TradeBlazer(TB)公式编写技术要点

蔡云华

深圳开拓者科技有限公司

内容安排

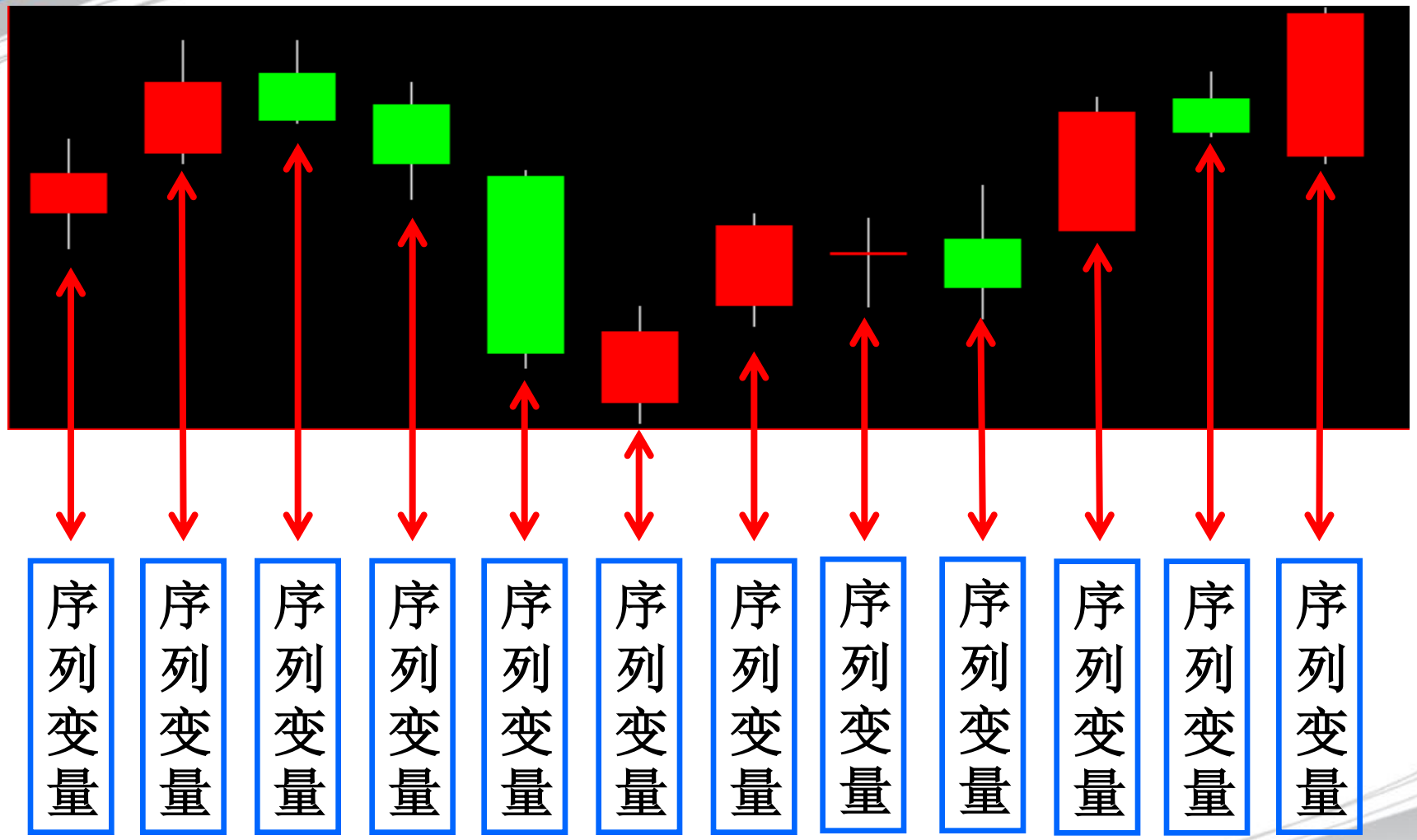
- 深入了解TB公式的运行机制
- 公式调试的一般方法
- 信号消失的根源及对公式编写的影响
- 序列变量、全局变量、读写数据库的使用技巧
- 序列函数问题的简单规避

Bar数据(K线数据)

当前时间周期下所有K线的相关数据，按照时间从先到后的顺序排列而成的序列数据。每根K线中包含的数据如下：

Bar数据	含义
Date	当前K线的日期
Time	当前K线的开始时间
Open	当前K线的开盘价
High	当前K线的最高价
Low	当前K线的最低价
Close	当前K线的收盘价（最新价）
Vol	当前K线成交量
OpenInt	当前K线持仓量
CurrentBar	当前K线的索引值（K线的编号，从0开始）
BarStatus	当前K线的状态值（0—第一根K线、2—最后即最新一根K线、1—其他K线）

序列数据



TB公式运行机制

从左到右，从上到下

历史回测和实时交易的区别

	历史回测	实时交易
Bar数据	确定不变	实时更新
公式运行	每根BAR一次	每个Tick一次
交易信号	固定不变	有可能变化
是否发单	否	是 （受公式机制控制）
函数调用	部分函数无效	有效

TB代码调试

- FileAppend函数
- Commentary函数
  - 在超级图表的当前BAR添加一行注释信息；
  - 参数：String strTip; // 提示的信息

TB代码调试

- FileAppend函数
- Commentary函数
  - 在超级图表的当前BAR添加一行注释信息；
  - 参数：String strTip; // 提示的信息

输出BAR数据

Sample:

```
Begin
        FileAppend("c:\\输出每根K线的Bar数据.txt","Date= "+DateToString(Date)
        +" Time= "+TimeToString(time)
        +" Open= "+Text(Open)
        +" High= "+Text(High)
        +" Low= "+Text(Low)
        +" Close="+Text(Close)
        +" CurrentBar= "+Text(CurrentBar)
        +" Barstatus= "+Text(BarStatus));
End
```

输出BAR数据到EXCEL

Sample:

```
Begin
    If(CurrentBar == 0)
    {
        FileAppend("c:\\输出每根K线的Bar数据.csv","CurrentTime,Date,Time,Open,
                High,Low,Close,CurrentBar,BarStatus");
    }
    FileAppend("c:\\输出每根K线的Bar数据.csv",
    DateTimeToString(CurrentDate+CurrentTime) + "," + DateToString(Date) + ","
    + TimeToString(time) + "," + Text(Open) + "," + Text(High) + "," + Text(Low)
     + "," + Text(Close) + "," + Text(CurrentBar) + "," + Text(BarStatus) );
End
```

通过BAR数据了解TB机制(1)

CurrentTime	Date	Time	Open	High	Low	Close	CurrentBar	BarStatus
2014/6/19 13:33	2014/6/19	13:24:00	2156	2156.4	2155.8	2156.4	0	0
2014/6/19 13:33	2014/6/19	13:25:00	2156.2	2156.4	2155.6	2156	1	1
2014/6/19 13:33	2014/6/19	13:26:00	2155.8	2156	2154.8	2155.2	2	1
2014/6/19 13:33	2014/6/19	13:27:00	2155.2	2156	2155.2	2155.4	3	1
2014/6/19 13:33	2014/6/19	13:28:00	2155.4	2155.6	2153.6	2154	4	1
2014/6/19 13:33	2014/6/19	13:29:00	2154.2	2155.2	2154	2154.8	5	1
2014/6/19 13:33	2014/6/19	13:30:00	2154.8	2154.8	2153.6	2154.6	6	1
2014/6/19 13:33	2014/6/19	13:31:00	2154.8	2155.2	2154.4	2154.8	7	1
2014/6/19 13:33	2014/6/19	13:32:00	2155	2155	2154.6	2154.6	8	1
2014/6/19 13:33	2014/6/19	13:33:00	2154.6	2154.8	2154.2	2154.4	9	2
2014/6/19 13:33	2014/6/19	13:33:00	2154.6	2154.8	2154.2	2154.4	9	2
2014/6/19 13:33	2014/6/19	13:33:00	2154.6	2154.8	2154.2	2154.6	9	2
2014/6/19 13:33	2014/6/19	13:33:00	2154.6	2154.8	2154.2	2154.6	9	2
2014/6/19 13:33	2014/6/19	13:33:00	2154.6	2154.8	2154.2	2154.6	9	2
2014/6/19 13:33	2014/6/19	13:33:00	2154.6	2154.8	2154.2	2154.6	9	2
2014/6/19 13:33	2014/6/19	13:33:00	2154.6	2154.8	2154.2	2154.8	9	2
2014/6/19 13:33	2014/6/19	13:33:00	2154.6	2154.8	2154.2	2154.8	9	2

通过BAR数据了解TB机制(2)

CurrentTime	Date	Time	Open	High	Low	Close	CurrentBar	BarStatus
2014/6/19 13:33	2014/6/19	13:33:00	2154.6	2155.4	2154.2	2155	9	2
2014/6/19 13:33	2014/6/19	13:33:00	2154.6	2155.4	2154.2	2155.4	9	2
2014/6/19 13:33	2014/6/19	13:33:00	2154.6	2155.4	2154.2	2155.4	9	2
2014/6/19 13:33	2014/6/19	13:33:00	2154.6	2155.4	2154.2	2155.2	9	2
2014/6/19 13:33	2014/6/19	13:33:00	2154.6	2155.4	2154.2	2155.2	9	2
2014/6/19 13:33	2014/6/19	13:33:00	2154.6	2155.4	2154.2	2155.4	9	2
2014/6/19 13:33	2014/6/19	13:33:00	2154.6	2155.4	2154.2	2155.4	9	2
2014/6/19 13:33	2014/6/19	13:33:00	2154.6	2155.4	2154.2	2155.4	9	2
2014/6/19 13:33	2014/6/19	13:33:00	2154.6	2155.4	2154.2	2155.4	9	2
2014/6/19 13:33	2014/6/19	13:33:00	2154.6	2155.4	2154.2	2155.4	9	1
2014/6/19 13:33	2014/6/19	13:34:00	2155.4	2155.4	2155.4	2155.4	10	2
2014/6/19 13:33	2014/6/19	13:34:00	2155.4	2155.4	2155.4	2155.4	10	2
2014/6/19 13:34	2014/6/19	13:34:00	2155.4	2155.4	2155.4	2155.4	10	2
2014/6/19 13:34	2014/6/19	13:34:00	2155.4	2155.4	2155.4	2155.4	10	2
2014/6/19 13:34	2014/6/19	13:34:00	2155.4	2155.4	2155.4	2155.4	10	2

TB图表讯号及发单处理机制

- 图表讯号与交易指令
- 持仓类函数(MarketPosition、CurrentContracts等)
- 虚拟账户
- 发单机制
  - 重复发单控制
  - 连续建仓次数
  - 同时满足买卖条件的发单
- 交易助手和监控器

交易指令 – Buy/Sell

- Buy -- 平掉所有空头持仓，开多头仓位;
- sell -- 平掉指定多头持仓;
- Sellshort -- 平掉所有多头持仓，开空头仓位;
- Buytocover -- 平掉指定空头持仓。
- 参数:
  - Numeric Share 买入数量，默认=0时，使用系统设置参数
  - Numeric Price 买入价格，为浮点数，默认=0时为使用现价(历史Bar为Close)。

交易指令 A_SendOrder

- 针对当前公式应用的帐户、商品发送委托单。
- 该函数直接发单，不经过任何确认，并会在每次公式计算时发送，一般需要配合着仓位头寸进行条件处理，在不清楚运行机制的情况下慎用。
- 不能使用于历史测试，仅适用于实时行情交易。
- 参数：
  - BuyOrSell ： 买卖类型，买Enum_Buy/卖Enum_Sell；
  - EntryOrExit: 开平仓类型, 开仓 Enum_Entry / 平仓Enum_Exit
  - / 平今 Enum_ExitToday；
  - fLot 委托单的交易数量；
  - fPrice 委托单的交易价格。

信号消失的根源及对公式编写的影响

- **最根本原因：**交易(开仓或平仓)判断条件的结果改变了
- **解决办法：**保持判断的结果不变，比如：
  - 用上一根Bar的数据来判断
  - 用最高价(最低价)来判断价位的突破(突破开仓、止损)
  - 历史K线和实时K线分别判断
- **带来的公式编写问题：**
  - 开仓BAR一般无法止损
  - 跟踪止损时无法以当前BAR的最高价（最低价）为基准来计算回落比例

序列变量(数据)

- 序列变量是和K线完全对应的一组数据。
- BAR数据是系统提供的序列数据
- 序列数据可以通过XXX[nOffset]进行数据回溯。
- 通过序列变量可以实现很多复杂的算法：从简单的求和，求平均，求累计值，到稍复杂的求最大值，最小值，到更复杂的求相关系数，之字转向等。
- 用序列数据来实现算法比用循环来实现效率更高

数据回溯越界

- 序列变量(数据)回溯可能存在越界的情况。
- 比如在CurrentBar = 0 的Bar上，如果再往前回溯就越界了，这种情况下，TB会自动用第一个有效值代替。所以，在第一根BAR
- 上面的举例是向前回溯，还有一种情况是向后回溯，比如Date[-1]，但除非需要特殊控制，不建议这么使用。最后一个Bar的Date[-1]是无效值。

序列变量的生命周期

- 简单变量在每次公式运行时，被赋默认值，公式运行完后不再存在
- 序列变量每次公式运行时，除了第一根BAR会被赋默认值，其他BAR会自动传递上一根BAR的值，公式运行完后仍然存在，并传递给下一根BAR
- 但序列变量在每个Bar都有且只有一个值，而且每次刷新数据运行公式时，都会先回到初始值。只有一根BAR的最后一个Tick公式运行完后，序列变量的值才能保留下来。
- 因此，序列变量无法记录盘中每个Tick运行公式产生的数据；比如：我们要对每个Tick计数，用序列变量就做不到

测试不同变量的生命周期

```
Vars
    Numeric jdbl;
    NumericSeries xlbl;
Begin
    If(CurrentBar == 0)
    {
        FileAppend("c:\\简单变量和序列变量的生命周期.csv","CurrentTime,Date,Time,CurrentBar,BarStatus,简单变量,序列变量");
    }
    FileAppend("c:\\简单变量和序列变量的生命周期.csv",DateTimeToString(CurrentDate+CurrentTime)
    + "," + DateToString(Date) + "," + TimeToString(time) + "," + Text(CurrentBar) + ","
    +Text(BarStatus) + "," + Text(jdbl) + "," + Text(xlbl) );

    jdbl = jdbl + 1;
    xlbl = xlbl + 1;

    FileAppend("c:\\简单变量和序列变量的生命周期.csv",DateTimeToString(CurrentDate+CurrentTime)
    + "," + DateToString(Date) + "," + TimeToString(time) + "," + Text(CurrentBar) + ","
    +Text(BarStatus) + "," + Text(jdbl) + "," + Text(xlbl) );

End
```

简单变量和序列变量的生命周期(1)

CurrentTime	Date	Time	CurrentBar	BarStatus	简单变量	序列变量
2014/6/20 11:11	2014/6/20	11:02:00	0	0	0	0
2014/6/20 11:11	2014/6/20	11:02:00	0	0	1	1
2014/6/20 11:11	2014/6/20	11:03:00	1	1	0	1
2014/6/20 11:11	2014/6/20	11:03:00	1	1	1	2
2014/6/20 11:11	2014/6/20	11:04:00	2	1	0	2
2014/6/20 11:11	2014/6/20	11:04:00	2	1	1	3
2014/6/20 11:11	2014/6/20	11:05:00	3	1	0	3
2014/6/20 11:11	2014/6/20	11:05:00	3	1	1	4
2014/6/20 11:11	2014/6/20	11:06:00	4	1	0	4
2014/6/20 11:11	2014/6/20	11:06:00	4	1	1	5
2014/6/20 11:11	2014/6/20	11:07:00	5	1	0	5
2014/6/20 11:11	2014/6/20	11:07:00	5	1	1	6
2014/6/20 11:11	2014/6/20	11:08:00	6	1	0	6
2014/6/20 11:11	2014/6/20	11:08:00	6	1	1	7
2014/6/20 11:11	2014/6/20	11:09:00	7	1	0	7
2014/6/20 11:11	2014/6/20	11:09:00	7	1	1	8
2014/6/20 11:11	2014/6/20	11:10:00	8	1	0	8
2014/6/20 11:11	2014/6/20	11:10:00	8	1	1	9
2014/6/20 11:11	2014/6/20	11:11:00	9	2	0	9
2014/6/20 11:11	2014/6/20	11:11:00	9	2	1	10
2014/6/20 11:11	2014/6/20	11:11:00	9	2	0	9
2014/6/20 11:11	2014/6/20	11:11:00	9	2	1	10
2014/6/20 11:11	2014/6/20	11:11:00	9	2	0	9
2014/6/20 11:11	2014/6/20	11:11:00	9	2	1	10

交易开拓者 TRADEBLAZER

简单变量和序列变量的生命周期(2)

CurrentTime	Date	Time	CurrentBar	BarStatus	简单变量	序列变量
2014/6/20 11:11	2014/6/20	11:11:00	9	2	0	9
2014/6/20 11:11	2014/6/20	11:11:00	9	2	1	10
2014/6/20 11:11	2014/6/20	11:11:00	9	2	0	9
2014/6/20 11:11	2014/6/20	11:11:00	9	2	1	10
2014/6/20 11:11	2014/6/20	11:11:00	9	2	0	9
2014/6/20 11:11	2014/6/20	11:11:00	9	2	1	10
2014/6/20 11:11	2014/6/20	11:11:00	9	2	0	9
2014/6/20 11:11	2014/6/20	11:11:00	9	2	1	10
2014/6/20 11:11	2014/6/20	11:11:00	9	2	0	9
2014/6/20 11:11	2014/6/20	11:11:00	9	2	1	10
2014/6/20 11:11	2014/6/20	11:11:00	9	1	0	9
2014/6/20 11:11	2014/6/20	11:11:00	9	1	1	10
2014/6/20 11:11	2014/6/20	11:12:00	10	2	0	10
2014/6/20 11:11	2014/6/20	11:12:00	10	2	1	11
2014/6/20 11:12	2014/6/20	11:12:00	10	2	0	10
2014/6/20 11:12	2014/6/20	11:12:00	10	2	1	11

全局变量

- 全局变量通过SetGlobalVar和GetGlobalVar函数来设置和读取，单个公式应用可以支持500个全局变量
- 全局变量的初始值为无效值，它的值不会因为当前BAR的变化而变化，而只能由SetGlobalVar函数来设置；
- 全局变量依附在超级图表上，一旦关掉超级图表后，所有与该图表有关的全局变量将不复存在；
- 全局变量值的变化只跟SetGlobalVar的执行顺序有关，因此在图表上进行刷新时，必须考虑因公式重新运行导致的全局变量值的变化。
- 例子：用全局变量记录Tick数（公式见文件）

使用全局变量记录Tick数(1)

Bar时间	CurrentBar	BarStatus	操作	正在记录的Bar时间	简单变量记录结果	序列变量记录结果	全局变量记录结果
2014/6/20 11:25	0	0	计数器初始化	2014/6/20 11:25	1	1	1
2014/6/20 11:26	1	1	新K线产生	2014/6/20 11:26	1	1	1
2014/6/20 11:27	2	1	新K线产生	2014/6/20 11:27	1	1	1
2014/6/20 11:28	3	1	新K线产生	2014/6/20 11:28	1	1	1
2014/6/20 11:29	4	2	新K线产生	2014/6/20 11:29	1	1	1
2014/6/20 11:29	4	1	同一K线记录Tick数	2014/6/20 11:29	1	2	2
2014/6/20 13:00	5	2	新K线产生	2014/6/20 13:00	1	1	1
2014/6/20 13:00	5	2	同一K线记录Tick数	2014/6/20 13:00	1	3	2
2014/6/20 13:00	5	2	同一K线记录Tick数	2014/6/20 13:00	1	3	3
2014/6/20 13:00	5	2	同一K线记录Tick数	2014/6/20 13:00	1	3	4
2014/6/20 13:00	5	2	同一K线记录Tick数	2014/6/20 13:00	1	3	5
2014/6/20 13:00	5	2	同一K线记录Tick数	2014/6/20 13:00	1	3	6
2014/6/20 13:00	5	2	同一K线记录Tick数	2014/6/20 13:00	1	3	7
2014/6/20 13:00	5	2	同一K线记录Tick数	2014/6/20 13:00	1	3	8
2014/6/20 13:00	5	2	同一K线记录Tick数	2014/6/20 13:00	1	3	9

使用全局变量记录Tick数(2)

Bar时间	CurrentBar	BarStatus	操作	正在记录的Bar时间	简单变量记录结果	序列变量记录结果	全局变量记录结果
2014/6/20 13:00	5	2	同一K线记录Tick数	2014/6/20 13:00	1	3	69
2014/6/20 13:00	5	2	同一K线记录Tick数	2014/6/20 13:00	1	3	70
2014/6/20 13:00	5	2	同一K线记录Tick数	2014/6/20 13:00	1	3	71
2014/6/20 13:00	5	2	同一K线记录Tick数	2014/6/20 13:00	1	3	72
2014/6/20 13:00	5	2	同一K线记录Tick数	2014/6/20 13:00	1	3	73
2014/6/20 13:00	5	2	同一K线记录Tick数	2014/6/20 13:00	1	3	74
2014/6/20 13:00	5	2	同一K线记录Tick数	2014/6/20 13:00	1	3	75
2014/6/20 13:00	5	2	同一K线记录Tick数	2014/6/20 13:00	1	3	76
2014/6/20 13:00	5	1	同一K线记录Tick数	2014/6/20 13:00	1	3	77
2014/6/20 13:01	6	2	新K线产生	2014/6/20 13:01	1	1	1
2014/6/20 13:01	6	2	同一K线记录Tick数	2014/6/20 13:01	1	4	2
2014/6/20 13:01	6	2	同一K线记录Tick数	2014/6/20 13:01	1	4	3
2014/6/20 13:01	6	2	同一K线记录Tick数	2014/6/20 13:01	1	4	4
2014/6/20 13:01	6	2	同一K线记录Tick数	2014/6/20 13:01	1	4	5

读写数据库文件

- 全局变量应用简单，速度快。但也有一些使用上的缺陷：个数有限，靠索引识别，不太直观，不能在不同图表之间交换数据，图表关闭后无法保存。
- 因此，如果需要跨图表、跨公式应用之间交换数据，或是需要在关闭软件后还能保存数据，就需要使用读写数据库文件的函数
- 通过这些函数，可以对数据进行有效管理，个数上完全没有限制，还能在不同的图表上进行数据交换。唯一的缺陷就是速度会稍慢。

读写数据库文件函数

- SetTBProfileString 写信息文件

  参数1：String strSection --- 指定的信息块的块名

  参数2：String strKey --- 指定的信息的键名

  参数3：String strValue --- 写入的字符串信息

- GetTBProfileString 读信息文件

  参数1：String strSection

  参数2：String strKey

数据补齐机制

- 升级到V4后，TB取消了无效值传递机制，因此Bar数据来源上我们就需要确保不存在无效值。否则就需要对所有Bar数据进行有效性判断。
- 对不存在的Bar数据需要进行补齐。分为两种方式：
- 1、后补齐；
- 2、前补齐；

数据补齐机制

- 升级到V4后，TB取消了无效值传递机制，因此Bar数据来源上我们就需要确保不存在无效值。否则就需要对所有Bar数据进行有效性判断。
- 对不存在的Bar数据需要进行补齐。分为两种方式：
- 1、后补齐；
- 2、前补齐；

序列函数的调用(1)

- 什么是**序列函数**?
- 函数的参数定义为序列类型，或者函数存在序列变量。这样的函数我们称为序列函数。
- 代码调用序列函数的函数也是序列函数。

- 参数为序列类型的例子:Average。
- 有序列变量的例子:NthCon。
- 代码调用序列函数的例子:AvgTrueRange。

序列函数的调用(2)

- 序列函数应该确保每个Bar都被调用；
- 建议将序列函数的调用放在在正文的外层。不要置于条件表达式或If,For,While等语句内。
- 错误调用示例1：
- If(MyCon && High > CloseD(1))
- …..
- 错误调用示例2:
- If(MyCon)
- {
- MyValue = Average(Close,5);
- }

关于MaxBarsBack

- 公式应用的最大回溯周期，公式自动计算。
- 小于MaxBarsBack的输出值数据都是不准确的。会自动清除掉。

谢谢大家！