 

#### **FLOOR | Floor | DONE**

- `list of floats = jhta.FLOOR(df, price='Close')`
-

#### DEGREES | Radians to Degrees | DONE

- `list of floats = jhta.DEGREES(df, price='Close')`
- 

#### RADIANS | Degrees to Radians | DONE

- `list of floats = jhta.RADIANS(df, price='Close')`
- 

#### ADD | Addition High + Low | DONE

- `list of floats = jhta.ADD(df, high='High', low='Low')`
- 

#### DIV | Division High / Low | DONE

- `list of floats = jhta.DIV(df, high='High', low='Low')`
- 

#### MAX | Highest value over a specified period | DONE

- `list of floats = jhta.MAX(df, n, price='Close')`
- 

#### MAXINDEX | Index of highest value over a specified period | DONE

- `list of ints = jhta.MAXINDEX(df, n, price='Close')`
- 

#### MIN | Lowest value over a specified period | DONE

- `list of floats = jhta.MIN(df, n, price='Close')`
- 

#### MININDEX | Index of lowest value over a specified period | DONE

- `list of ints = jhta.MININDEX(df, n, price='Close')`
-

**MINMAX | Lowest and Highest values over a specified period | DONE**

- dict of lists of floats = `jhta.MINMAX(df, n, price='Close')`
- 

**MINMAXINDEX | Indexes of lowest and highest values over a specified period | DONE**

- dict of lists of ints = `jhta.MINMAXINDEX(df, n, price='Close')`
- 

**MULT | Multiply High \* Low | DONE**

- list of floats = `jhta.MULT(df, high='High', low='Low')`
- 

**SUB | Subtraction High - Low | DONE**

- list of floats = `jhta.SUB(df, high='High', low='Low')`
- 

**SUM | Summation | DONE**

- list of floats = `jhta.SUM(df, n, price='Close')`
- 

**Momentum Indicators**

**ADX | Average Directional Movement Index |**

- 
- 

**ADXR | Average Directional Movement Index Rating |**

- 
-

### APO | Absolute Price Oscillator | DONE

- `list of floats = jhta.APO(df, n_fast, n_slow, price='Close')`
  - <https://www.fmlabs.com/reference/default.htm?url=PriceOscillator.htm>
- 

### AROON | Aroon |

- 
- 

### AROONOSC | Aroon Oscillator |

- 
- 

### BOP | Balance Of Power |

- 
- 

### CCI | Commodity Channel Index |

- 
- 

### CMO | Chande Momentum Oscillator |

- 
- 

### DX | Directional Movement Index |

- 
- 

### MACD | Moving Average Convergence/Divergence |

- 
-

**MACDEXT | MACD with controllable MA type |**

- 

---

**MACDFIX | Moving Average Convergence/Divergence Fix 12/26 |**

- 

---

**MFI | Money Flow Index |**

- 

---

**MINUS\_DI | Minus Directional Indicator |**

- 

---

**MINUS\_DM | Minus Directional Movement |**

- 

---

**MOM | Momentum | DONE**

- list of floats = jhta.MOM(df, n, price='Close')
- <https://www.fmlabs.com/reference/default.htm?url=Momentum.htm>

---

**PLUS\_DI | Plus Directional Indicator |**

- 

---

**PLUS\_DM | Plus Directional Movement |**

- 

---

### **PPO | Percentage Price Oscillator |**

- 
- 

### **RMI | Relative Momentum Index | DONE**

- `list of floats = jhta.RMI(df, n, price='Close')`
  - <https://www.fmlabs.com/reference/default.htm?url=RMI.htm>
- 

### **ROC | Rate of Change | DONE**

- `list of floats = jhta.ROC(df, n, price='Close')`
- 

### **ROCP | Rate of Change Percentage | DONE**

- `list of floats = jhta.ROCP(df, n, price='Close')`
- 

### **ROCR | Rate of Change Ratio | DONE**

- `list of floats = jhta.ROCR(df, n, price='Close')`
- 

### **ROCR100 | Rate of Change Ratio 100 scale | DONE**

- `list of floats = jhta.ROCR100(df, n, price='Close')`
  - <https://www.fmlabs.com/reference/default.htm?url=RateOfChange.htm>
- 

### **RSI | Relative Strength Index | DONE**

- `list of floats = jhta.RSI(df, n, price='Close')`
  - <https://www.fmlabs.com/reference/default.htm?url=rsi.htm>
-



### STOCH | Stochastic | DONE

- `list of floats = jhta.STOCH(df, n, price='Close')`
  - <https://www.fmlabs.com/reference/default.htm?url=Stochastic.htm>
- 

### STOCHF | Stochastic Fast |

- 
- 

### STOCHRSI | Stochastic Relative Strength Index |

- 
- 

### TRIX | 1-day Rate-Of-Change (ROC) of a Triple Smooth EMA |

- 
- 

### ULTOSC | Ultimate Oscillator |

- 
- 

### WILLR | Williams' %R | DONE

- `list of floats = jhta.WILLR(df, n, high='High', low='Low', close='Close')`
  - <https://www.fmlabs.com/reference/default.htm?url=WilliamsR.htm>
- 

### Overlap Studies

### BBANDS | Bollinger Bands | DONE

- `dict of lists of floats = jhta.BBANDS(df, n, f=2, high='High', low='Low', close='Close')`
- <https://www.fmlabs.com/reference/default.htm?url=Bollinger.htm>

---

### **BBANDW | Bollinger Band Width | DONE**

- `list of floats = jhta.BBANDW(df, n, f=2, high='High', low='Low', close='Close')`
  - <https://www.fmlabs.com/reference/default.htm?url=BollingerWidth.htm>
- 

### **DEMA | Double Exponential Moving Average |**

- 
- 

### **EMA | Exponential Moving Average | DONE**

- `list of floats = jhta.EMA(df, n, price='Close')`
  - <https://www.fmlabs.com/reference/default.htm?url=ExpMA.htm>
- 

### **ENVP | Envelope Percent | DONE**

- `dict of lists of floats = jhta.ENVP(df, pct=.01, price='Close')`
  - <https://www.fmlabs.com/reference/default.htm?url=EnvelopePct.htm>
- 

### **KAMA | Kaufman Adaptive Moving Average |**

- 
- 

### **MA | Moving Average |**

- 
-

### **MAMA | MESA Adaptive Moving Average |**

- 
- 

### **MAVP | Moving Average with Variable Period |**

- 
- 

### **MIDPOINT | MidPoint over period | DONE**

- `list of floats = jhta.MIDPOINT(df, n, price='Close')`
  - <http://www.tadoc.org/indicator/MIDPOINT.htm>
- 

### **MIDPRICE | MidPoint Price over period | DONE**

- `list of floats = jhta.MIDPRICE(df, n, high='High', low='Low')`
  - <http://www.tadoc.org/indicator/MIDPRICE.htm>
- 

### **MMR | Mayer Multiple Ratio | DONE**

- `list of floats = jhta.MMR(df, n=200, price='Close')`
  - <https://www.theinvestorspodcast.com/bitcoin-mayer-multiple/>
- 

### **SAR | Parabolic SAR | DONE**

- `list of floats = jhta.SAR(df, af_step=.02, af_max=.2, high='High', low='Low')`
  - book: New Concepts in Technical Trading Systems
- 

### **SAREXT | Parabolic SAR - Extended |**

- 
-

### SMA | Simple Moving Average | DONE

- `list of floats = jhta.SMA(df, n, price='Close')`
  - <https://www.fmlabs.com/reference/default.htm?url=SimpleMA.htm>
- 

### T3 | Triple Exponential Moving Average (T3) |

- 
- 

### TEMA | Triple Exponential Moving Average |

- 
- 

### TRIMA | Triangular Moving Average | DONE

- `list of floats = jhta.TRIMA(df, n, price='Close')`
  - <https://www.fmlabs.com/reference/default.htm?url=TriangularMA.htm>
- 

### WMA | Weighted Moving Average

- 
- 

### Pattern Recognition

#### CDL2CROWS | Two Crows |

#### CDL3BLACKCROWS | Three Black Crows |

#### CDL3INSIDE | Three Inside Up/Down |

#### CDL3LINESTRIKE | Three-Line Strike |

#### CDL3OUTSIDE | Three Outside Up/Down |

CDL3STARSINSOUTH | Three Stars In The South |

CDL3WHITESOLDIERS | Three Advancing White Soldiers |

CDLABANDONEDBABY | Abandoned Baby |

CDLADVANCEBLOCK | Advance Block |

CDLBELTHOLD | Belt-hold |

CDLBREAKAWAY | Breakaway |

CDLCLOSINGMARUBOZU | Closing Marubozu |

CDLCONSEALBABYSWALL | Concealing Baby Swallow |

CDLCOUNTERATTACK | Counterattack |

CDLDARKCLOUDCOVER | Dark Cloud Cover |

CDLDOJI | Doji |

CDLDOJISTAR | Doji Star |

CDLDRAGONFLYDOJI | Dragonfly Doji |

CDLENGULFING | Engulfing Pattern |

CDLEVENINGDOJISTAR | Evening Doji Star |

CDLEVENINGSTAR | Evening Star |

CDLGAPSIDESIDEWHITE | Up/Down-gap side-by-side white lines  
|

CDLGRAVESTONEDOJI | Gravestone Doji |

CDLHAMMER | Hammer |

CDLHANGINGMAN | Hanging Man |

CDLHARAMI | Harami Pattern |

CDLHARAMICROSS | Harami Cross Pattern |

CDLHIGHWAVE | High-Wave Candle |

CDLHIKKAKE | Hikkake Pattern |

CDLHIKKAKEMOD | Modified Hikkake Pattern |

CDLHOMINGPIGEON | Homing Pigeon |

CDLIDENTICAL3CROWS | Identical Three Crows |

CDLINNECK | In-Neck Pattern |

CDLINVERTEDHAMMER | Inverted Hammer |

CDLKICKING | Kicking |

CDLKICKINGBYLENGTH | Kicking - bull/bear determined by the longer marubozu |

CDLLADDERBOTTOM | Ladder Bottom |

CDLLONGLEGGEDDOJI | Long Legged Doji |

CDLLONGLINE | Long Line Candle |

CDLMARUBOZU | Marubozu |

CDLMATCHINGLOW | Matching Low |

CDLMATHOLD | Mat Hold |

CDLMORNINGDOJISTAR | Morning Doji Star |

CDLMORNINGSTAR | Morning Star |

CDLONNECK | On-Neck Pattern |

CDLPIERCING | Piercing Pattern |

CDLRICKSHAWMAN | Rickshaw Man |

CDLRISEFALL3METHODS | Rising/Falling Three Methods |

CDLSEPARATINGLINES | Separating Lines |

CDLSHOOTINGSTAR | Shooting Star |

CDLSHORTLINE | Short Line Candle |

CDLSPINNINGTOP | Spinning Top |

CDLSTALLEDPATTERN | Stalled Pattern |

CDLTICKSANDWICH | Stick Sandwich |

CDLTAKURI | Takuri (Dragonfly Doji with very long lower shadow)  
|

CDLTASUKIGAP | Tasuki Gap |

**CDLTHRUSTING | Thrusting Pattern |**

**CDLTRISTAR | Tristar Pattern |**

**CDLUNIQUE3RIVER | Unique 3 River |**

**CDLUPSIDEGAP2CROWS | Upside Gap Two Crows |**

**CDLXSIDEGAP3METHODS | Upside/Downside Gap Three Methods |**

**Price Transform**

**AVGPRICE | Average Price | DONE**

- list of floats = `jhta.AVGPRICE(df, open='Open', high='High', low='Low', close='Close')`
  - <https://www.fmlabs.com/reference/default.htm?url=AvgPrices.htm>
- 

**MEDPRICE | Median Price | DONE**

- list of floats = `jhta.MEDPRICE(df, high='High', low='Low')`
  - <https://www.fmlabs.com/reference/default.htm?url=MedianPrices.htm>
- 

**TYPPRICE | Typical Price | DONE**

- list of floats = `jhta.TYPPRICE(df, high='High', low='Low', close='Close')`
  - <https://www.fmlabs.com/reference/default.htm?url=TypicalPrices.htm>
- 

**WCLPRICE | Weighted Close Price | DONE**

- list of floats = `jhta.WCLPRICE(df, high='High', low='Low', close='Close')`
- <https://www.fmlabs.com/reference/default.htm?url=WeightedCloses.htm>



---

## Statistic Functions

### MEAN | Arithmetic mean (average) of data | DONE

- list of floats = `jhta.MEAN(df, n, price='Close')`
- 

### HARMONIC\_MEAN | Harmonic mean of data | DONE

- list of floats = `jhta.HARMONIC_MEAN(df, n, price='Close')`
- 

### MEDIAN | Median (middle value) of data | DONE

- list of floats = `jhta.MEDIAN(df, n, price='Close')`
- 

### MEDIAN\_LOW | Low median of data | DONE

- list of floats = `jhta.MEDIAN_LOW(df, n, price='Close')`
- 

### MEDIAN\_HIGH | High median of data | DONE

- list of floats = `jhta.MEDIAN_HIGH(df, n, price='Close')`
- 

### MEDIAN\_GROUPED | Median, or 50th percentile, of grouped data | DONE

- list of floats = `jhta.MEDIAN_GROUPED(df, n, price='Close', interval=1)`
- 

### MODE | Mode (most common value) of discrete data | DONE

- list of floats = `jhta.MODE(df, n, price='Close')`
-

### **PSTDEV | Population standard deviation of data | DONE**

- `list of floats = jhta.PSTDEV(df, n, price='Close', mu=None)`
- 

### **PVARIANCE | Population variance of data | DONE**

- `list of floats = jhta.PVARIANCE(df, n, price='Close', mu=None)`
- 

### **STDEV | Sample standard deviation of data | DONE**

- `list of floats = jhta.STDEV(df, n, price='Close', xbar=None)`
- 

### **VARIANCE | Sample variance of data | DONE**

- `list of floats = jhta.VARIANCE(df, n, price='Close', xbar=None)`
- 

### **COV | Covariance | DONE**

- `float = jhta.COV(list1, list2)`
  - [https://en.wikipedia.org/wiki/Algorithms\\_for\\_calculating\\_variance#Covariance](https://en.wikipedia.org/wiki/Algorithms_for_calculating_variance#Covariance)
- 

### **COVARIANCE | Covariance | DONE**

- `list of floats = jhta.COVARANCE(df1, df2, n, price1='Close', price2='Close')`
  - [https://en.wikipedia.org/wiki/Algorithms\\_for\\_calculating\\_variance#Covariance](https://en.wikipedia.org/wiki/Algorithms_for_calculating_variance#Covariance)
- 

### **COR | Correlation | DONE**

- `float = jhta.COR(list1, list2)`
-

### **CORRELATION | Correlation | DONE**

- `list of floats = jhta.CORRELATION(df1, df2, n, price1='Close', price2='Close')`
- 

### **PCOR | Population Correlation | DONE**

- `float = jhta.PCOR(list1, list2)`
- 

### **PCORRELATION | Population Correlation | DONE**

- `list of floats = jhta.PCORRELATION(df1, df2, n, price1='Close', price2='Close')`
- 

### **BETA | Beta | DONE**

- `float = jhta.BETA(list1, list2)`
  - [https://en.wikipedia.org/wiki/Beta\\_\(finance\)](https://en.wikipedia.org/wiki/Beta_(finance))
- 

### **BETAS | Betas | DONE**

- `list of floats = jhta.BETAS(df1, df2, n, price1='Close', price2='Close')`
  - [https://en.wikipedia.org/wiki/Beta\\_\(finance\)](https://en.wikipedia.org/wiki/Beta_(finance))
- 

### **LSR | Least Squares Regression | DONE**

- `list of floats = jhta.LSR(df, price='Close', predictions_int=0)`
  - <https://www.mathsisfun.com/data/least-squares-regression.html>
-

### **SLR | Simple Linear Regression | DONE**

- `list of floats = jhta.SLR(df, price='Close', predictions_int=0)`
  - <https://machinelearningmastery.com/implement-simple-linear-regression-scratch-python/>
- 

### **SLOPE | Slope | DONE**

- `float = jhta.SLOPE(x1, y1, x2, y2)`
  - book: An Introduction to Algorithmic Trading
- 

### **SLOPES | Slopes | DONE**

- `list of floats = jhta.SLOPES(df, n, price='Close')`
  - book: An Introduction to Algorithmic Trading
- 

### **Uncategorised**

### **HR | Hit Rate / Win Rate | DONE**

- `float = jhta.HR(hit_trades_int, total_trades_int)`
  - <http://traderskillset.com/hit-rate-stock-trading/>
- 

### **PLR | Profit/Loss Ratio | DONE**

- `float = jhta.PLR(mean_trade_profit_float, mean_trade_loss_float)`
  - [https://www.investopedia.com/terms/p/profit\\_loss\\_ratio.asp](https://www.investopedia.com/terms/p/profit_loss_ratio.asp)
- 

### **EV | Expected Value | DONE**

- `float = jhta.EV(hittrade_float, mean_trade_profit_float, mean_trade_loss_float)`
  - [https://en.wikipedia.org/wiki/Expected\\_value](https://en.wikipedia.org/wiki/Expected_value)
-

### **POR | Probability of Ruin (Table of Lucas and LeBeau) | DONE**

- `int = jhta.POR(hittrade_float, profit_loss_ratio_float)`
  - book: Computer Analysis of the Futures Markets
- 

### **BPPS | Basis Points per Second | DONE**

- `float = jhta.BPPS(trade_start_price, trade_end_price, trade_start_timestamp, trade_end_timestamp)`
  - book: An Introduction to Algorithmic Trading
- 

### **Volatility Indicators**

#### **AEM | Arms Ease of Movement | DONE**

- `list of floats = jhta.AEM(df, high='High', low='Low', volume='Volume')`
  - <https://www.fmlabs.com/reference/default.htm?url=ArmsEMV.htm>
- 

#### **ATR | Average True Range | DONE**

- `list of floats = jhta.ATR(df, n, high='High', low='Low', close='Close')`
  - <https://www.fmlabs.com/reference/default.htm?url=ATR.htm>
- 

#### **NATR | Normalized Average True Range |**

- 
- 

#### **RVI | Relative Volatility Index | DONE**

- `list of floats = jhta.RVI(df, n, high='High', low='Low')`
  - <https://www.fmlabs.com/reference/default.htm?url=RVI.htm>
-

## **INERTIA | Inertia |**

- 
- 

## **TRANGE | True Range | DONE**

- `list of floats = jhta.TRANGE(df, high='High', low='Low', close='Close')`
  - <https://www.fmlabs.com/reference/default.htm?url=TR.htm>
- 

## **Volume Indicators**

### **AD | Chaikin A/D Line | DONE**

- `list of floats = jhta.AD(df, high='High', low='Low', close='Close', volume='Volume')`
  - <https://www.fmlabs.com/reference/default.htm?url=AccumDist.htm>
- 

### **ADOSC | Chaikin A/D Oscillator |**

- 
- 

### **OBV | On Balance Volume | DONE**

- `list of floats = jhta.OBV(df, close='Close', volume='Volume')`
  - <https://www.fmlabs.com/reference/default.htm?url=OBV.htm>
- 

### **PVR | Price Volume Rank | DONE**

- `list of ints = jhta.PVR(df, price='Close', volume='Volume')`
  - <https://www.fmlabs.com/reference/default.htm?url=PVrank.htm>
-

### **PVT | Price Volume Trend | DONE**

- `list of floats = jhta.PVT(df, price='Close', volume='Volume')`
  - <https://www.fmlabs.com/reference/default.htm?url=PVT.htm>
- 

### **PVI | Positive Volume Index | DONE**

- `list of floats = jhta.PVI(df, price='Close', volume='Volume')`
  - <https://www.fmlabs.com/reference/default.htm?url=PVI.htm>
- 

### **NVI | Negative Volume Index | DONE**

- `list of floats = jhta.NVI(df, price='Close', volume='Volume')`
  - <https://www.fmlabs.com/reference/default.htm?url=NVI.htm>
- 

## **Notebooks**

- <https://github.com/joosthoeks/jhTALib/tree/master/notebooks>

### **Recession Probability**

- [https://colab.research.google.com/github/joosthoeks/jhTALib/blob/master/notebooks/recession\\_probability.ipynb](https://colab.research.google.com/github/joosthoeks/jhTALib/blob/master/notebooks/recession_probability.ipynb)
- 

## **Donation and Funding**

- BTC: 3KCoXMyUDgVABoFSuV8GQT3k8qkUhEDG9X
-