海龟交易系统TradeBlazer公式实现

著名的商品投机家理查德.丹尼斯相信，伟大的交易员是后天培养的，他可以教会人们成为伟大的交易员。1983年，他挑选了13个人，对他们进行培训，并为他们提供真实的账户进行交易，希望证明自己的论断。他教给这13个学员一套完整的交易系统，由于学员们被称为“海龟”，后来人们就把这套系统称为海龟交易系统。下面将介绍这套交易系统的具体实现。

由于海龟系统涉及到的变量计算和操作比较多，因此下文把各个变量的计算及一些操作 出来进行说明

关键变量的计算

1) N值的计算

每天开盘第一根k线计算前LengthAtr长度的平均真实波幅;

2) 价值波动 = N\*每点的价值(ContractUnit()\*BigPointValue())

3) 标准头寸计算

根据每天开盘时计算的N和价值波动率计算最新标准头寸.

标准单位头寸(TurtleUnits) = (k线开盘价计算的动态权益\*风险比例)/价值波动率;

 首次入市条件

1) LengthFast(20)长度的唐奇安通道突破;

2) LengthSlow(55)长度的唐奇安通道突破;

备注:前一次平仓是因为止损,随后的开仓需满足条件一入市.如果前一次平仓是一般离市,则随后的开仓需满足条件二入市;

 加仓

1) 与最近一次开仓价格间隔1/2倍N增加仓位;

 止损

1) 最近一次开仓价格反向2N;

 离市

1) LengthExit(10)长度的唐奇安通道被方向突破;

2) LengthAtr(20)长度的唐奇安通道被方向突破;

具体实现如下：

参数

Numeric Direction(0); // 多空信号过滤条件,值为0不过滤,值为1只做多,值为1只做空;

Numeric RiskRatio(1); // % Risk Per N ( 0 - 100)

Numeric LengthAtr(20); // 平均波动周期 ATR Length

Numeric LengthFast(20); // 短周期 BreakOut Length

Numeric LengthSlow(55); // 长周期 FailSafe Length

Numeric LengthExit(10); // 离市周期 Trailing Exit Length

Bool LastProfitableTradeFilter(True); // 使用入市过滤条件

源代码：

//------------------------------------------------------------------------

// 简称: Turtle

// 名称: 海龟交易系统

// 类别: 公式应用

// 类型: 用户应用

// 输出:

//------------------------------------------------------------------------

//------------------------------------------------------------------------

// 简称: ITF\_TurtleRules\_V102

// 名称: TurtleRules\_V102

// 类别: 公式应用

// 类型: 用户应用

// 输出:

//------------------------------------------------------------------------

Params

Numeric RiskRatio(1); // % Risk Per N ( 0 - 100)

Numeric LengthAtr(20); // 平均波动周期 ATR Length

Numeric LengthFast(20); // 短周期 BreakOut Length

Numeric LengthSlow(55); // 长周期 FailSafe Length

Numeric LengthExit(10); // 离市周期 Trailing Exit Length

Bool LastProfitableTradeFilter(True); // 使用入市过滤条件

Vars

Numeric MinPoint; // 最小变动单位

NumericSeries N; // N 值

NumericSeries TotalEquity; // 按最新收盘价计算出的总资产

NumericSeries TurtleUnits; // 交易单位

NumericSeries DonchianFastUpper; // 短周期唐奇安通道上轨,长度LengthFast;

NumericSeries DonchianFastLower; // 短周期唐奇安通道下轨,长度LengthFast;

NumericSeries DonchianSlowUpper; // 长周期唐奇安通道上轨,长度LengthSlow;

NumericSeries DonchianSlowLower; // 长周期唐奇安通道下轨,长度LengthSlow;

NumericSeries DonchianExitUpper; // 离市时判断需要的N周期最高价,长度LengthExit;

NumericSeries DonchianExitLower; // 离市时判断需要的N周期最低价,长度LengthExit;

NumericSeries MyPrice; // 开仓价格

NumericSeries MyExitPrice; // 平仓价格

BoolSeries PreBreakoutFailure(false); // 前一次突破是否失败

Bool SendOrderThisBar(False); // 当前Bar有过交易

NumericSeries AtrAve; // 记录日线级别长度为lengthAtr的平均真实波幅;

NumericSeries DayHigh; // 一天中的最高价;

NumericSeries DayLow; // 一天中的最低价;

Numeric i; // 循环计算所需指针;

NumericSeries Atr(0); // 当天的ATR

Numeric AtrIndex(0); // 存储真实波幅的全局变量的环的索引值;

NumericSeries DayCount(0); // 交易日数;

NumericSeries CloseD\_1; // 上一个交易日收盘价,过度CloseD(1);

NumericSeries MyBarsSinceToday; // 过度BarsSinceToday;

Begin

MinPoint = MinMove\*PriceScale;

MyBarsSinceToday = BarsSinceToday;

CloseD\_1 = CloseD(1);

if(BarsSinceToday==0)

{

DayCount = DayCount + 1; //记录交易日数;

DayHigh = High;

DayLow = Low;

//从第二个交易日开始记录;

if(DayCount>=2)

{

// 前一日真实波幅;

Atr = max(max(Abs(DayHigh[1]-CloseD\_1),Abs(DayLow[1]-CloseD\_1)),DayHigh[1]-DayLow[1]);

// 存储前一天的Atr到全局变量环0-19;

AtrIndex = (DayCount-2)%LengthAtr;

SetGlobalVar(AtrIndex,Atr[1]);

}

// 从第LengthAtr+1个交易日开始计算;

if(DayCount>=LengthAtr+1)

{

// 计算日线级别长度为LengthAtr的平均真实波幅;

AtrAve = 0 ;

for i=0 to LengthAtr-1

{

AtrAve = AtrAve + getGlobalVar(i);

}

AtrAve = AtrAve/LengthAtr;

N = IntPart(AtrAve);

DonchianFastUpper = Highest(DayHigh[1],(LengthAtr-0.1)\*(MyBarsSinceToday[1]+1));

DonchianFastLower = Lowest(DayLow[1],(LengthAtr-0.1)\*(MyBarsSinceToday[1]+1));

DonchianSlowUpper = Highest(DayHigh[1],(LengthSlow-0.1)\*(MyBarsSinceToday[1]+1));

DonchianSlowLower = Lowest(DayLow[1],(LengthSlow-0.1)\*(MyBarsSinceToday[1]+1));

DonchianExitUpper = Highest(DayHigh[1],(LengthExit-0.1)\*(MyBarsSinceToday[1]));

DonchianExitLower = Lowest(DayLow[1],(LengthExit-0.1)\*(MyBarsSinceToday[1]));

}

//账户最新资产 = 按当前Bar开盘价计算的可用资金 + 持仓保证金 ;

TotalEquity = Portfolio\_CurrentCapital() + Portfolio\_UsedMargin();

//标准单位头寸 = (本金\*风险比例)/价值波动率

TurtleUnits = (TotalEquity\*RiskRatio/100) /(N \* ContractUnit()\*BigPointValue());

TurtleUnits = IntPart(TurtleUnits); // 对小数取整

// PlotString("Units","Units="+Text(TurtleUnits),High+(High-Low),white);

// PlotString("N","N="+Text(N),Low-(High-Low)/2,white);

}

Else

{

DayHigh = Max(DayHigh,High);

DayLow = Min(DayLow,Low);

}

If(MarketPosition==0 and TurtleUnits>=1)

{

// 首次入市1:带过滤条件的,如果过滤条件为真,则需要前一笔交易是否是突破失败;

if(LastProfitableTradeFilter==false Or PreBreakoutFailure )

{

//开多1:突破短周期唐奇安通道上轨;

If(High>DonchianFastUpper)

{

MyPrice = Max(Open,DonchianFastUpper)+MinPoint;

Buy(TurtleUnits,MyPrice);

MyExitPrice = MyPrice - 2\*N;

SendOrderThisBar = true;

}

//开空1:突破短周期唐奇安通道下轨;

if(Low<DonchianFastLower)

{

MyPrice = Min(Open,DonchianFastLower)-MinPoint;

SellShort(TurtleUnits,MyPrice);

MyExitPrice = MyPrice + 2\*N;

SendOrderThisBar = true;

}

}

// 首次入市2:过滤条件之外的,长周期唐奇安通道突破开仓;

//开多2:突破长周期唐奇安通道上轨;

else

{

if(CrossOver(High,DonchianSlowUpper))

{

MyPrice = Max(Open,DonchianSlowUpper)+MinPoint;

Buy(TurtleUnits,MyPrice);

MyExitPrice = MyPrice - 2\*N;

SendOrderThisBar = true;

}

//开空2:突破长周期唐奇安通道下轨;

if(CrossUnder(Low,DonchianSlowLower))

{

MyPrice = Min(Open,DonchianSlowLower)+MinPoint;

SellShort(TurtleUnits,MyPrice);

MyExitPrice = MyPrice + 2\*N;

SendOrderThisBar = true;

}

}

}

// 当持有多单;

if(MarketPosition==1)

{

//离市:价格反向穿越离市的唐奇安通道;

if(CrossUnder(Low,DonchianExitLower))

{

MyPrice = Min(Open,DonchianExitLower)-MinPoint;

Sell(0,MyPrice);

PreBreakoutFailure = False; //假突破值为假;

// PlotBool("突破",PreBreakoutFailure,High,white);

}

Else

{

//加仓:与最近一次开仓价间隔0.5\*N开仓;

if(High>LastEntryPrice+0.5\*N and TurtleUnits>=1)

{

MyPrice = Max(Open,LastEntryPrice+0.5\*N)-MinPoint;

Buy(TurtleUnits,MyPrice);

SendOrderThisBar = true;

}

//止损:价格反向穿越最近一次开仓价格-2N;

if(Low<LastEntryPrice-2\*N)

{

MyPrice = Min(Open,LastEntryPrice-2\*N)-MinPoint;

Sell(0,MyPrice);

PreBreakoutFailure = True; //假突破值为真;

// PlotBool("突破",PreBreakoutFailure,High,white);

}

}

}

// 当持有空单;

else if(MarketPosition==-1)

{

// 离市:价格反向穿越离市的唐奇安通道;

if(CrossOver(High,DonchianExitUpper))

{

MyPrice = Max(Open,DonchianExitUpper)+MinPoint;

BuyToCover(0,MyPrice);

PreBreakoutFailure = False; //假突破值为假;

// PlotBool("突破",PreBreakoutFailure,High,white);

}

Else

{

// 加仓:与最近一次开仓价格间隔0.5\*N开仓;

if(Low<LastEntryPrice-0.5\*N and TurtleUnits>=1)

{

MyPrice = Min(Open,LastEntryPrice-0.5\*N)-MinPoint;

SellShort(TurtleUnits,MyPrice);

SendOrderThisBar = true;

}

// 止损:价格反向穿越最近一次开仓价格+2N;

if(High>LastEntryPrice+2\*N)

{

MyPrice = Max(Open,LastEntryPrice+2\*N)+MinPoint;

BuyToCover(0,MyPrice);

PreBreakoutFailure = True; //假突破值为真;

// PlotBool("突破",PreBreakoutFailure,High,white);

}

}

}

PlotNumeric("DonchianFastUpper",DonchianFastUpper);

PlotNumeric("DonchianFastLower",DonchianFastLower);

PlotNumeric("DonchianSlowUpper",DonchianSlowUpper);

PlotNumeric("DonchianSlowLower",DonchianSlowLower);

PlotNumeric("DonchianExitUpper",DonchianExitUpper);

PlotNumeric("DonchianExitLower",DonchianExitLower);

// Commentary("点值="+Text(ContractUnit()\*BigPointValue()));

Commentary("TotalEquity="+Text(TotalEquity));

Commentary("N="+Text(N));

Commentary("Units="+Text(TurtleUnits));

// Commentary("Atr="+Text(Atr));

// Commentary("AtrAve="+Text(AtrAve));

// PlotNumeric("AtrAve",AtrAve);

Commentary("DayCount="+Text(DayCount));

Commentary("MyBarsSinceToday="+Text(MyBarsSinceToday[1]+1));

if(PreBreakoutFailure == True)

{

Commentary("PreBreakoutFailure = True");

}Else

{

Commentary("PreBreakoutFailure = false");

}

if(LastProfitableTradeFilter==true)

{

Commentary("LastProfitableTradeFilter = True");

}Else

{

Commentary("LastProfitableTradeFilter = False");

}

End