



Developer KIT

**ProRealTime**.com  
Online streaming charts & quotes

**ProBacktest** 1.0c  
2004 edition



# SYNOPSIS

ProBacktest presentation .....	2
ProBacktest commands .....	3
Orders simulation .....	3
Stop-loss programming .....	7
Backtest strategies .....	8
State variables .....	10
Variables of position following.....	11
Bars access relatively to the latest executed order .....	13
Examples .....	14
Penny stocks .....	14
Optimized entry .....	15
Trend detected on price .....	16
Sell in may and go away ! .....	17
Intraday break-out .....	18



## ProBacktest presentation

ProBacktest is the backtesting module of ProRealTime. It is an extension of the ProBuilder language. It allows you to create and test trading strategies using the well known TA indicators or your own indicators created within the ProBuilder language

ProBacktest is a BASIC-like language for automatic trading systems. You will thus be able to create your own systems using any price provided by ProRealTime software. Quotes available are :

- Opening price of each bar
- Closing price of each bar
- Highest price of each bar
- Lowest price of each bar
- Volume of each bar

Bars are those displayed in ProRealTime software.

An arrow is displayed for each buying or selling order generated from your system. Such arrows indicate the buying or selling prices of your simulated orders. Furthermore, ProBacktest calculates an equity curve that shows the performance of your system. The equity curve is displayed as an indicator upside the quotes.

The ProBacktest takes into account the values of each bar from the 1st one to the latest one and executes your own program in order to determine the orders to simulate for each bar.

## ProBacktest commands

### Orders simulation

An order is defined by : the transaction characteristic (buy/sell), the number of shares involved, the conditions of its execution and the date of execution. One or more of these parameters may be omitted.

### The transaction characteristic

#### Buy, sell

**BUY** *count* **SHARES** (AT MARKET | AT price **LIMIT** | AT price **STOP**)

This command opens a long position. If the backtest portfolio is short when such order is executed, then short positions are bought as well in order to have a long position of *count* shares. If the backtest portfolio is already long when such order is executed, then it depends on the money management rules you enter.

**SELL** [*count* **SHARES**] (AT MARKET | AT price **LIMIT** | AT price **STOP**)

This command sells the long positions. If the backtest portfolio is empty or short when such order is executed, nothing happens. If the backtest portfolio is long then *count* shares are sold. Even if *count* is greater than the number of shares you own, we don't short the market but the long position is completely sold. If the *count* parameter is omitted, then all shares are sold.

#### Sellshort, exitshort

**SELLSHORT** *count* **SHARES** (AT MARKET | AT price **LIMIT** | AT price **STOP**)

This command opens a short position. If the backtest portfolio is long when such order is executed, then long positions are sold as well in order to have a short position of *count* shares. If the backtest portfolio is already short when such order is executed, then it depends on the money management rules you enter.

**EXITSHORT** [*count* **SHARES**] (AT MARKET | AT price **LIMIT** | AT price **STOP**)

This command sells the short positions. If the backtest portfolio is empty or long when such order is executed, nothing happens. If the backtest portfolio is short then *count* shares are bought. Even if *count* is greater than the number of shares you short, we don't buy the market but the short position is completely bought. If the *count* parameter is omitted, then all shares are bought.



## Number of shares

The number of shares may be entered as an amount of cash units, as a fraction of the capital, or as a fraction of the available cash.

- **SHARES**                    number of shares
- **CASH**                        amount in cash unit (like €or \$)
- **%CAPITAL**                fraction of the current capital (shown by the equity curve)
- **%LIQUIDITY**              fraction of the current available cash

### Example Cash

REM Buy 1000€ (if quotes are expressed in €)  
BUY 1000 Cash AT MARKET

### Example %Capital

REM Buy with 70% of the current capital (shown by the equity curve for each bar)  
BUY 70 %Capital AT MARKET

### Example %Liquidity

REM Buy with 40% of the current available cash  
BUY 40 %Liquidity AT MARKET

### Remark :

When you exit the market (*SELL, EXITSHORT*) the count of shares is optional.

REM Exits all long positions  
SELL AT MARKET



## Execution condition

Three different orders are available : at market price, at the best limit, and stop orders.

- **AT MARKET** at market price
- **AT price LIMIT** at the best limit
- **AT price STOP** stop orders

### Example AT MARKET

REM Buy 100 shares at market price  
BUY 100 Shares AT MARKET

### Example LIMIT

REM Buy 100 shares at the best limit 15.45€  
BUY 100 Shares AT 15.45 LIMIT

### Example STOP

REM Buy 100 shares with a stop order at 16.18€  
BUY 100 Shares AT 16.18 STOP



## The date of execution

If not precised, orders are simulated on the next bar. However, orders “at market” can be simulated before or after such bar, thanks to one of the below keywords:

- **ThisBarOnClose** at the close of the current bar
- **NextBarOpen** at the open of the next bar (**default order**)
- **NextBarClose** at the close of the next bar
- 
- **TodayOnClose** at the close of the current day(used in intraday)
- **TomorrowOpen** at the open of the day after (used in intraday)
- **TomorrowClose** at the close of the day after (used in intraday)

### Example ThisBarOnClose

REM Buy 100 shares at the close of the current bar  
BUY 100 Shares AT MARKET ThisBarOnClose

### Example NextBarClose

REM Buy 100 shares at the close of the next bar  
BUY 100 Shares AT MARKET NextBarClose

### Example TodayOnClose

REM Buy 100 shares at the close of the current day (intraday)  
BUY 100 Shares AT MARKET TodayOnClose

### Example TomorrowOpen

REM Buy 100 shares at the open of the next day (intraday)  
BUY 100 Shares AT MARKET TomorrowOpen

### Example TomorrowClose

REM Buy 100 shares at the close of the next day (intraday)  
BUY 100 Shares AT MARKET TomorrowClose



## Stop-loss programming

### Set Stop

#### SET STOP price

This command allows you to add a customized stop that uses your own algorithms. Please note that 4 common Stop orders can be simulated within the ProBacktest window.

#### Example Set Stop

REM A following stop

```
IF close > AVERAGE[30] AND NOT OnMarket THEN
  BUY 100 Shares AT MARKET
```

```
REM Compute the difference between ideal price and real price
Spread = openOfNextBar - low
```

```
REM Initialize the highest quote since the entry on market
Up = openOfNextBar
ENDIF
```

```
REM Updates the stop to let its distance to the highest price constant
IF OnMarket THEN
  Up = MAX(Up, high)
  SET STOP (Up - Spread)
ENDIF
```



## **Backtest strategies**

### **AS**

Commands and variables are linked to one or more strategies. You can define as many strategies as you want.

If you never refer to a strategy in your code, then ProBacktest creates only one strategy and all your commands are linked to it.

If you want to define several strategies within the same system, you have to use the keyword **AS**. Indeed when you enter a command or a variable with an explicit strategy, the simulation applies only on this strategy.

But, if you don't specify any strategy then your commands apply on all your strategies.

#### **Example orders linked to a single strategy**

REM Strategy named "moving average"

```
IF close > AVERAGE[30](close) AND NOT OnMarket THEN
  BUY 100 Shares AT MARKET AS « moving average»
ENDIF
```

Then each position characteristics is evaluated in the context of its strategy. For instance, you can have a long global position in the market while you are short for a certain strategy.

But in the following example...

#### **Example orders linked to all the strategies**

REM Order computed on all the strategies  
SELL 50 Shares AT MARKET

The sell order is simulated on all your strategies. If you have 2 strategies, it simulates the sell of  $2 * 50$  shares = **100 shares**. It is useful when you want to sell all your long positions with only one instruction.



This principles apply also to any backtesting variable or command.

**Example EntryIndex, strategies**

REM Strategy "moving average"

```
IF close > AVERAGE[30](close) AND NOT OnMarket THEN  
  BUY 100 Shares AT MARKET AS « moving average»  
ENDIF
```

REM Sell on the breakout of the low of the entry bar

```
IF close < low[BarIndex - EntryIndex AS « moving average»] THEN  
  SELL AT MARKET AS "moving average"  
ENDIF
```



## State variables

### OnMarket, LongOnMarket, ShortOnMarket

These variables allow you to know the state of the backtest portfolio. The state of the portfolio can be long, short or empty.

### Description

Such State variables are important since they tell you the current status of your backtest portfolio.

In major cases, when you backtest a strategy, buying or selling orders are executed depending on the status of your current positions in the market. Indeed, an uncovered sell or an exit long order are two different things. In a case the position is opened, in the other it is closed.

Conditions associated to the execution of orders are not always the same. For instance, you will open a position when a technical signal is given, and you will close it for money management considerations.

### Example OnMarket

```
REM Buy on moving average breakout
```

```
c1 = close > AVERAGE[30](close)
```

```
REM This condition is sufficient to enter the market
```

```
IF NOT OnMarket THEN
```

```
    IF c1 THEN
```

```
        BUY 10 SHARES AT MARKET
```

```
    ENDIF
```

```
ENDIF
```

```
REM But we have 2 conditions to exit
```

```
IF OnMarket THEN
```

```
    REM The following adds a condition given by a « new low » breakout
```

```
    C2 = close < LOWEST[10](low[1])
```

```
    REM The exit depends on a double condition
```

```
    IF NOT c1 OR c2 THEN
```

```
        SELL 10 SHARES AT MARKET
```

```
    ENDIF
```

```
ENDIF
```



## Variables of position following

### CountOfLongShares, CountOfShortShares, CountOfPosition

These variables are:

- The count of shares in a long position (0 if not long)
- The count of shares in a short position (0 if not short)
- The count of accumulated positions (if pyramid is allowed)

### Description

These variables give a more precise information than the state variables. Indeed, they allow you to know your current position in the market but also they give you the number of shares in the market and the number of executed orders.

For instance, you can count orders to pyramid only three times:

#### Example CountOfLongShares, CountOfPosition

```

REM Buy on moving average breakout
c1 = close > AVERAGE[30](close)

REM This condition is sufficient to enter the market
IF NOT OnMarket THEN

    IF c1 THEN
        BUY 10 SHARES AT MARKET
    ENDIF

ENDIF

REM Pyramids 3 times if the buy condition is still true
IF OnMarket THEN

    REM The following adds a condition given by a « new low » breakout
    c2 = close < LOWEST[10](low[1])

    REM The exit depends on a double condition
    IF NOT c1 OR c2 THEN
        SELL CountOfLongShares SHARES AT MARKET

    REM Pyramids 3 times until (while the exit condition is false)
    ELSIF CountOfPosition < 3 THEN
        BUY 10 SHARES AT MARKET
    ENDIF

ENDIF

```

**Remark :**

Pyramids are allowed only if the option « cumulate positions » is activated in the capital management section of the ProBacktest windows.

If you check the option « 1 stop for all the positions » then all positions are merged in one, so **CountOfPosition** can't be higher than 1.



## **Bars access relatively to the last executed order**

### **EntryIndex**

This is the index of the bar on which the latest order has been executed.

### **Description**

This constant allows you to consider the candlestick where the system entered in the market, and to compute an associated stop.

#### **Example EntryIndex**

```

REM Buy in case of a moving average breakout
IF NOT OnMarket THEN
    IF close > AVERAGE[30](close) THEN
        BUY 100 %CAPITAL AT MARKET
    ENDIF
ENDIF

REM Exit under the lowest price of the candlestick of entry
IF OnMarket THEN
    SELL AT low[BarIndex - EntryIndex] STOP
ENDIF
    
```

### **EntryQuote**

This is the execution price of the last simulated order.

### **Description**

This variable allows you to compute a stop linked to the last entry point.

#### **Example EntryQuote**

```

REM Buy in case of a moving average breakout
IF NOT OnMarket THEN
    IF close > AVERAGE[30](close) THEN
        BUY 100 %CAPITAL AT MARKET
    ENDIF
ENDIF

REM Exit under the entry price
IF OnMarket THEN
    SELL AT EntryQuote STOP
ENDIF
    
```



## Examples

### *Penny stocks*

#### Description

Here is an example of a long system conceived to generate few orders and detect interesting opportunities.

It is composed of a filter based on the absolute value of quotes. This is a “penny stocks” filter that avoid to do anything where quotes are higher than 10 (€ \$, £...)

The entry condition is a new highest breakout, confirmed by a upward trend on a moving average.

The exit signal is the break down of a moving average (we use a moving average on the low prices to limit wrong signals).

Indeed, we want to let profits increase even if we take some risks on the exit condition. The idea is that we can earn much more on winning trades with penny stocks (much more than 100%) than we can loose (less than 100%).

***Warning : such system may not be profitable. We just believe that it can help to understand how to program strategies. You should optimize it or even change completely the buying and selling conditions.***

#### Example PennyStocks

##### REM BUY CONDITION

REM if the highest of the bar higher than the highest of the 10 previous bars ?

c1 = **high** > highest[10](**high**[1])

REM is the highest of the bar higher than the 13bar exponential moving average of close ?

c2 = **high** > exponentialAverage[13](**close**)

REM is the close price lower than 10 ?

c3 = **close** < 10

REM is the candlestick white ?

c4 = **close** > **open**

IF c1 AND c2 AND c3 AND c4 THEN

BUY **70 %capital** AT MARKET

ENDIF

##### REM EXIT CONDITION

REM does the low of the bar crose under the 30bars exponential moving average of low ?

IF **low** CROSSES UNDER exponentialAverage[30](**low**) THEN

SELL AT MARKET

ENDIF



## Optimized entry

### Description

Here is the example of a “long” system. It aims at entering long position only after a sufficient decrease of prices inside a long trend.

The trend is detected by a 26 bars weighted moving average.

In order to buy at the « best » price in terms of rentability/risk, we will use the « Parabolic SAR » indicator. It a trend following indicator.

It confirms the long trend and it gives us the lowest level of price that does not invert the trend.

We will buy at the level of the “Parabolic SAR”. The risk is to see such level broken. However, taking into account the condition on the 26 bars weighted moving average, we have good probabilities to detect properly the long trade. And it is well known that there are more probabilities for a trend to continue than to invert itself. Our hypothesis would be invalidate if the weighted average was broken. This is our exit condition.

***Warning : such system may not be profitable. We just believe that it can help to understand how to program strategies. You should optimize it or even change completely the buying and selling conditions.***

### Example Optimized entry

```
REM Parabolic SAR as trade following indicator  
follow = SAR
```

```
REM is the close higher than the 26 bars weighted moving average ?  
ha1 = close > weightedaverage[26](close)
```

```
REM is the SAR indicator under the close?  
ha2 = close > follow
```

```
IF ha1 THEN  
  IF ha2 THEN  
    BUY 80 %capital AT follow LIMIT  
  ENDIF  
ELSE  
  SELL AT MARKET  
ENDIF
```



## Trend detected on price

### Description

Here is a long/short system that allows you to be always in the market. The goal of this system is to reverse the position to earn from panic buying/selling attitude.

We try to detect such panic from the prices. We assume that it is predictable thanks to the increase of volatility. Our hypothesis: abnormal strong variation means panic.

The money management is :70% of equity used on long position and 40% on short positions. Short positions should be less risky than long positions because it is well known that panic may be more important on downward trends rather than upward trends.

Furthermore, money management is harder with short positions (potential loss is unlimited !)

**Warning : such system may not be profitable. We just believe that it can help to understand how to program strategies. You should optimize it or even change completely the buying and selling conditions.**

### Example Trend detected on prices

```
REM compute the average variation of quotes
avt = AverageTrueRange[20](close)

REM buy condition: strong increase of prices (=panic buy)
IF close > close[1] + avt THEN
  BUY 70 %capital AT MARKET
ENDIF

REM short condition : strong decrease of prices (=panic sell off)
IF close < close[1] - avt THEN
  SELLSHORT 40 %capital AT MARKET
ENDIF
```



## ***Sell in may and go away !***

### **Description**

It is also possible to create system that don't take into account prices, but statistical numbers.

One of this facts is given by the saying « sell in may and go away ! ». An attentive study of markets on large histories shows that worst performances are those of the months between may and september.

We will check it with a trading system. So the system is simple: buy in october, and sell short in may.

It is a good system (in terms of drawdown) because he often changes the sens of exposition and escapes too long adverse trends (and furthermore the timing is good thanks to statistics!)

Warning : such system may not be profitable. We just believe that it can help to understand how to program strategies. You should optimize it or even change completely the buying and selling conditions.

### **Example Trend detected on months (!!)**

```
REM Sell short in may !  
IF Month = 5 THEN  
  
    SELLSHORT 50%capital AT MARKET  
  
REM Buy in october  
ELSIF Month = 10 THEN  
  
    BUY 50%capital AT MARKET  
  
ENDIF
```



## ***Intraday break-out***

### **Description**

Now let see a famous intraday system. It is a breakout system based on the limits given by the two first bars of the day.

Entry on market is determined by the breakout of one of these limits.

Exit occurs at 16 o'clock. (local hour)

*Warning : such system may not be profitable. We just believe that it can help to understand how to program strategies. You should optimize it or even change completely the buying and selling conditions.*

### **Example Intraday break out**

```

REM At the close of the second bar of the day (index 1 because it starts at 0)
IF intradayBarIndex = 1 THEN

    REM Compute the levels of the highest and the lowest of the 2 first bars of the day
    up = Highest[2](high)
    down = Lowest[2](low)

ENDIF

REM Buy / Sell on breakout between the 3rd bar and 16 o'clock (local hour)
IF intradayBarIndex > 1 AND Time < 160000 THEN

    REM Breakout of the highest
    IF close > up THEN
        BUY 70%capital AT MARKET
        SELL AT MARKET TodayOnClose

    REM Breakout of the lowest
    ELSIF close < down THEN
        SELLSHORT 70%capital AT MARKET
        EXITSHORT AT MARKET TodayOnClose

ENDIF

ENDIF

```