Backtest zum LRBot07

Im Gegensatz zu vielen anderen Bots wird dieser Bot mit einem Backtest-Modul ausgeliefert.

Dieses Modul wurde im Lauf der Betatester-Phase sehr weit aufgebohrt und ermöglicht das Testen und Bewerten der Bot-Parameter auf unüblich hohem Niveau.

Highlights:

- automatisch viele Parametersätze am Stück durchtesten
- Wichtung der Ergebnisse nach Equity und Trading-Kennzahlen
- Vermeidung von Overfitting durch mehrere Testzeiträume pro Satz
- Pro-Version: Massive Beschleunigung durch Caching-Techniken
- Pro-Version: Unterstützung paralleler Backtests

Inhaltsverzeichnis

1. Grundsätzliches zu Backtests

DISCLAIMER: Der Backtest stellt eine ideale Welt dar, in der z.B. jede Order durchgeht, während die echte Welt trotz aller Simulationsmühen viele Störungen und Abweichungen bereithält, die die realen Ergebnisse deutlich verändern.
Ein Backtest kann somit gute Parametereinstellungen von schlechten trennen, er kann aber keine Ergebnisse vorhersagen oder im Nachhinein die realen Bot-Aktionen widerspiegeln!

Sinnvolle Backtests sind schwieriger, als man gemeinhin denkt.

Ein weit verbreiteter Irrtum ist, dass ein möglichst hohes Ergebnis beim Backtest der beste Hinweis für zukünftig erfolgreiches Traden wäre.

Tatsächlich ist eher das Gegenteil der Fall. Die höchsten Ergebnisse bei Backtests sind normalerweise die, die am besten an genau den damaligen historischen Kursverlauf angepasst sind.

Spezialisten kommen bekanntermaßen außerhalb ihres Spezialgebietes schlecht zurecht. Zum Traden brauchen wir aber Generalisten, die mit annähernd jeder Situation brauchbar zurecht kommen ...

Faustregeln und Beurteilungskriterien:

Faustregel 1:

Die allerbesten Endbalancen bei Backtests gleich wegschmeissen - das sind die Spezialisten mit dem Kurvenfitting, die nur für genau diesen historischen Kursverlauf brauchbar sind.

Faustregel 2:

Sucht euch einen genügend langen Testzeitraum aus. Idealerweise sollte der alles enthalten - Seitwärtsphasen, leichte Anstiege und leichte Abfälle, starke Anstiege und starke Abfälle.

Das ist schwierig zu erfüllen, deswegen kann man inzwischen bis zu fünf verschiedene Zeiträume angeben.

Es empfehlen sich **mindestens drei Zeiträume**:

- einer mit hauptsächlich steigenden Kursen
- einer mit hauptsälich fallenden Kursen
- einer mit seitwärts verlaufenden Kursen

Faustregel 3:

Um dem Spezialistentum zu entgehen, muss man die brauchbar scheinenden Parametersätze unbedingt auf verschiedenen Zeiträumen testen!

Viele - vor allem sehr überzeugend erscheinende - Parameterkombinationen ergeben klägliche Resultate auf anderen Zeiträumen (siehe Faustregel 1).

Siehe die Ergänzung zur Faustregel 2: Seit der Version V0.7.x kann man automatisch pro Kombination mehrere Zeiträume durchtesten lassen.
Durch die Sortierung der Ergebnisse nach der Gesamtwertung landen Spezialisten automatisch auf den hinteren Plätzen und Generalisten vorne.

Faustregel 4:

Fehler in den Kursdaten können ebenfalls zu tollen Ergebnissen führen. Wenn plötzliche Sprünge im Verlauf der Balance auftreten - egal, ob nach oben oder nach unten - sollte man misstrauisch sein und im Logfile die Trades im betreffenden Zeitraum genau überprüfen.
Ab der Version V0.7.1 werden die Kursdaten deshalb beim Laden auf Anomalien geprüft und ggfs. vom Test ausgeschlossen.

Beurteilungskriterien:

Kennzahlen zum Trading geben eine genauere Vorstellung, wie sich die Endbalance entwickelt hat - vertrauenserweckend oder mit zufälligen Sprüngen und Einbrüchen?

Folgende Kennzahlen werden vom Backtest berechnet (wem sie wenig sagen, möge sie im Internet nachschlagen):

ATH und MaxDrawdown:
Wieviel von den erreichten Gewinnen geben die Strategie und ihre Einstellungen zwischendurch wieder ab? Je kleiner der maximale Drawdown ist, desto besser.

Profitfaktor
Das Verhältnis von Gewinnen zu Verlusten. Je höher, desto besser.

Ulcer Index
Je stärker Drawdowns sind oder je länger sie dauern, desto höher wird der Ulcer Index, d.h. niedrigere Werte sind besser.

Kelly Criterion
Gibt Hinweise, wie "verlässlich" die Gewinne kommen. Je höher, desto besser.

Zusätzlich wird noch etwas Statistik zu den erfolgten Trades ausgegeben - wieviele Trades, wieviele erfolgreich, wieviele mit Verlust, durchschnittliche Gewinne und Verluste, Gesamtgewinne und -verluste.

2. Grundsätzliches zum LRBacktest

Mit dem Backtest kann man überprüfen, wie sich die Einstellungen des LRBot über einen bestimmten Zeitraum in etwa geschlagen hätten.

Nur im Bot wird abgefragt, ob die Strategie lizenziert ist. Dagegen kann man im Backtest alle mitgelieferten Strategien testen, auch die, die man nicht gekauft hat!
Für die Nutzung des Backtest reicht eine Bot-Basislizenz.

Die zu prüfende Parameter muss man dem Backtest in der Config-Datei 'lrbacktest07.ini' vorgeben.

Der Backtest wurde für die Version 0.7 stark erweitert:

- für jeden Parameter kann man mehrere Werte angeben; der Backtest erzeugt daraus selbst die verschiedenen Parametersets
- man kann bis zu fünf Testzeiträume angeben; der Backtest führt jede Parameterkombination für jeden Zeitraum aus
- der Backtest vergibt für jede Kombination pro Testzeitraum eine Wertung und sortiert die Ergebnistabelle nach der Gesamtwertung über alle Testzeiträume.

Damit wird das Suchen brauchbarer Kombinationen und das Sichten und Einordnen der Ergebnisse wesentlich erleichtert.

Sinnvollerweise wählt man mindestens drei Zeiträume aus - einen mit steigenden Kursen, einen mit fallenden Kursen und einen mit Seitwärtskursen.

Da nach Durchlaufen aller Kombinationen und aller Zeiträume nach der Gesamtwertung sortiert wird, landen Spezialisten für einen Markttypus durch ihre schlechteren Wertungen in den für sie ungeeigneten Testzeiträumen automatisch bestenfalls im Mittelfeld.

Die Gesamtwertung wird als geometrischer Durchschnitt aus den Einzelwertungen berechnet; je höher die Wertungen sind, desto besser.

In die Einzelwertung gehen die Equity dreifach, und die Werte von MaxDD, Profitfaktor und Kelly-Kriterium einfach ein.
Höhere Werte für Equity, Profitfaktor und Kelly-Kriterium verbessern die Wertung, während beim MaxDD niedrigere Werte besser sind.

Die Abschlusstabelle enthält nicht nur die Gesamtwertung, sondern auch die Wertungen und Kennziffern für jeden einzelnen Zeitraum.

Beispiel:

Drei Testzeiträume seien in der 'lrbacktest07.ini' eingetragen:

```
# TODO: period of time to test
```

```
a_startdate = 01-03-2018
a_enddate = 01-06-2018
b_startdate = 07-11-2017
b_enddate = 06-02-2018
c_startdate = 01-06-2018
c_enddate = 01-09-2018
```

Dann werden hier insgesamt 4 Tabellen erzeugt:

lr_backtest_a_in.csv
  enthält die erzeugten Parameterkombinationen
lr_backtest_a_out.csv
  enthält die Testergebnisse für Zeitraum a samt Parametern
lr_backtest_b_out.csv
  enthält die Testergebnisse für Zeitraum a bis b samt Parametern
lr_backtest_c_out.csv
  enthält die Testergebnisse für Zeitraum a bis c samt Parametern
lr_backtest_end_out.csv
  enthält die Testergebnisse für alle Zeiträume samt der Gesamtwertung

Der Backtest ermittelt aus seiner Konfigurationsdatei selbständig, welche Candles er für alle Zeiträume braucht und lädt dann fehlende Kursdaten automatisch herunter. Anschliessend startet der eigentliche Backtest.

Linux: `python3 backtest.py`
Windows: `python backtest.py`

Die Config-Datei 'lrbacktest07.ini' enthält die relevanten Werte aus der Config des Bots (lrbot07.ini) und die zusätzlichen Felder (wie den Testzeitraum), die vom Backtest benötigt werden.

Der Backtest schreibt im Verlauf des Tests pro Parametersatz und Testzeitraum eine CSV-Datei mit den Feldern time, date, equity, balance, coins, buyandhold und eine Logdatei mit den wichtigsten Konsolen-Ausgaben.
Beide Dateien enthalten als Default im Dateinamen die fortlaufende Nummer des jeweiligen Parametersets.

Wer sich intensiv mit den Backtests beschäftigen möchte, für den empfiehlt sich die Pro-Version des Backtests.

Sie ermöglicht das Aufteilen der zu testenden Parametersätze auf mehrere Dateien und nachfolgend das parallele Abarbeiten.
Zusätzlich ermöglicht sie durch das intensive Zwischenspeichern von Rechenergebnissen eine Beschleunigung um etwa den Faktor 20 und damit eine entsprechend höhere Zahl an Tests im gleichen Zeitraum.

3. Varianten beim Aufruf

Man kann den Backtest beim Aufruf anweisen, eine andere Datei als Configdatei zu verwenden. Damit kann man sich verschiedene Parametersets in Form mehrerer Configdateien vorbereiten und sie alle per Script/Batch in einem Rutsch nacheinander abarbeiten lassen.

Linux: `python3 backtest.py -c <configfile> [-o <name_für_test>]  [-nof]`
Windows: `python backtest.py -c <configfile> [-o <name_für_test>] [-nof]`

Zum Verändern der Namen der Protokollfiles (Log und CSV) des Backtests mittels des Parameters '-o <outfile>', siehe das nachfolgende Kapitel "4 Paralleles Testen".

Die Option -nof unterdrückt die Erzeugung der Protokollfiles (Log und CSV). Das ist nützlich, wenn man viele Parametersets durchtesten, vielleicht sogar noch parallel, und nur an den Endergebnissen interessiert ist.

Der Pro-Backtest erlaubt zusätzlich noch die Varianten:

```
python backtest.py -g <xxx> [-s <yyy>] [-o <name_für_test>]
```

(Erzeuge nur eine Parameterdatei mit <xxx> Parametersets, ggfs gesplittet in Einzeldateien mit <yyy> Parametersets.)

und

```
python backtest.py -r <name_für_test>_a_in.csv
```

(Starte den Backtest mit der Eingabedatei <name_für_test>_a_in.csv.)

4. Paralleles Testen (Pro-Version)

Das parallele Testen wird mit der Pro-Version des Backtests unterstützt.

Es wird ein zusätzliches Script (testmanager.py) mitgeliefert, das das parallele Testen automatisiert.

Zuerst erzeugt man eine große Menge an Parametersätzen, die automatisch in Stücke wählbarer Größe aufgeteilt werden kann.

```
python backtest.py -g <xxxxx> -o <name_für_diesen_test>
```

Dieser Befehl erzeugt eine Parameterdatei <name_für_diesen_test>_a_in.csv, die die vorgegebene Zahl von <xxxxx> Parametersets enthält.

Diese Eingabedatei könnte man zwar direkt ausführen
`python backtest.py -r <name_für_diesen_test>_a_in.csv`

müsste dabei aber zwei Nachteile in Kauf nehmen - einmal läuft der gesamte Test dann nur auf einem einzigen Kern, und zum zweiten bekommt man erst nach dem Durchlauf aller Parametersets die Ergebnistabelle.

testmanager.py erlaubt es, die Eingabedatei <name_für_diesen_test>_a_in.csv vollautomatisch in kleine Stücke zu brechen und parallel auf mehreren Kernen auszuführen. Immer, wenn so ein kleinerer Backtest abgeschlossen ist, wird automatisch der nächste gestartet.

```
python testmanager-py -r <name_fuer_diesen_test>_a_in.csv -s <yy> -t
<th>
```

(Wobei -s <yy> für die Größe der kleineren Stücke steht und -t <th> für die Zahl der Backtests, die gleichzeitig laufen sollen.)

Zusätzlich werden durch das Script alle bisherigen Ergebnisse zu einer neuen kompletten Ergebnistabelle zusammengefasst. (Dateiname lrbacktest_xxxx_tmp_end_out.csv, wenn -o <name> verwendet wird, dann <name>_xxxx_tmp_end_out.csv)

(Achtung: Diese Tabelle ist noch nicht sortiert! Bitte selber in einer Tabellenkalkulation sortieren!)

D.h. mit jedem weiteren abgeschlossenen Einzelbacktest entsteht wieder eine neue zusammengefasste Ergebnistabelle, die um die Ergebnisse des soeben abgeschlossenen Tests erweitert wurde.

Damit kann man mit einer unrealistisch hohen Zahl an Parametersets in den parallelen Backtest gehen, alle Stunde die neueste zusammengefasste Ergebnistabelle kontrollieren, und - wenn sich bei den Spitzenergebnissen nichts wesentliches mehr zu ändern scheint - den ganzen Backtest abbrechen.

Das Abbrechen ist unter Windows etwas mühsam, weil ein einfaches STRG-C nur einen der gerade laufenden Backtests abbricht, worauf das Script automatisch den nächsten Backtest startet.

Mit einem Tool wie procmon.exe aus der Sysinternals-Suite von (inzwischen) Microsoft kann man gezielt den Eltern-Python-Prozess abschießen, was die ganze kaskade in einem beendet.

Es geht auch mit Bordmitteln, wie ein Benutzer (danke!) herausgefunden hat:
In einer Eingabeaufforderung mit Admin-Rechten eingeben
`taskkill /F /IM "python.exe" /T`

Unter Linux hingegen reicht weiterhin ein einfaches STRG-C, weil Linux die parallelen Backtests unter der Haube anders startet als Windows.

Zwei wichtige Hinweise:

Damit die Beschleunigung durchs Cachen funktioniert, muss in der Configdatei 'lrbacktest07.ini' in der Sektion der jeweiligen Strategie der Eintrag `cache_enable = yes` stehen!

Da die Backtests für die Candledaten und das Zwischenspeichern einiges an Speicher benötigen, sollte man sich an einen sinnvollen Wert für die Zahl der parallelen Backtests herantasten.
Ein einzelner Backtest kann bei - je nach Testzeiträumen - durchschnittlich zwischen 0.3 GB und 4 GB benötigen. Je mehr Zeiträume oder je längere Zeiträume, desto mehr Speicher wird benötigt.

Außerdem skaliert - vermutlich aufgrund des hohen Speicherverbrauchs - die Geschwindigkeit nicht linear mit der Zahl der gleichzeitigen Backtests.
D.h. 2 parallele Backtests sind nicht ganz doppelt so schnell wie ein einzelner Backtest, vier sind nicht erst recht nicht viermal so schnell, etc. ...

Ein Beispiel:

```
python backtest.py -g 1000 -o erster_test
```

Es wird 1 Datei mit 1000 Parametersätzen erzeugt:

```
- saving 'erster_test_a_in.csv' ...
```

Starten von drei gleichzeitigen Backtests mit je 100 Parametersätzen:

```
python testmanager.py -r erster_test_a_in.csv -s 100 -t 3
```

Das Testmanager-Script zerlegt zuerst die Eingabedatei erster_test_a_in.csv in mehrere Teildaten mit je 100 Parametersets.
Als zweites startet das Script einen einzelnen Testlauf, der ausschließlich auf fehlende Candles prüft und diese ggfs. herunterlädt. Danach werden die ersten drei parallelen Backtests gestartet.

Der erste Backtest, also der, der mit der Datei '`erster_test_0000_a_in.csv`' gestartet wird, schreibt übrigens seinen Zwischenspeicher-Cache automatisch auf Platte, damit nachfolgende Backtests sofort mit vollem Tempo laufen.

Wenn man später weitere Backtests startet, existieren die gespeicherten Ergebnisse bereits und beschleunigen somit auch schon die ersten Backtests des neuen Testlaufs.

Nochmal der Hinweis - die vom Script erzeugten zusammengefassten Ergebnis-Tabellen muss man in einer Tabellenkalkulation nochmal nach 'rank_all' oder anderen Feldern sortieren lassen!

5. Die Ausgaben

Der Backtest gibt auf der Konsole pro Backtest einige zentrale Informationen pro Backtest aus - ein Beispiel:

```
Test 'a'  -  from 01-03-2018 to 01-06-2018

Run A-0000/idx:0000 - scanning price table from 3240 to 5447
```

```
Final balance now  972.41, ATH 1000.00, MaxDD  3.5%, profit factor 0.24,
ulcer index 2.08, kelly criterion -0.55
- win:    3 trades, avg $2.65, sum $7.96
- loss:  14 trades, avg $2.35, sum $-32.89
Backtest needed 648 secs

Run A-0001/idx:0001 - scanning price table from 3240 to 5447
Final balance now 1056.10, ATH 1107.72, MaxDD  6.0%, profit factor 2.16,
ulcer index 3.12, kelly criterion 0.21
- win:   15 trades, avg $10.99, sum $164.81
- loss:  24 trades, avg $3.18, sum $-76.27
Backtest needed 89 secs

Run A-0002/idx:0002 - scanning price table from 3240 to 5447
Final balance now 1080.27, ATH 1093.14, MaxDD  3.7%, profit factor
13.24, ulcer index 0.81, kelly criterion 0.49
- win:    8 trades, avg $13.82, sum $110.54
- loss:   7 trades, avg $1.19, sum $-8.35
Backtest needed 12 secs
```

[...]

Hier sieht man auch sehr schön, wie der Beschleunigungseffekt einsetzt, weitere Backtests brauchten in diesem Testlauf dann zwischen 10 und 20 Sekunden.

In der Protokoll-Log-Datei, die für jeden Testzeitraum und für jede Parameterkombination erzeugt wird, werden für jede Candle der Zeitpunkt und die durchgeführten Aktionen festgehalten, also die Aufrufe des Stoploss-Checks, ggfs der Strategie und die eventuell dabei entstehenden Aktionen wie Käufe und Verkäufe. Ebenso werden die Equity und der jeweilige Wert gekaufter Coins mitprotokolliert.

Die Protokoll-CSV-Datei, die für jeden Testzeitraum und für jede Parameterkombination erzeugt wird, enthält zu jeder Candle folgende Felder enthält:

- time: Datum und Uhrzeit als Unix-Timestamp
- date: Datum und Uhrzeit menschenlesbar
- equity: das Gesamtvermögen zum jeweiligen Zeitpunkt
- balance: der Anteil des Vermögens in USDT
- coins: die Zahl der Coins, die aktuell im Besitz sind
- range: -1 = bear, 0 = side, 1 = bull
- buyandhold: hätte man anfänglich mit der Startbalance BTC gekauft und würde diese einfach nur hodeln, dann ergäbe sich zu jedem Zeitpunkt der Wert in dieser Spalte

Mit den Diagrammfunktionen einer Tabellenkalkulation kann man die Spalten 'date' bis 'buyandhold' in ein aussagekräftiges und hilfreiches Diagramm verwandeln - ein Bild sagt mehr als 1000 Worte. ;-)

Für einen schnellen Überblick kann man auch das beiliegende Script show_bt.py verwenden. Es liest diese CSV-Datei aus und zeigt die Werte für equity, balance und buyandhold über den Testzeitraum grafisch an. Mit dem Lupensymbol kann man Teilzeiträume herausvergrößern und so gezielt interessante Zeitpunkte unter die Lupe nehmen.

Damit das Script funktioniert, müssen in der aktuellen Umgebung noch das Modul matplotlib und bei Linux ggfs noch über die Linux-Paketverwaltung das Programm 'Phython-tk' (kein Python-Modul!) nachinstalliert werden:

Windows: `pip install matplotlib`
Linux: `pip3 install matplotlib`
u. ggfs: Python-tk [aus der Paketverwaltung]

6. Fallstricke und Wissenswertes bei den Parametern

Es gibt ein paar Punkte, die man wissen sollte, um die Parameterbereiche sinnvoll wählen zu können, und um die Ergebnisse richtig interpretieren zu können.

a) welche Werte für xxx_min_r2?

Alle xxx_min_r2-Werte liegen aus mathematischen Gründen ausschliesslich zwischen 0.0 und 1.0. Je genauer die Kurswerte einer Gerade bzw einer Parabel nahe kommen, desto mehr nähert sich der errechnete min_r2-Wert der 1.0 an.
1.0 entspricht dabei einer perfekte Gerade/Parabel ohne auch nur den Hauch einer Abweichung. Im Gegenzug heißt das für reale Kursen, dass die Qualität eines echten Kursverlaufs immer unter 1.0 bleiben wird.

Wählt man folglich für xxx_min_r2 einen Wert von 1.0 oder höher, wird die errechnete Trendfunktion immer eine Genauigkeit darunter haben - Bot bzw Backtest könnten folglich nie kaufen!

Das ist eine elegante Methode, um im Backtest auszuprobieren, ob z.B. der Bereich Side zum Ergebnis positiv beiträgt oder nicht.
Einfach zu den Werten für side_trend_min_r2 zusätzlich noch 1.0 dazu, z.B. `side_trend_min_r2 = 0.85 0.90 0.95 1.0` dann wird auch mitgetestet, wie die Ergebnisse ohne Side aussehen.

b) Trendwert für Bull muss größer sein als der für Side

Bot und Backtest gehen davon aus, dass der Trendwert für den Bereich Bull höher ist als für den Bereich Side! Setzt man in den Parametern 'side_trend' auf einen höheren Wert als 'bull_trend', führt das dazu, dass der Bereich Side nie genutzt wird, sondern Bot oder Backtest nur zwischen Bull und Bear hin- und herschalten.

c) Was bedeutet die Telegram-Infomeldung 'Pairs Switch'?

Gehört zu den Parametern bull_difftrend_pair und side_difftrend_pair bei QuadReg.

Das ist ein Trick, um einen Coin loszuwerden, der eher mäßig performed, aber nicht so schlecht ist, dass er verkauft werden würde.

Wenn es einen Coin gibt, den man kaufen könnte, und dessen Trend mindestens um <difftrend_pair> besser ist, dann wird der Minderperformer verkauft und der bessere Coin dafür gekauft.

Wenn bei 'owned pairs' 'none' steht, dann besitzt du keinen Coin, bei dem sich das Tauschen gegen das 'Best of unowned pairs' lohnen würde.

Entweder weil deine Coins alle recht gut sind, und deswegen der Tausch zuwenig bringen würde.

Oder, weil deine Coins alle so schlecht sind, dass sie der Bot sowieso verkaufen möchte, und dann keiner mehr zum Tauschen übrig bleibt.

d) Log- und CSV-Files / testmanager.py / back_single.py

Wenn man den Backtest direkt aufruft ('`python backtest.py`' bzw '`python3 backtest.py`'), dann werden für jeden Einzeltest ein Log- und ein CSV-File erzeugt.

Mit der Option '-nof' ("no output files") kann man verhindern, dass diese detaillierten Files erzeugt werden.

Das kann sinnvoll sein, wenn man große Testserien fährt und erst mal an den Gesamtergebnissen interessiert ist und nicht an Details.

Das TestManager-Script ruft deshalb intern die Backtests mit genau dieser Option auf. So kann man Tausende von Kombinationen überprüfen, ohne nebenbei gleich mehrere Tausend Dateien zu erzeugen, was bei sehr schnellen CPUs durch die IO-Last den Backtest merkbar bremsen kann.

Für einzelne interessante Kombination kann man diese Detail-Dateien aber leicht nachträglich aus der Ergebnistabelle erstellen lassen:

Windows: `python back_single.py -in <tabelle.csv> -idx <indexnummer>`
Linux: `python3 back_single.py -in <tabelle.csv> -idx <indexnummer>`

Das Script back_single.py liest die Tabelle aus, findet die zu <indexnummer> gehörenden Parameter und erzeugt aus der momentanen lrbacktest07.ini eine angepasste lrbacktest_<indexnummer>.ini, mit der anschliessend ein einzelner Backtest über die definierten Zeiträume gestartet wird.

Damit hat man zwei Fliegen mit einer Klappe:

- aus der neuerzeugten Ini kann man die mundgerecht formatierten Zeilen für die lrbot07.ini einfach rauskopieren - kein Abschreiben mehr
- man bekommt die detaillierten Log- und CSV-Dateien für genau diese eine Indexnummer

e) Welche Coins bringen's und welche nicht?

Die Kursverläufe mancher Coins eignen sich besonders für die Strategie QuadReg, während Kurse mit zackigem, unruhigem Verlauf für Regressionsmethoden ungeeignet sind.

Deshalb gibt das Detail-Log, das beim Aufruf von backtest.py für jede Parameterkombination und jeden Testzeitraum erstellt wird, am Ende der Datei die Ergebnisse pro Coin aus - ein Ausschnitt:

```
Coin Results:

- DOGE/BTC  : total $   359, win $   359, loss  $     0
- XRP/BTC   : total $    75, win $    75, loss  $     0
- IOST/BTC  : total $    72, win $    72, loss  $     0
- XVG/BTC   : total $    51, win $    51, loss  $     0
[...]
- WAVES/BTC : total $   -16, win $     0, loss  $   -16
- DATA/BTC  : total $   -18, win $     0, loss  $   -18
- XEM/BTC   : total $   -20, win $     0, loss  $   -20
- ONT/BTC   : total $   -26, win $     0, loss  $   -26
- BTC/USDT  : total $   -34, win $     0, loss  $   -34
```

Hier erkennt man, dass DOGE zu einem Großteil des Gewinns beigetragen hat, während sich die Verluste etwas gleichmäßiger auf mehrere Coins aufteilen.

Wenn man die Ergebnisse der Testzeiträume der gleichen (!) Parameterkombination vergleicht, dann kann man möglicherweise Coins identifizieren, die in einem oder mehreren Zeiträumen starke Verluste erzeugen.

Wenn ein Coin in einem oder mehreren Zeiträumen negativ ausfällt und auch in den restlichen Zeiträumen keinen merklichen, positiven Beitrag leistet, dann ist das ein starkes Indiz dafür, dass sein Kursverlauf nicht für die Methoden des LRBot geeignet ist.

So bin ich darauf gekommen, dass REP und AGI sehr ruppige, ungünstige Kursverläufe aufweisen. Ihr Ausschluss aus dem Backtest über die Blacklist hat die Ergebnisse in zwei meiner drei Zeiträumen merklich verbessert und deren Drawdowns verringert. Der dritte Zeitraum (Bull) blieb unverändert.

In Konsequenz wurden REP und AGI auch im Live-Bot auf die Blacklist gesetzt.

f) Warum erhalte ich weniger Ergebnisse als meine eingestellte Testzahl?

Wenn ein Backtest einen Drawdown von 50% oder schlechter erreicht, wird dieser Test abgebrochen. Das Ergebnis dieses Tests wird konsequenterweise auch nicht in die Ergebnistabellen aufgenommen.

7. Grundeinstellungen für den Backtest

Nachfolgend die wichtigsten Grundeinstellungen für den Backtest:

Sektion [common]

pairslist
Hier werden die Coins angeben, die zum Test verwendet werden sollen

rangelist
Hier werden die Coins angegeben, aus deren Kursverläufen der Marktzustand (Bull, Side, Bear) abgeleitet werden soll.

Sektion [test]

Hier werden die Testzeiträume festgelegt. a_startdate und a_enddate legen den ersten Testzeitraum fest, b_startdate und b_enddate den zweiten und so weiter.
Maximal sind fünf Testzeiträume möglich, also 'a_' bis 'e_'. Nicht benötigte Zeiträume aus der Config löschen!

strat_firststratcheck
Viele größere Marktteilnehmer agieren auf Basis von Tageskursen, weswegen es um 0:00h UTC oft leicht stärkere Kursbewegungen gibt. Deswegen startet der Backtest auch immer mit 0:00h UTC. Wer mit anderen Startzeiten experimentieren mchte, kann hiermit eine andere Startstunde im UTC gerechnet einstellen.

startbalance_usdt
Hier gibt man an, mit welchem Anfangskapital der Backtest starten soll.

fees u. spread
Die Werte sind als Prozentangaben zu verstehen, d.h. fees = 0.10 entspricht 0.10% (Binance). Der Spread kann sehr variieren, umsatzschwache Coins haben einen höheren, Coins mit einem dicken Orderbook einen niedrigeren. Der vorab eingetragene Wert von 0.25 (0.25%) ist nur geraten - Hinweise auf Erfahrungswerte sind willkommen!

volume_min
Coins mit geringem Volumen sind sehr leicht manipulierbar, weil jede größere Order eine deutliche Preisveränderung zur Folge hat, was von Pump- und Dump-Gruppen auch weidlich ausgenutzt wird.
Andererseits entstehen durch größere Preisschwankungen auch bessere Chancen.
Insofern kann man hier eine Schwelle angeben, welches Mindestvolumen (in BTC pro Tag angegeben) ein Coin mindestens aufweisen muss, um vom Bot zum Traden herangezogen zu werden. Coins unter dieser Schwelle werden nicht gekauft.
Ich verwende normalerweise 100.0.

price_min

Coins mit Preisen unter 100 Satoshi haben einen zunehmend "eckigeren" Kurs-Verlauf, weil die kleinste mögliche Kursveränderung bei z.B. 50 Satoshi bereits 2% beträgt. Der Kurs springt somit gerne zwischen zwei Werten hin- und her, weil es keine Zwischenwerte gibt. Das erschwert dem Bot die Einschätzung, weswegen ich normalerweise 0.00000100 als Schwellwert verwende.

max_combinations
Da man ab der V0.7 bei allen Parametern aus der Strategie-Sektion mehrere Werte angeben kann, ergeben sich sehr viele Kombinationsmöglichkeiten. Da es sehr schnell unmöglich wird, tatsächlich alle Kombinationen durchzurechnen, kann man hiermit begrenzen, wieviele Kombinationen tatsächlich durchgerechnet werden sollen.

Ein Rechenbeispiel: In [quadreg] gibt es 32 Parameter, denen man mehrere Werte vorgeben kann. Wenn man pro Parameter nur drei verschiedene Werte angibt, ergeben sich bereits 3 ^ 32 = 1,8×10^15 verschiedene Kombinationen.
Wenn das Durchrechnen eines Zeitraums mit einer Parameterkombination z.B. 10 Sekunden dauert, kann man über Nacht ca. 8h / 10 sec = knapp 3000 Kombinationen duchrechnen.

filter_keep_best
Um die Rechenzeit zu senken, kann man nach jedem Testzeitraum die schlechtesten xx% der Ergebnisse verwerfen und mit der Restmenge in den nächsten Testzeitraum starten.

Beispiel: Sei max_combinations = 100 und es sollen drei Testzeiträume berechnet werden. Ohne filter_keep_best müsste der Backtest 100 + 100 + 100 = 300 Durchläufe rechnen. Mit den voreingestellten filter_keep_best = 81 wirft der Backtest jedesmal die schlechtesten 19% der Ergebnisse weg, es bleiben somit immer 81% der vorherigen Menge übrig.
Somit sind "nur" 81 + 65 + 52 = 198 Durchläufe zu rechnen, also immerhin ein Drittel weniger.

Stellt man filter_keep_best = 70 ein, werden von den ursprünglichen 300 noch 70 + 49 + 34 = 153 Ergebnisse berechnet.

zip_logcsv
Mit 'zip_logcsv = yes' werden die Einzelfiles (CSV u. Log) nach jedem Test-Zeitraum in ein Zip wegkomprimiert, um das Verzeichnis nicht mit Tausenden von Protokolldateien zu überfluten.

8. Gemeinsame Felder aller Strategien

Bis auf "candle_length" kann man bei jedem Parameter aus einer StrategieSektion (z.B. [quadreg]) mehrere Werte angeben (die durch ein Leerzeichen getrennt sein müssen).

cache_enable = yes
Nur für Inhaber einer Lizenz mit Pro-Backtest! Schaltet das Caching der Zwischenergebnisse ein und erhöht die Testgeschwindigkeit um etwa den Faktor 20 - um den Preis eines höheren Speicherverbrauchs, der durchaus zwischen 1 und 4 GB erreichen kann.

candle_length = 1h
Legt fest, auf welcher Intervalle-Länge der Bot arbeiten soll. Nach meinen Tests bringen kürzere Intervalle als 1h keine Vorteile.
Hinweis: Die eingestellte Candlelänge sollte auf der Exchange, auf der der LRBot später traden soll, natürlich auch verfügbar sein!

ACHTUNG: Bei kürzeren Intervallen als 1h bekommt man schon bei moderaten Test-Zeiträumen Fehlermeldungen von der vom Backtest verwendeten Datenquelle cryptocompare.com!
(Vernünftige Zeiträume mit Candles unter einer Stunde gibt es im Internet nicht kostenlos.)

check_every4buy = 12h 18h 24h
check_every4sell = 12h 18h 24h
Die Strategie wird im Normalfall nicht nach jeder neuen Candle aufgerufen, sondern seltener, um weniger aufs Rauschen und kleine Korrektur hereinzufallen. Nach meiner bisherigen Erfahrung ergeben hier die acht- bis zwölf-fachen Werte von "candle_length" die besten Ergebnisse. Bei kleineren Werten sinken die Ergebnisse merklich.
Über diese beiden Parameter stellt man getrennt ein, alle wieviel Candles nach neuen Kauf- oder neuen Verkaufsgelegenheiten geprüft werden soll.

reinvest = 0
Achtung: Zum Suchen oder Tunen einer Strategie sollte reinvest auf 0 bleiben!
Reinvests verzerren die Ergebnisse, weil Täler tiefer und Höhen höher werden. Erst, wenn die Strategie fertig ist, kann man mit einem Wert zwischen 20 und 50 nicht noch ein paar Prozente herauskitzeln kann.
Über reinvest kann man steuern, ob der Bot nach Gewinnen seinen zukünftigen Kaufbetrag pro Coin vergrößert oder nicht. Bei reinvest = 0 bleibt der Kaufbetrag immer gleich, bei reinvest = 100 würde der Bot bei verdoppelter Balance auch doppelt so viel pro Kauf verwenden - allerdings wird mit vollständiger Reinvestion der Gewinne jeder Bot instabiler und anfälliger in schlechten Kurszeiten. Deshalb wird der Wert auf den Bereich zwischen 0 und 50 begrenzt.

market_bull_trend = 0.35
market_side_trend = 0.00 0.05 0.10
market_bear_trend = -0.10 -0.05 -0.00
Der Durchschnittstrend der Rangecoins wird errechnet und anhand dieser Schwellwerte in Bull, Side, Bear oder Badbear einsortiert.
In Badbear wird grundsätzlich immer sofort alles verkauft, weil das Risiko von größeren Kursverlusten sehr hoch ist.

len_min = 24 30 36 42 48
len_max = 92 144 200 300 400 500
len_short_min = 6 8
len_short_max = 6 8 10 15
Diese Längen dienen zur Ermittlung des Durchschnittstrends der Rangecoins. Zuerst wird pro Rangecoin die beste lange Regressionsgerade in den Grenzen von len_min und len_max gesucht. Da die langen Regressionsgeraden nur sehr träge auf Änderungen der Marktverhältnisse reagieren, wird noch eine kurze Regressionsgerade in den Grenzen von len_short_min und len_short_max berechnet.
Der Durchschnitt der langen Geraden und der Durchschnitt ihrer Qualität (R2) wird dann gegen den Durchschnitt der kurzen Geraden und deren R2 abgewogen.

QuadReg verwendet diese Grenzen auch zur Berechnung der Regressionsparabeln der einzelnen Coins.

Stop Loss (SL)

Ab dem Kauf eines Coins wird der Kurs bei jeder neuen Candle auf einen neuen Höchstwert (ATH) geprüft. Der höchste Wert seit dem Kauf wird gespeichert und ggfs aktualisiert.

Fällt der Kurs wieder, wird in Abhängigkeit von den Stoploss-Parameter ggfs ein Verkauf bei Überschreiten einer vorgegebenen Verlustschwelle ausgelöst.

Achtung: Da der Bot die Kurse nicht in Realtime prüft, sondern nur nach jeder Candle, bietet dieser Stoploss nur einen gewissen Schutz. Der Kurs kann im Extremfall in der Zwischenzeit zwischen zwei Candeln deutlich tiefer fallen!
Andererseits löst auch nicht jedes kurze Zucken des Kurses gleich den Stoploss aus.

Um unruhigen Coins mehr Luft zu lassen und oder bei ruhig verlaufenden Coins schneller zu reagieren, kann die Verlustschwelle nicht nur in Prozent, sondern auch in "ATR" angegeben werden. ("ATR" stimmt nicht ganz, da der Bot nur auf Close-Werten operiert.)
ATR misst die durchschnittliche Kursveränderung zwischen den Candles. Damit passt sich der SL-Abstand dem Kursverlauf an. Bei größeren Schwankungen wächst der Abstand ein bisschen, bei glattem Verlauf rutscht der SL näher an den Kurs hin.

sl_usage = yes no
Damit schaltet man die Nutzung von Stop Loss ein oder aus. Auch die mit der V0.7.7a hinzugefügte Funktion sl_takeprofit_side wird damit ein- und ausgeschaltet!
Wer Stop Loss nicht, aber sl_takeprofit_side schon verwenden möchte, muss folglich hier einschalten, dann aber die eigentlichen Stop Loss-Parameter so wählen, dass der eigentliche Stop Loss nicht zum Tragen kommen kann, d.h. sl_loss = -99.

WICHTIG: Auch, wenn ein ausgeschalteter Stop Loss-Block keinen Einfluss mehr auf die Berechnung hat, zählen Varianten seiner Parameter mit zu den zu berechnenden Parameterkombinationen.
Deshalb nicht nur ausschalten, sondern auch bei allen Stop Loss-Parametern nur noch einen einzigen Wert stehen haben!

sl_useatr = yes no
Berechnung des SL-Abstandes in ATR oder in Prozent

sl_candlecount = 6 8 10 12
Falls sl_useatr = yes, gibt man hier an, wieviele Candles zur Berechnung des ATR herangezogen werden sollen

sl_loss = -2 -4 -6 -8
Größe des SL-Abstandes, entweder in Prozent oder in ATR-Einheiten

sl_loss_bull = 1.0 1.25 1.50 1.75 2.00
Tuning: Wenn der Markt bullisch ist, wird der SL-Abstand mit diesem Faktor multipliziert, der Bot wird sozusagen etwas großzügiger, was die Ergebnisse verbessert.

sl_takeprofit_side = 3.0 4.0 5.0
In den Bereichen Side und Bear ist die Gefahr groß, dass Kurse schnell wieder aus der Gewinnzone rutschen. Deshalb gibt es die Möglichkeit, über den Parameter sl_takeprofit_side beim Erreichen eines bestimmten Profitwertes von z.B. +4% direkt zu verkaufen.
Anhand umfangreicher Backtests hat sich gezeigt, dass man in Bullenmärkten Trades besser laufen lässt, Verkäufe beim Erreichen fester Profitschwellen senken das Gesamtergebnis!
Deshalb wird sl_takeprofit_side nur in den Bereichen Side oder schlechter geprüft. Bei der Strategie MA3 muss zusätzlich noch der Coin selbst im Bereich Side (oder schlechter) gekauft worden sein.

Achtung: Wie der SL-Check findet diese Prüfung ebenfalls nur für jede neue Candle statt - wenn der Kurs zwischen zwei Candles in die Höhe geht und wieder fällt, kann kein Verkauf ausgelöst werden.

Wie weiter oben schon ausgeführt, rechnet sich der Mut zum Risiko! Wer hier risikoscheu schnelle Verkäufe aufgrund kleiner SL-Abstände oder niedriger sl_takeprofit_side Limits einstellt, zahlt in Summe für die höhere Sicherheit massiv drauf!

9. Die wichtigsten Parameter der Strategie QuadReg

Einführung:

QuadReg basiert auf der Idee, den Kursverlauf mit Trendparabeln anzunähern.

Je zuverlässiger sich der Kurs eines Coins an eine Parabel hält, und je mehr diese Parabel ansteigt, desto eher wird der jeweilige Coin gekauft.

Wenn diese Parabel nicht mehr steigt, sondern waagerecht verläuft oder sogar zu fallen beginnt, wird der jeweilige Coin wieder verkauft.

Diese Methode funktioniert deswegen, weil sie das Kursrauschen ausmittelt und sich auf die großen Kursbewegungen konzentriert. Kleine Wackler werden ausgesessen.
Deswegen empfiehlt es sich auch, seine Verkaufsparameter nicht "ängstlich" auszuwählen, sondern eher großzügig, denn durchschnittlich verliert man mehr beim Warten auf einen neuen Einstieg, als man beim frühen Verkauf rettet!

Grundlagen zur Regressionsmethode und zu Regressionsgeraden und -parabeln finden sich im Dokument **regression.pdf**.

Die Strategie QuadReg baut auf dem Verläufer LinReg auf, der die Kurse mit Geraden annähert. Geraden sind jedoch ziemlich eingeschränkt, weil die Kurse gerne steigen oder fallen oder seitwärts laufen, mit anderen Worten "oft um die Kurve gehen".

Im Gegensatz zu Geraden sind Parabeln viel variabler - eng, breit, steil, flach, wie eine Gerade, nach oben geöffnet wie ein großes U oder nach unten geöffnet wie ein auf dem Kopf stehendes U.

Ein wichtiger Unterschied zwischen Parabeln und Gerade ist die fehlende "Eindeutigkeit" von Parabeln.

Eine Gerade nimmt einen gleichbleibenden Trend an. Bei einer guten Steigung und einem ordentlichen R2 kann man laut Theorie einfach kaufen.

Bei der Parabel ist er aufgrund der "Kurvigkeit" schwieriger. Die Steigung, die für die neueste Candle zurückgemeldet wird, gilt nur für diese Candle. Davor war sie normalerweise anders anders, und für zukünftige Candles wird sie wohl auch anders sein, da die Parabel gebogen ist.

Wenn eine Parabel also einen positiven Trend mit gutem R2 zurückliefert, muss man beispielsweise aufpassen - stammt der positive Trend von der rechten Hälfte eines aufrechten Us oder von der linken Hälfte eines kopfstehenden Us?

Im ersten Fall vermutet die Theorie, dass der Kurs immer schneller steigen wird. Im zweiten Fall ist anzunehmen, dass das Wachstum abbaut und bald zum Stillstand kommt.

In allen bisherigen Tests konnten keine signifikanten Erfolge für das vorsichtige Zulassen von Käufen auf der linken Hälfte eines kopfstehenden Us gefunden werden.

Deshalb wird in QuadReg nur auf dem rechten Ast einer aufrechten Parabel gekauft.

Die einzelnen Felder der Sektion [quadreg]

len_min = 30
Soviele Candles lang soll eine Trendgerade mindestens sein, bevor der Bot den Coin in Betracht zieht. Kurze Werte lassen den Bot schneller auf einen Coin aufspringen, die Gefahr ist allerdings auch größer, dass sich ein Kursanstieg als Bullenfalle entpuppt.
In volatilen Zeiten habe ich gute Erfahrungen mit dem Bereich von 24 bis 48 gemacht; im zweiten Halbjahr 2018 mit seinen geringen und kurzen Anstiegen muss man mit kürzeren Werten zwischen 10 und 20 experimentieren.

len_max = 400
Maximale Länge einer Trendgerade, die der Bot noch berücksichtigt. Je größer die maximale Länge, desto besser werden zwischenzeitliche Korrekturen ignoriert und ausgesessen. Desto höher aber auch der Rechenaufwand und die Testzeit.
Gute Erfahrungen habe ich mit dem Bereich von 300 bis 500 gemacht.

len_short_min = 7
len_short_max = 12
Da die Cryptomärkte sehr viel volatiler als die Aktienmärkte sind, auf denen die Methode des Regressionsgeraden-Tradings entwickelt wurde, braucht man einen Sicherheitsgurt, Kursverschlechterungen schneller zu erkennen.
Hiermit wird ein Schnelltest über die letzten x Candles durchgeführt. Wenn diese kurze Trendfunktion ihre Candles genauer annähert als die lange Funktion ihre vielen Candles, dann übersteuert der Trendwert dieser kurzen Parabel die normale, längere Trendparabel.
Gute Erfahrungen habe ich mit dem Bereich von 7 bis 12 gemacht.

trend_short_min_r2 = 0.50
Wenn der Kurs bei der kurzen Parabel nicht nur schlechter steigt, sondern sogar fällt, dann tritt die zweite Stufe des Sicherheitsgurtes in Kraft.
Die Signifikanz der kurzer Parabel muss dann nicht mehr besser sein als die der langen Parabel, sondern muss nur noch mindestens den Wert von "trend_short_min_r2" erreichen, um die lange Parabel zu übersteuern.
Gute Erfahrungen habe ich mit dem Bereich von 0.30 bis 0.50 gemacht.

trend_short_drop_limit = -0.30
Dieser Wert ist schwierig zu erklären. Siehe die Erklärung unter bull_trend und side_trend.
Lasst ihn am besten im Bereich vom -0.20 bis -0.30

Ranges 'bull' und 'side'

bull_trend = 0.20
Wenn der Parabeltrend des Coins den Wert von bull_trend übersteigt, schaltet der Bot/Backtest auf die im Bereich Bull eingetragenen Parameter.
Der Wert von bull_trend sollte über dem von side_trend liegen, ansonsten kann der Bereich Side nie aktiviert werden.

side_trend = 0.20
Wenn der Parabeltrend des Coins den Wert von side_trend übersteigt, aber noch unter dem von bull_trend liegt, schaltet der Bot/Backtest auf die im Bereich Side eingetragenen Parameter.
Der Wert von bull_side sollte unter dem von bull_trend liegen, ansonsten kann der Bereich Side nie aktiviert werden.
Liegt der Parabeltrend des Coins unter dem Wert von side_trend, sind Käufe verboten, und Coins im Besitz werden verkauft, falls der Trend des Coins unter den Wert von bear_trend_sell fällt.

Eine Besonderheit gibt es zur Bereichsbestimmung noch - der kurzfristige Trend muss zusätzlich über dem Wert von trend_short_drop_limit liegen.
Fällt der kurze Durchschnittstrend aller Coins unter den Wert von trend_short_drop_limit, wird sofort auf Bear geschaltet.
Das ist ein Art von Dropschutz, der etwas schneller reagiert als der langfristige Durchschnittstrend.

xxxx_maxcoins = 10
Über xxxx_maxcoins wird der Kaufbetrag pro Coin errechnet
kaufbetrag = startbalance_usdt / maxcoins.
Weniger maxcoins erhöhen die Chance in Bullenmärkten, steigern aber die Anfälligkeit, wenn der Markt dreht oder fällt.
Ein guter Mittelweg ist ein Wert zwischen 10 und 20.

xxxx_trend_sell = 0.05
Wenn die Trendgerade diesen Wert unterschreitet, wird der Coin verkauft. Um bei drehenden Märkten nicht zuviel wieder abzugeben, empfehlen sich Werte zwischen 0.05 und 0.10, in Bärenmärkten ggfs noch ein bisschen höher. In Märkten mit geringen Kursschwankungen wie im 2. Halbjahr 2018 sollte man mit niedrigeren Werten experimentieren.

xxxx_trend_buy = 0.40
Wenn die Trendparabel diesen Wert überschreitet, wird der Coin zum Kaufen in Betracht gezogen. Gute Erfahrungen habe ich im Bereich 'bull' mit Werten von 0.35 bis 0.50 gemacht, in 'side' mit 0.50 bis 0.70.

xxxx_trend_min_r2 = 0.75
Dieser Wert kontrolliert, wie zuverlässig und berechenbar sich ein Coin in den letzten Candles entwickelt haben muss. Je wilder ein Coin umherspringt, desto niedriger sein Wert hierbei und desto größer das Risiko beim Kaufen und Verkaufen.

Gute Erfahrungen habe ich in 'bull' mit Werten von 0.70 bis 0.80 gemacht, in 'side' mit 0.80 bis 0.95.

xxxx_difftrend_btc = 0.20
Hiermit lässt sich eine Optimierung zu- oder abschalten. Wenn der BTC stärker steigt als der gekaufte Coin, ist es unter Umständen sinnvoll, vom Coin auf den BTC umzusatteln.
Nach meinen Erfahrungen ist es nicht ratsam, zu leichtfertig umzusatteln, sondern nur bei einem merklich besseres BTC-Steigen!
Gute Erfahrungen habe ich mit dem Bereich um 0.20 bis 0.30 gemacht. Ein Wert von 9 schaltet diese Optimierung ab.

xxxx_difftrend_pair = 0.60
Hiermit lässt sich eine zweite Optimierung zu- oder abschalten. Es kann durchaus vorkommen, dass ein gekaufter Coin einige Zeit lang knapp über der Verkaufsschwelle dahindümpelt und dadurch den Kauf deutlich besser performenderer Coins blockiert.
Deshalb kann einmal pro Strategiedurchlauf das schlechteste eigentlich noch zu behaltende Paar abgestoßen werden, wenn der Unterschied zum besten kaufbaren Coin mindestens <xxxx_difftrend_pair> beträgt.
Nach meinen Erfahrungen rentiert sich der Tausch nur, wenn der Unterschied sehr groß ist - gute Erfahrungen liegen bei Werten von '0.60' bis '0.90' vor. Ein Wert von 9 schaltet diese Optimierung ab.

Zu den Stopp Loss-Parametern
Die Ergebnisse der Backtests legen nahe, dass die Nutzung von Stop Loss bei QuadReg kontraproduktiv ist. Man verliert mehr an Gewinnen, als man vor Verlusten rettet.
Das scheint auch für sl_takeprofit_side zu gelten.
Die Empfehlung ist somit für QuadReg, den SL entweder komplett auszuschalten, oder nur extreme Risiken durch sehr hohe SL-Werte (z.B. sl_loss = -20) abzusichern.

10. Die Strategie MA3

Einführung:

Im Gegensatz zu QuadReg sucht MA3 nicht nach den großen langen Trends, sondern Trends mittleren und kürzerer Dauer. Daher ergeben sich hier mehr Trades kleineren Umfangs.

MA3 kombiniert trendfolgende Anteile mit Antitrend-Anteilen.

Exkurs Antitrend = Mean Reversion

Mean Reversion geht davon aus, dass das zu tradende Objekt einen "fairen" Wert besitzt. Sinkt der momentane Preis, wird angenommen, dass er mittelfristig mit einiger Wahrscheinlichkeit wieder zum "wahren" Wert zurückkehren wird.
Analog erwartet man, dass, wenn der Preis um einiges gestiegen ist, er wieder zum "wahren" Wert zurückgehen wird.
Man erwartet also immer eine Rückkehr zum Mittelwert ("mean").
Ein bekannter Vertreter dieses Stils sind die Bollinger Bänder.

MA3 teilt die Marktsituation ebenso wie QuadReg in die vier Ranges Bull, Side, Bear und Badbear ein.

Zusätzlich wird jeder Coin individuell in Moon, Bull, Side und Bear eingeordnet.

Zur Entscheidung, ob ein Coin gekauft oder verkauft werden soll, werden mehrere Kriterien benutzt:

- kreuzende MAs aus einem schnellen und einem langsamen MA
- Erfüllen einer RSI-Bedingung
- Erfüllen eines bestimmten Wertes in den Bollingerbändern

Moon:
Der Kurs steigt massiv, Anhand der kreuzenden Durchschnitte wird entschieden, ob gekauft wird. Solange der Coin in Moon bleibt, wird er nicht verkauft.

Bull:
Der Kurs steigt merklich. Anhand zweier kreuzender Durchschnitte und einer zusätzlichen RSI-Bedingung wird entschieden, ob gekauft oder verkauft wird.

Side:
Die Regressionsgerade des Coins verläuft halbwegs waagerecht, d.h. der Coin schwankt mehr oder weniger in einem Preisbereich. Wenn sich der Coin ein gewisses Mindestmaß unterhalb der Geraden befindet, und die Bedingungen von RSI und BB erfüllt sind, wird gekauft.
Wenn die Verkaufsbedingungen von RSI und BB erfüllt sind, wird verkauft.

Bear:
Hier wird direkt verkauft.

Dieser individuelle Bereich wird mit dem allgemeinen Marktbereich "gefiltert".

Damit ein Coin Moon oder Bull erreichen kann, muss der allgemeine Markt-Bereich Bull oder zumindest Side sein.

Wenn der allgemeine Marktbereich Bear ist, wird analog zu QuadReg sicherheitshalber verkauft.

In allen anderen Fälle wird Antitrend-Trading gemacht, wie oben zu Side beschrieben.

Exkurs kreuzende Durchschnitte

Man verwendet zwei Durchschnitte mit unterschiedlicher Periodenlänge, einen schnelleren (kürzere Periodenlänge) und einen langsameren (höhere Periodenlänge). Je nachdem, ob der schnellere Durchschnitt den langsameren von unten nach oben schneidet, oder von oben nach unten, kann man kaufen oder verkaufen.

Ein Bild sagt mehr als 1000 Worte:

(https://www.cashbackforex.com/Portals/0/images/School/Charts/50_200_MACross_EURUSDH4_2.png)

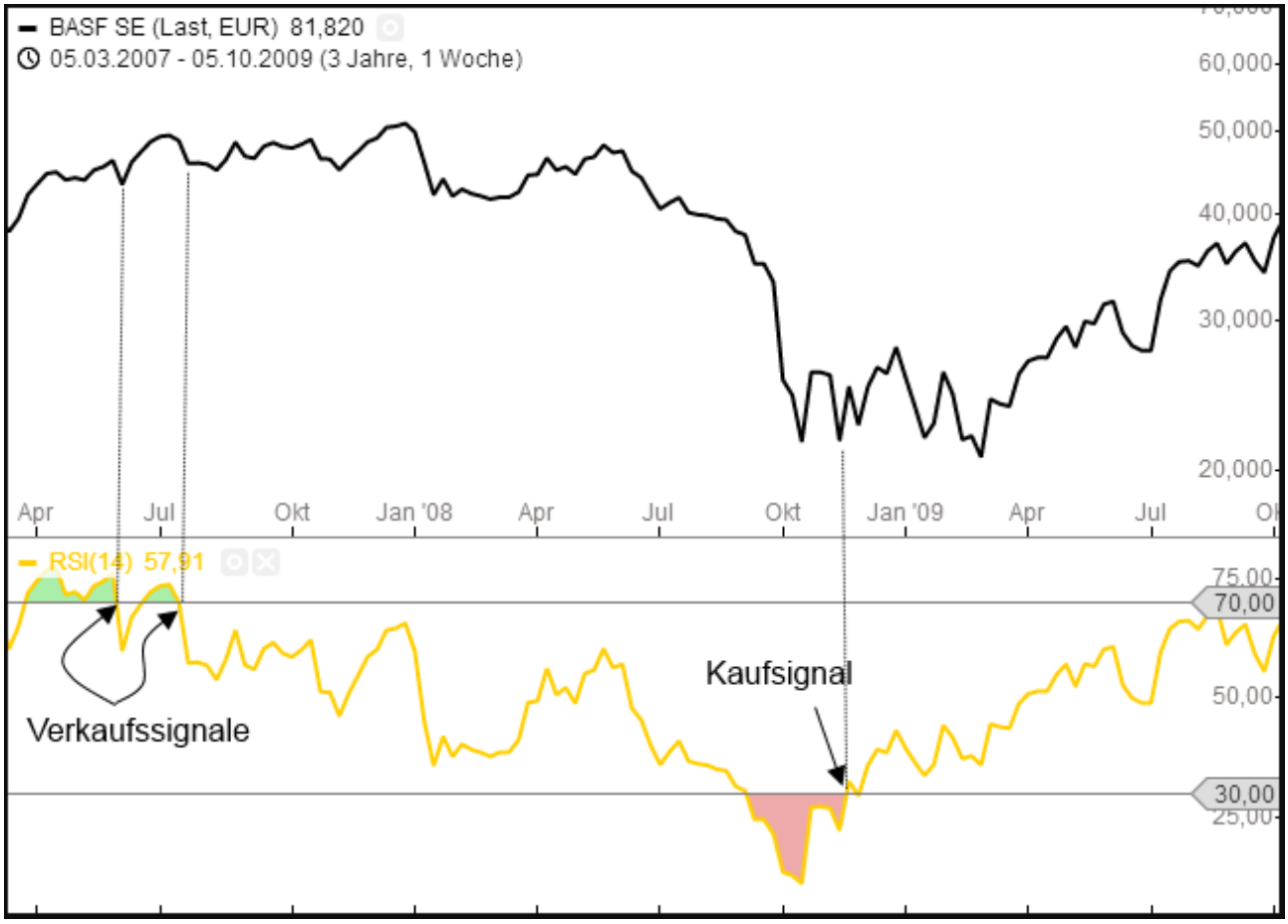
Exkurs RSI-Bedingung

Der RSI gibt - ganz grob ausgedrückt - an, wie stark oder schwach sich gerade der Kurs verändert. Steigt der Kurs stark, liegt der RSI deutlich über dem Wert 50, fällt der Kurs stark, liegt der Wert deutlich über 50.

Beim Antitrend-Trading wird der RSI gerne verwendet: Bei niedrigem RSI kauft man, weil man annimmt, dass der Kurs nur kurz fallen kann und im Kürze umdrehen muss, um zum Mittelwert zurückzukehren.
Umgekehrt beim Verkaufen: Weil man annimmt, dass die steigenden Phasen nur kurz sein können und der Kurs zum Mittelwert zurück möchte, verkauft man, sobald der RSI hoch ist.

Aber auch beim Traden mit dem Trend kann man den RSI verwenden, um den Einstieg und/oder den Ausstieg etwas zu pimpen.

Auch zum RSI ein Bild:



(https://img.godmode-trader.de/charts/104/2014/05/Abb5.png)

Exkurs BB-Bedingung

Nach der Mean-Reversion-Vermutung geht man davon aus, das der Kurs zu seinem Mittelwert tendiert. Anhand der Kursschwankungen um diesen Mittelwert wird ein "Schlauch" um den Mittelwert errechnet, in dem sich der Kurs mit hoher Wahrscheinlichkeit aufhalten sollte.
Wandert der Kurs in Richtung der Grenzen dieses Kanals, dann steigt die Wahrscheinlichkeit, dass der Kurs bald wieder in die wahrscheinlicheren Bereiche des Kanals zurückdreht.

In der Wikipedia ist das ausführlich erläutert:
https://de.wikipedia.org/wiki/Bollinger-B%C3%A4nder

Wartet man also ab, bis der Kurs sogar unter dem untere Bollinger Band liegt, ist die Chance recht groß, dass der Coin bald wieder steigt, weil sich der Kurs durchschnittlich nur zu knapp 4% unterhalb des unteren Bandes aufhält.
Allerdings kommt der Kurs aus den gleichen Gründen nur sehr selten so tief, d.h. die Kaufgelegenheiten sind dann zwar erfolgversprechend, aber selten.
Analoges gilt für den Verkauf über dem oberen Band.

Die einzelnen Felder der Sektion [ma3]

MA3 teilt viele Felder mit der Strategie QuadReg. Deshalb sind hier nur die MA3-spezifischen Felder aufgeführt.

EMA
Der EMA dient als langsamer Durchschnitt.

MA3
Der MA3 ist ein gewichteter Durchschnitt ähnlich dem EMA, nur deutlich reaktionsschneller und etwas glatter (smoother). Er funktioniert ähnlich wie der TEMA (Tripple-EMA), soll laut Literatur aber besser sein.

Zur Konstruktion des MA3 werden zwei Werte benötigt, ein höherer und ein niedriger. Der höhere Wert darf ähnlich dem Wert des EMA sein, *der niedrige muss konstruktionsbedingt zwischen einem Viertel und der Hälfte des höheren Wertes liegen*.
Auch wenn der größere der beiden MA3-Werte der EMA-Periodenlänge ähnlich gewählt wird, ist der MA3 trotzdem deutlich schneller als der EMA.

Wer sich das ganze visualisieren will, kann auf Binance in der Tradeansicht eines Coins auf TradingView schalten und sich einen TEMA und einen EMA gleicher Periodenlänge anzeigen lassen.

RSI
Eigentlich verwendet man RSI-Werte zwischen 0 und 100; ich habe bei der Berechnung den Faktor 100 übersehen - sorry - deswegen müssen die Angaben einstweilen zwischen 0.0 und 1.0 liegen. (Das wird vermutlich in einer späteren Version behoben.)

buyshare
Dieser Wert gibt im Zusammenspiel mit startbalance_usdt an, für welchen Betrag der Bot einen Coin kaufen soll.
Beispiel: startbalance_usdt = 1000, buyshare = 0.10, dann wird der Bot pro Kauf 0.10 * $100 = $100 verwenden.

<range>_allcoins = xx
<range>_maxcoins = yy
Da MA3 nicht wie QuadReg zur Risikobegrenzung auf einen guten MinR2-Wert zurückgreifen kann, kann es leichter durch kurze Anstiege mit sofort anschließenden Abfällen auf dem falschen Fuß erwischt werden: Die Kurse fangen an zu steigen, MA3 kauft sich ein, die Kurse fallen wieder, MA3 verkauft mit Verlusten wieder.
Deshalb kann man her festlegen, wieviel Coins die Strategie pro Strategiecheck maximal auf einmal (<range>_maxcoins) nachkaufen darf und wieviel Coins die Strategie insgesamt maximal besitzen darf (<range>_allcoins).
Damit kann man einstellen, dass MA3 bei Kursanstiege mit weniger Coins einsteigt und nur langsam Schritt für Schritt Positionen aufbaut. Auch kann man hiermit regeln, dass MA3 in z.B. Bären- oder Seitwärtsmärkten grundsätzlich weniger einsetzt als in Bullenmärkten.

xxxx_rsi_buy und xxxx_rsi_sell
die Schwellwerte für den Kauf oder Verkauf

xxxx_rsi_buy_type und xxxx_rsi_sell_type
Erlaubt sind die Werte up, down, above, below, hill, valley, off.

- "off": schaltet den RSI aus, die RSI-Bedingung ist dann immer true
- "up": die RSI-Bedingung ist erfüllt, wenn der RSI der zweitneusten Candle noch unter der angegebenen RSI-Schwelle liegt und der RSI der aktuellen Candle darüber
- "down": die RSI-Bedingung ist erfüllt, wenn der RSI der zweitneusten Candle noch über der angegebenen RSI-Schwelle liegt und der RSI der aktuellen Candle darunter
- "above": die RSI-Bedingung ist erfüllt, wenn der aktuelle RSI-Wert über der angegebenen RSI-Schwelle liegt
- "below": die RSI-Bedingung ist erfüllt, wenn der aktuelle RSI-Wert unter der angegebenen RSI-Schwelle liegt
- "hill": die RSI-Bedingung ist erfüllt, wenn der vorletzte RSI höher als die angegebene RSI-Schwelle liegt, und höher als der vorvorletzte und der neuste - sie formen einen "Hügel"
- "valley": die RSI-Bedingung ist erfüllt, wenn der vorletzte RSI tiefer als die angegebene RSI-Schwelle liegt, und tiefer als der vorvorletzte und der neuste - sie formen ein "Tal"

xxx_rsi_candles
gibt an, wieviele Candles zur Berechnung des RSI herangezogen werden sollen

side_bblow_buyval = -20 -10 0 10 20 30 40 50 100
Beim Kaufen soll der Coin weiter unten im BB-Kanal liegen. Hiermit gibt man an, wie tief der Preis liegen soll.
0 entspricht dabei dem unteren Bollinger Band, 50 der Mitte und 100 dem oberen Band. Negative Werte liegen somit unter dem unteren Band.
Sinnvolle Werte liegen etwa zwischen -25 und +30.

side_bbhigh_sellval = -20 -10 0 10 20 30 40 50 100
Beim Kaufen soll der Coin weiter oben im BB-Kanal liegen. Hiermit gibt man an, wie hoch der Preis liegen soll.
0 entspricht dabei dem oberen Bollinger Band, 50 der Mitte und 100 dem unteren Band. Negative Werte liegen somit über dem oberen Band.
Sinnvolle Werte liegen etwa zwischen +25 und -25.

side_below_min (nur im Range Side)

Dieser Wert gibt an, um welchen Bruchteil der Coin aktuell mindestens unter seinem BB-Mittelwert liegen muss, um gekauft werden zu dürfen. ("Hat sich der Coin weit genug unter seinen Mean begeben?")
Damit stellt man sicher, dass nicht auf Miniveränderungen reagiert wird, deren Gewinn vermutlich nicht einmal die Tradegebühren decken könnte, sondern erst auf merkliche Veränderungen.

Beispiel:
Binance verlangt inzwischen 0.15% Gebühren pro Trade, in Summe fallen also für Kauf und Verkauf 0.30% an. Setzt man side_below_min = 0.01, dann darf der Bot nur kaufen, wenn der Kurs mindestens um 1.0% unter seinem Mittelwert liegt. Damit würden bei einer Rückkehr zum Mittelwert zumindest 0.7% Gewinn übrigbleiben.

Zu den Stopp Loss-Parametern
Die Ergebnisse der Backtests legen nahe, dass die Nutzung von Stop Loss bei MA3 leicht kontraproduktiv sein kann. Man verliert meist mehr an Gewinnen, als man vor Verlusten retten kann. Es empfiehlt sich, sl_loss so hoch zu wählen, dass er normalerweise nicht in Aktion tritt und nur noch vor extremen Kursverfällen eingreift.

sl_takeprofit_side
Im Gegensatz zu QuadReg kann eine vorsichtige Nutzung von Take Profit tatsächlich die Endergebnisse im Bereich Side oder Bear leicht verbessern und MaxDD leicht senken.