Testing Kandle-based strategies using an ocaml simulator

Mangrove Research

November 5, 2024

You need to install ocaml.
Eg via the opam package manager

1 Setup

1.1 Setup price series

To run a test we first need to get a price series.
There are two options.
We can use a csv file of historical data (see `h` function):

```
let h
~number_of_lines:nb_lines (* nb_lines to read after header *)
~filename:filename        (* path to csv file *)
~column:col               (* index of price column in file *)
~splitchar:c              (* split char in the csv file *)
= ...
```

Or we can use a BM or GBM with parameters:

```
let gbrowmo
~initial_value:ival       (* initial price value *)
~drift:drift              (* drift of BM *)
~volatility:vol           (* volatility of BM *)
~timestep:dt              (* integration step *)
```

```
~duration:duration        (* total duration *)
= ...
```

Both methods output a price series as an array of floats.

1.2 Setup Kandle

To define a Kandle instance one needs:
- a (relative) price grid
- an initial allocation of capital (in quote),
- and the fraction α of the said quote to be swapped to base (to provide Kandle's asks)

The relative price grid is defined by picking any 2 of the following 3 parameters:
- (half) number of points,
- range multiplier,
- gridstep (aka ratio)

The initial price (or entry price) used to center the grid is taken from the price series (hence the name 'relative' price grid).

2 Plain simulation

A simulation is the interaction of a Kandle with a price series.

The output of a simulation run is a 6-uple:
- the return
- the number of up crossings and up exits (above range)
- the number of down crossings and down exits (below range)
- the final price of the price series

To set up a run of the simulation, we combine both sets of parameters (using here range and gridstep for the price grid).

Inputs:

```
let sim
~rangeMultiplier:rangeMultiplier (* pmax/p0 = p0/pmin *)
~gridstep:gridstep               (* ratio of price grid *)
~quote:qB                        (* total budget in quote *)
~cashmix:alpha                   (* 0 <= alpha <= 1 *)
~start:start                     (* entry in position *)
~duration:duration               (* number of price moves *)
```

```
~price_series:price_series       (* series of price *)
```

Outputs:

```
(mtmf /. mtmi -. 1.),
!rebu,
!cross_above,
!rebd,  !cross_below,
price_series.(start + duration - 1)
```

Fig. 1 shows an example of a 1000 runs, 10000 steps each.

The syntax is

```
./a.out 1000 10 1.25
```

with arguments (in this order): number of repetitions, number of points on the grid, and ratio of the grid.

The compiled code will generate a csv file `return_vs_final_price_scatter_1.25_10_stopped.csv`.

The csv file can be visualised using a python script called `hist.py` with similar syntax `python hist.py 10 1.25`.

3 Simulation with exit

There is a variant, called `sim_stopped`, that will stop if the price exits the range. So no need to specify a duration. Either the simulation will stop (an return the time of exit and price), and it will run all the way to the end of the price series.

Inputs:

```
let sim_stopped
 ~rangeMultiplier:rangeMultiplier
 ~gridstep:gridstep
 ~quote:qB
 ~cashmix:alpha
 ~start:start
 ~price_series:price_series
```

Outputs:

```
mtmf /. mtmi, (* growth rate, because more compositional *)
!rebu, !upper_exit,
```

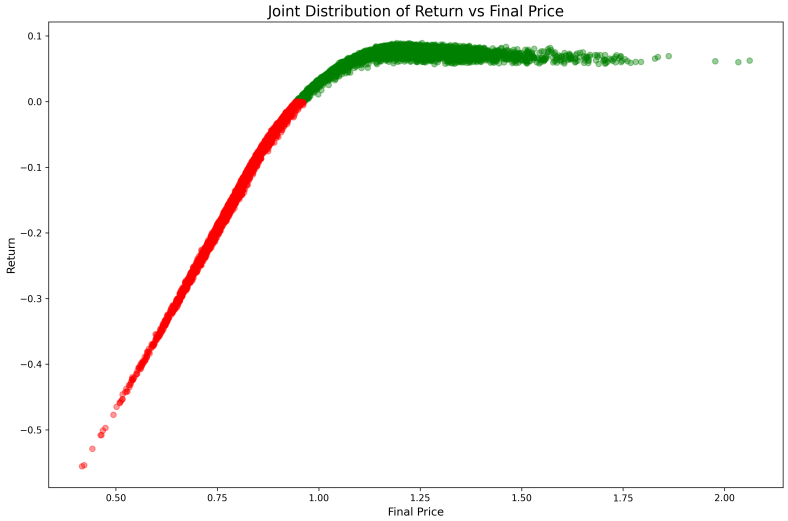

Figure 1: *On the x-axis the final price, on the y-axis the return. In red (green) realisations with a negative (positive) return.*

```
!rebd, !lower_exit,
!current_index,
!current_price
```

NB1 - `sim_stopped` returns the growth rate `mtmf /. mtmi` not the return `mtmf /. mtmi` -1.

Fig. 2 shows an example of a 1000 runs, 10000 steps each (under 2sec of execution, using the native compiler ocamlopt). We see the green and red walls at the ends of the range near the $\pm 20\%$ mark.

4 Simulation with reset

There is also a `sim_reset` variant where each time the price exits the range, the strategy re-enters with the same grid translated to the new current price.

Note that price-tracking Kandle is subtly different from a vanilla MM strategy which translates its price grid each time the price moves (at a given frequency), regardless of whether one of its offers was consumed or not. The difference is that in the latter case even

Figure 2: *On the x-axis the final price, on the y-axis the return. The Kandle grid has 4 points (on each side of the entry price of* 100*), with a ratio of* 1.05.

in the absence of crossing the price stays always ϵ away from the current price.

Todo: implement the vanilla MM strategy and measure the payoff difference (MM should be better).

Fig. 3 shows an example where $r = 1.02$, with 1 offer on each side of the entry price, and a price GBM $\mu = 0$, $\sigma = 0.08$. One sees that the payoff shape is very different, and looks similar to a buy and hold with half the cash invested (black line) with some low amount of noise induced by the path-dependent crossings.

4.1 Discussion of price-tracking Kandle (sim reset)

Idea: There is a strong correlation between the last price and the pay-off (or return). One can 'neutralise' the price-deterministic part of the pay-off by adding a correction using the regression line with slope b, intercept c.

Write priceChange for the relative price change lastPrice/initialPrice $- 1$:

$$(\text{priceChange}, \text{return}) \quad \longrightarrow \quad (\text{lastPrice}, \text{return} - (b \times \text{priceChange} + c))$$

If $b = 1/2$, $c = 0$ we are subtracting the price-deterministic return of the buy-and-hold where half of the initial cash is invested in the underlying. The exact offsets b, c, to be determined by fitting a regression line to a numerical experiment depend on (r, σ).

Figure 3: *The regression line is close to that of a buy-and-hold strategy with half the cash invested in the risky asset.*

Concretely that can be done eg by shorting a certain amount b of the underlying on a perp market, or by borrowing some of the underlying on a mending platform. What does c represent: the deterministic cost of financing the strategy (a sort of premium)?

For large ratios $r > 1.1$ (aka grid steps), there are < 1 crossings on average and the constant is very nearly $b = \frac{1}{2}$ (ie a complete buy-and-hold). As r decreases, the mean number of crossings increases and the constant b, c seem to decrease. To investigate empirically.

QS1 - Is there for given σ a value $r^\star(\sigma)$ of r that maximises the mean return? [similar to prior work done for plain Kandle]

Todo: Try same experiments with multiple price points (not just 2)

Can we combine sim with reset with a stop loss? Eg, is it useful to stop resetting after k moves, or a certain minimum residual capital (or on a momentum signal)?

4.2 Analysis of sim reset

Let $\alpha \in [0, 1]$ be the cash ratio (amount of initial capital kept in quote). Set $\alpha' := 1 - \alpha$ to simplify notations.

Proposition 1 *The total gain under a given price run is:*

$$\Gamma(\alpha) \quad = \quad (\alpha + \alpha' r)^{n_u} (\alpha + \alpha'/r)^{n_d} \tag{1}$$

where n_u is the number of up-crossings, n_d is the number of down-crossings along the price trajectories.

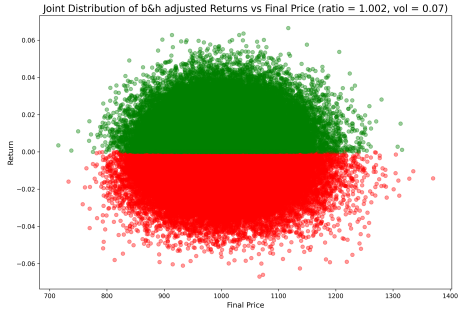

Figure 4: *adjusted return vs exit price (ratio = 1.002, vol = 0.07)*

Proof: We start with a capital v_B^0 of cash (base), of which we keep αv_B^0, and buy $\alpha' v_B^0/p_0$ of base.

Suppose an uc:

$$\begin{pmatrix} \alpha' v_B^0/p_0 \\ \alpha v_B^0 \end{pmatrix} \longrightarrow^c \begin{pmatrix} 0 \\ \alpha v_B^0 + (rp_0)\alpha' v_B^0/p_0 \end{pmatrix}$$

Our new position is worth $v_B^0(\alpha + \alpha' r)$, so a growth rate $\gamma_u = \alpha + \alpha' r$.

Symmetrically for a dc:

$$\begin{pmatrix} \alpha' v_B^0/p_0 \\ \alpha v_B^0 \end{pmatrix} \longrightarrow^d \begin{pmatrix} \alpha' v_B^0/p_0 + \alpha v_B^0/(p_0/r) \\ 0 \end{pmatrix}$$

with a new position worth $v_B^0(\alpha'/r + \alpha)$, and we find $\gamma_d = \alpha + \alpha'/r$. □

NB - Importantly, in the proof above, we assume that the price at which we can resell what we buy on a down-crossing is the same price at which we have bought. In reality, on a down-crossing $p_n > p_0/r > p_{n+1}$ and there is a loss factor proportional to $p_{n+1}/(p_0/r) < 1$ (and, symmetrically one could have sold higher for an upcrossing with a "manque à gagner" proportional to $p_{n+1}/(rp_0)$). In the simulations we do mark to market at the real price p_{n+1}.

Note that the crossing numbers are random and only depend on vol-and-grid, σ and r.

As $\alpha + \alpha' r > 1$ (barycenter of 1 and $r > 1$), and $\alpha + \alpha'/r < 1$ (barycenter of 1 and $1/r < 1$), it follows that $\Gamma > 1$ ie the strategy is profitable along a given price trajectory as soon as $n_u > n_d$.

In particular if $\alpha = \alpha' = \frac{1}{2}$, we get:

$$\begin{aligned} \log(\Gamma(1/2)) &= n_u \log(\tfrac{1+r}{2}) + n_d \log(\tfrac{1+1/r}{2}) \\ &= N(f_u \log(\tfrac{1+r}{2}) + f_d \log(\tfrac{1+1/r}{2})) \end{aligned}$$

where in the last line we assume $N = n_u + n_d > 0$ (not always true! especially for large ranges).

The above is > 0 iff:

$$f_d/f_u \quad \leq \quad \frac{\log(1+r) - \log 2}{\log 2 - \log(1+r^{-1})} =: k(r)$$

where $k(r) \geq 1$ is increasing and converges to $\log(1+r)/\log(2) - 1$ for large r. This suggests that larger rs are more likely to generate profits. To make sure we need to study how the f_d, f_u distribution depends on r. It seems unlikey but it could be that larger r are biased downwards, ie f_d/f_u is increasing function of r. [in sims up and down crossings look equally likely]

Todo: study the distribution of f_u as a function of r and σ. [Also look at std]

We can see the effect of changing α.

If $\alpha = 0$ (base only), $\Gamma(0) = r^{n_u - n_d}$. If we go full base, we win iff the process crosses more up than down.

If $\alpha' = 0$ (quote only), $\Gamma(1) = 1$. Indeed if we never buy the underlying, all that can happen is the execution of a bid, the output of which that strategy re-sells immmediately at same price (assuming no fees, no slippage), and therefore the initial wealth never changes. Ie (modulo fees and slippage), setting $\alpha = 0$ amounts to not playing.

4.3 Other examples

The cost of financing a price-tracking Kandle (intercept) with $r = 1.005$, $\sigma = 0.06$ is a negative return of $\sim 4.6\%$. See the comparison with the B&H line which has an intercept of zero (Fig. 5). The same cloud of points is represented with the regression line subtracted Fig. 6.

It would be interesting to understand how the parameters of the regression line depend on r and σ.

Fig. 7 gives the distribution of adjusted returns. From the figure it seems one can decompose the return into a random normal component and a deterministic one :

$$ret(r, \sigma) \quad = \quad \mathcal{N}(0, s(r,\sigma)) + (c(r,\sigma) + b(r,\sigma)(\Delta p/p_0)) \tag{2}$$

where $\Delta p/p_0$ is the relative price change at the end of the price sequence.

Todo: explore the coefficients in (Eq 2) above.

4.4 Random remarks

A small remark: concretely, when running on chain, price-tracking Kandle does not have access to the next price (as the simulation does) only that $p_{n+1} > rp_n$; to correct this one can use multiple offers to improve the upper bound, and/or use an oracle at update time (eg consult another liquidity source).

Euler-Maruyama for large $\sqrt{dt}$, large σ can lead to negative prices in a GBM; so better use the integral form.

Figure 5: *non-adjusted returns vs relative price change ($r = 1.005$, $\sigma = 0.06$); the dotted line is the buy-and-hold return for half the capital invested in base ($\alpha = 1/2$); the intercept of the regression line is the cost of the strategy per unit of capital provided.*

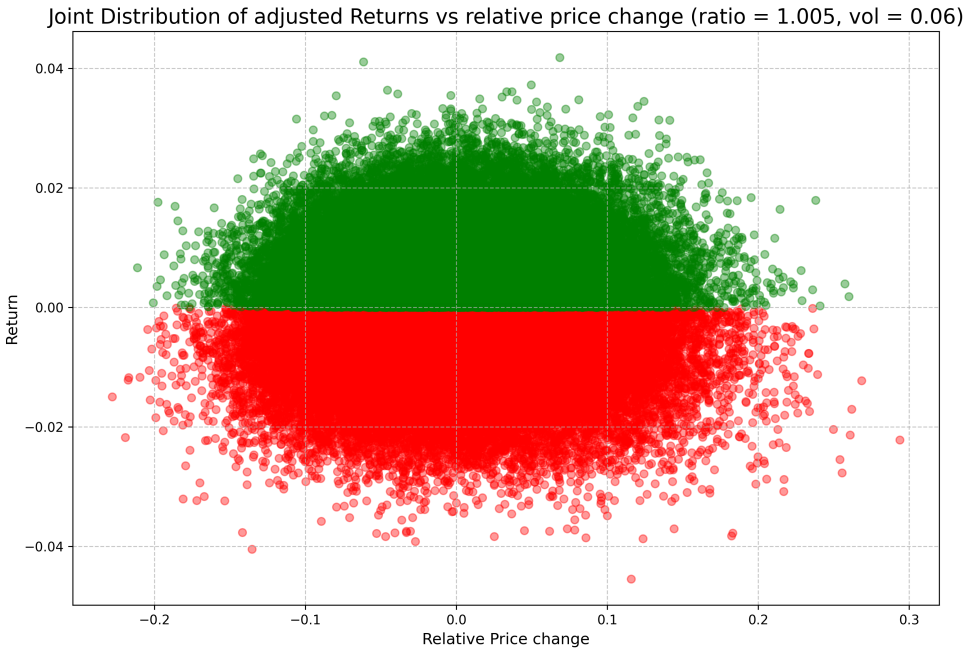

Figure 6: *adjusted returns vs relative price change* ($r = 1.005$, $\sigma = 0.06$).

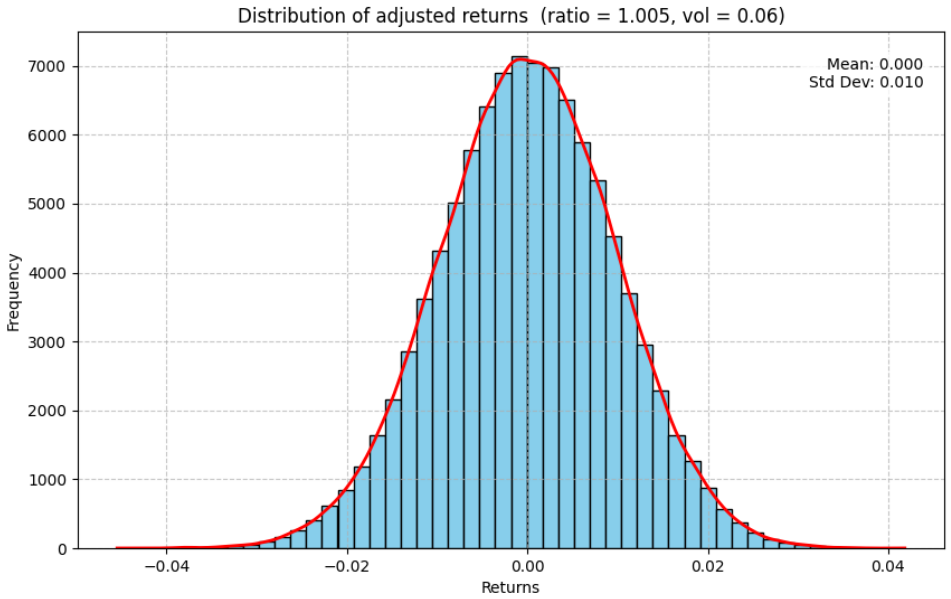

Figure 7: *histograms of adjusted returns* ($r = 1.005$, $\sigma = 0.06$).