(http://custom.shinnytech.com/)

(https://www.shinnytech.com)

海龟交易法则（难度：中级）

什么是海龟交易法则

海龟交易法是著名的公开交易系统，其法则覆盖了交易的各个方面，并且不给交易员留下一点主观想象决策的余地。它是一套非常完整的趋势跟随型的自动化交易策略，具备一个完整的交易系统的所有成分。.这个复杂的策略在入场条件、仓位控制、资金管理、止损止盈等各个环节，都进行了详细的设计，它基本上可以作为复杂交易策略设计和开发的模板。对于币市的大涨大跌行情，海龟交易法则正是应付这种极端行情的利器。

海龟交易法则的战绩

海龟交易的创始人是七八十年代著名的期货投机商Richard Dennis，他相信优秀的交易员是后天培养而非天生的。他在1983年12月招聘了23名新人，昵称为海龟，并对这些交易员进行了一个简单的趋势跟踪交易策略培训。随后给予每个新人100万美元的初始资金。经过5年的运作，大部分“海龟”的业绩非常惊人，其中最好的业绩达到1.72亿美元。N年后海龟交易法则公布于世，我们才有幸看到曾名噪一时的海龟交易法则全貌。

策略的实现方法

趋势信号的捕捉

在趋势信号的扑捉上，海龟交易法则使用了一个非常重要的技术指标——唐奇安通道（Donchian channel）。这个通道很类似布林通道（Bollinger Bands），只是在具体计算方式上有些不一样。 唐奇安通道指标是Richard Donchian发明的，由3条不同颜色的曲线组成，该指标用周期（一般都是20，可以进行修改）内的最高价和最低价来显示市场价格的波动性，当其通道窄时表示市场波动较小，反之通道宽则表示市场波动比较大。

当价格冲破该通道的上轨道时，就是可能的买信号；反之，冲破下轨时就是可能的卖信号。唐奇安通道的各项指标的计算方法为：

上轨 = Max（最高价，n）， n日最高价的最大值

下轨 = Min（最低价，n），n日最低价的最小值

中轨 = (上轨 + 下轨) / 2

仓位的基本单位Unit

海龟法则的加仓原则是定义好一个小单位（Unit），使得该仓位的预期价值波动与总净资产的1%对应。也就是说，如果买入了这1个小单位的资产，那当天该仓位的市值变动幅度不会超过总净资产的1%。

那么，如何定义这个小单位？又如何预估这个小单位能带来的价值波动呢？首先，在预估这个小单位带来的价值波动（该价值波动被称为N）上，海龟策略使用了对历史的价格波动进行统计的方法。具体计算公式如下：

TrueRange = Max(High − Low, High − PreClose, PreClose − Low)

N = (前19日的N值之和 + 当时的TrueRange) / 20

其中，High表示当日最高价，Low表示当日最低价，PreClose表示前一日收盘价。
我们可以从定义上看出，N值确实能很恰当地表达该资产在价格上的最近波动幅度。
这样，一个Unit就应该是这样计算出来的：

Unit = (1% * Total_net) / N

其中total_net就是总资产净值。
可以看出，一个Unit的资产的价格波动幅度 = 总净资产的1%。

建仓

建仓的动作来自于趋势突破信号的产生。如果当前价格冲破唐奇安通道上轨，就产生了一个买的建仓信号，如果当前价格跌破下轨，就产生了一个卖空的建仓信号。初始建仓的大小为1个Unit。

加仓

如果开的底仓是多仓，且行情最新价在上一次建仓（或者加仓）的基础上又上涨了0.5N，就再加一个Unit的多仓。

如果开的底仓是空仓，且行情最新价在上一次建仓（或者加仓）的基础上又下跌了0.5N，就再加一个Unit的空仓。

我们看到，海龟策略其实是一个追涨杀跌的策略的。

止损

如果开的底仓是多仓，且行情最新价在上一次建仓（或者加仓）的基础上又下跌了2N，就卖出全部头寸，平仓止损。

如果开的底仓是空仓，且行情最新价在上一次建仓（或者加仓）的基础上又上涨了2N，就买入全部头寸，平仓止损。

当然，也可以自定义止损方案，使止损策略更符合所选的合约、适应自定义的个性化策略优化方案。

止盈

如果开的底仓是多仓，且行情最新价跌破了10日唐奇安通道的下轨，就清空所有头寸结束策略。
如果开的底仓是空仓，且行情最新价升破了10日唐奇安通道的上轨，就清空所有头寸结束策略。
当然，可以自定义动态止盈方案。

代码实现

工具：
免费期货模拟、实盘天勤量化程序：https://www.shinnytech.com/tianqin/ (https://www.shinnytech.com/tianqin/)

Python 金融指数处理库Ta-Lib

Talib文档：https://mrjbq7.github.io/ta-lib/doc_index.html (https://mrjbq7.github.io/ta-lib/doc_index.html)

指标计算

获取K线序列，根据定义计算出唐奇安通道上下轨。（策略源码在文章最后）

```
# 唐奇安通道的天数周期(开仓)
donchian_channel_open_position= 20
# 获取klines数据
klines = api.get_kline_serial("", 24 * 60 * 60, data_length=100)
# 唐奇安通道上轨：前N个交易日的最高价
donchian_channel_high = max(klines.high[donchian_channel_open_position - 1:-1])
# 唐奇安通道下轨：前N个交易日的最低价
```

```
8.  donchian_channel_low = min(klines.low[donchian_channel_open_position - 1:-1])
```

把K线转为pandas.DataFrame便于计算；使用talib库的ATR函数计算出N值（即平均真实波幅），然后根据账户权益的1%计算买卖单位unit。

```
1.  # 平均真实波幅(N值即为ATR值)
2.  n = ATR(klines, atr_day_length)["atr"].iloc[-1]
3.  # 计算一个ATR波幅的买卖单位
4.  unit = int((account.balance * 0.01) / (quote.volume_multiple * n))
```

建仓

在净持仓数为0时：获取最新行情价，如果当前最新价大于唐奇安通道的上轨，则买入一个unit（此时持多仓）；如果当前最新价小于唐奇安通道的下轨，则卖出一个unit（此时持空仓）。

```
1.  # 当前价>唐奇安通道上轨，买入1个Unit；(持多仓)
2.  if quote.last_price > donchian_channel_high:
3.      print("当前价>唐奇安通道上轨，买入1个Unit(持多仓)：%d 手" % unit)
4.      set_position(state["position"] + unit)
5.  elif quote.last_price < donchian_channel_low:  # 当前价<唐奇安通道下轨，卖出1个Unit；(持空仓)
6.      print("当前价<唐奇安通道下轨，卖出1个Unit(持空仓)：%d 手" % unit)
7.      set_position(state["position"] - unit)
```

加仓、止损、止盈

在净持仓数不为0时：判断是持多仓还是空仓，获取最新行情价，根据加仓、止损、止盈策略的条件进行相应的仓位操作。

```
1.    if state["position"] > 0:  # 持多单
2.      # 加仓策略：如果是多仓且行情最新价在上一次建仓（或者加仓）的基础上又上涨了0.5N，就再加一个Unit的多仓，并且风险度在设定范围内(以防爆仓)
3.      if quote.last_price >= state["last_price"] + 0.5 * n and account.risk_ratio <= max_risk_ratio:
4.          print("加仓:加1个Unit的多仓")
5.          set_position(state["position"] + unit)
6.      # 止损策略：如果是多仓且行情最新价在上一次建仓（或者加仓）的基础上又下跌了2N，就卖出全部头寸止损
7.      elif quote.last_price <= state["last_price"] - 2 * n:
8.          print("止损:卖出全部头寸")
9.          set_position(0)
10.     # 止盈策略：如果是多仓且行情最新价跌破了10日唐奇安通道的下轨，就清空所有头寸结束策略,离场
11.     if quote.last_price <= min(klines.low[-donchian_channel_stop_profit - 1:-1]):
12.         print("止盈:清空所有头寸结束策略,离场")
13.         set_position(0)
14.
15.   elif state["position"] < 0:  # 持空单
16.     # 加仓策略：如果是空仓且行情最新价在上一次建仓（或者加仓）的基础上又下跌了0.5N，就再加一个Unit的空仓，并且风险度在设定范围内(以防爆仓)
17.     if quote.last_price <= state["last_price"] - 0.5 * n and account.risk_ratio <= max_risk_ratio:
18.         print("加仓:加1个Unit的空仓")
19.         set_position(state["position"] - unit)
20.     # 止损策略：如果是空仓且行情最新价在上一次建仓（或者加仓）的基础上又上涨了2N，就平仓止损
21.     elif quote.last_price >= state["last_price"] + 2 * n:
22.         print("止损:卖出全部头寸")
23.         set_position(0)
```

```
    # 止盈策略：如果是空仓且行情最新价升破了10日唐奇安通道的上轨，就清空所有头寸结束策略,离场
    if quote.last_price >= max(klines.high[-donchian_channel_stop_profit - 1:-1]):
        print("止盈:清空所有头寸结束策略,离场")
        set_position(0)
```

回测

回测初始参数设置

初始账户资金：1000万

回测日期：2018.9.10 —— 2018.11.12

唐其安通道开仓天数周期：20

唐其安通道止盈天数周期：10

计算ATR所用天数：20

允许下单的最高风险度：50%

回测时盘口行情quote的更新频率：和K线分钟线的更新频率一致

回测结果

海龟策略回测结果					
合约代码	合约品种	收益率	风险度	最大回撤	年化夏普率
SHFE.hc1901	热卷	22.12%	44.64%	14.96%	2.9268

上表回测合约的累积收益走势图

天勤量化中策略源码：

```
#!/usr/bin/env python
#  -*- coding: utf-8 -*-
__author__ = 'limin'

'''
海龟策略
参考：https://www.shinnytech.com/blog/turtle/
注：该示例策略仅用于功能示范，实盘时请根据自己的策略/经验进行修改
'''

import json
import time
from tqsdk import TqApi, TargetPosTask
from tqsdk.ta import ATR


class Turtle:
    def __init__(self, symbol, account=None, donchian_channel_open_position=20, donchian_channel_stop_profit=10, atr_day_length=20, max_risk_ratio=0.5):
        self.account = account  # 交易账号
        self.symbol = symbol  # 合约代码
        self.donchian_channel_open_position = donchian_channel_open_position  # 唐奇安通道的天数周期(开仓)
        self.donchian_channel_stop_profit = donchian_channel_stop_profit  # 唐奇安通道的天数周期(止盈)
        self.atr_day_length = atr_day_length  # ATR计算所用天数
        self.max_risk_ratio = max_risk_ratio  # 最高风险度
        self.state = {
            "position": 0,  # 本策略净持仓数(正数表示多头，负数表示空头，0表示空仓)
            "last_price": float("nan"),  # 上次调仓价
        }

        self.n = 0  # 平均真实波幅(N值)
        self.unit = 0  # 买卖单位
        self.donchian_channel_high = 0  # 唐奇安通道上轨
        self.donchian_channel_low = 0  # 唐奇安通道下轨

        self.api = TqApi(self.account)
        self.quote = self.api.get_quote(self.symbol)
        # 由于ATR是路径依赖函数，因此使用更长的数据序列进行计算以便使其值稳定下来
        kline_length = max(donchian_channel_open_position + 1, donchian_channel_stop_profit + 1, atr_day_length * 5)
        self.klines = self.api.get_kline_serial(self.symbol, 24 * 60 * 60, data_length=kline_length)
        self.account = self.api.get_account()
        self.target_pos = TargetPosTask(self.api, self.symbol)

    def recalc_paramter(self):
        # 平均真实波幅(N值)
        self.n = ATR(self.klines, self.atr_day_length)["atr"].iloc[-1]
        # 买卖单位
        self.unit = int((self.account.balance * 0.01) / (self.quote.volume_multiple * self.n))
        # 唐奇安通道上轨：前N个交易日的最高价
        self.donchian_channel_high = max(self.klines.high[-self.donchian_channel_open_position - 1:-1])
        # 唐奇安通道下轨：前N个交易日的最低价
        self.donchian_channel_low = min(self.klines.low[-
```

```
            self.donchian_channel_low = min(self.klines.low[-self.donchian_channel_open_position - 1:-1])
            print("唐其安通道上下轨: %f, %f" % (self.donchian_channel_high, self.donchian_channel_low))
            return True

        def set_position(self, pos):
            self.state["position"] = pos
            self.state["last_price"] = self.quote["last_price"]
            self.target_pos.set_target_volume(self.state["position"])

        def try_open(self):
            """开仓策略"""
            while self.state["position"] == 0:
                self.api.wait_update()
                if self.api.is_changing(self.klines.iloc[-1], "datetime"):  # 如果产生新k线,则重新计算唐奇安通道及买卖单位
                    self.recalc_paramter()
                if self.api.is_changing(self.quote, "last_price"):
                    print("最新价: %f" % self.quote.last_price)
                    if self.quote.last_price > self.donchian_channel_high:  # 当前价>唐奇安通道上轨，买入1个Unit；(持多仓)
                        print("当前价>唐奇安通道上轨，买入1个Unit(持多仓): %d 手" % self.unit)
                        self.set_position(self.state["position"] + self.unit)
                    elif self.quote.last_price < self.donchian_channel_low:  # 当前价<唐奇安通道下轨，卖出1个Unit；(持空仓)
                        print("当前价<唐奇安通道下轨，卖出1个Unit(持空仓): %d 手" % self.unit)
                        self.set_position(self.state["position"] - self.unit)

        def try_close(self):
            """交易策略"""
            while self.state["position"] != 0:
                self.api.wait_update()
                if self.api.is_changing(self.quote, "last_price"):
                    print("最新价: ", self.quote.last_price)
                    if self.state["position"] > 0:  # 持多单
                        # 加仓策略：如果是多仓且行情最新价在上一次建仓（或者加仓）的基础上又上涨了0.5N，就再加一个Unit的多仓,并且风险度在设定范围内(以防爆仓)
                        if self.quote.last_price >= self.state["last_price"] + 0.5 * self.n and self.account.risk_ratio <= self.max_risk_ratio:
                            print("加仓:加1个Unit的多仓")
                            self.set_position(self.state["position"] + self.unit)
                        # 止损策略：如果是多仓且行情最新价在上一次建仓（或者加仓）的基础上又下跌了2N，就卖出全部头寸止损
                        elif self.quote.last_price <= self.state["last_price"] - 2 * self.n:
                            print("止损:卖出全部头寸")
                            self.set_position(0)
                        # 止盈策略：如果是多仓且行情最新价跌破了10日唐奇安通道的下轨，就清空所有头寸结束策略,离场
                        if self.quote.last_price <= min(self.klines.low[-self.donchian_channel_stop_profit - 1:-1]):
                            print("止盈:清空所有头寸结束策略,离场")
                            self.set_position(0)

                    elif self.state["position"] < 0:  # 持空单
                        # 加仓策略：如果是空仓且行情最新价在上一次建仓（或者加仓）的基础上又下跌了0.5N，就再加一个Unit的空仓,并且风险度在设定范围内(以防爆仓)
                        if self.quote.last_price <= self.state["last_price"] - 0.5 * self.n and self.account.risk_ratio <= self.max_risk_ratio:
                            print("加仓:加1个Unit的空仓")
                            self.set_position(self.state["position"] - self.unit)
                        # 止损策略：如果是空仓且行情最新价在上一次建仓（或者加仓）的基础上又上涨了2N，就平仓
```

```
                        # 止损策略：如果是空仓且行情最新价在上一次建仓（或者加仓）时的基础上又上涨了2N，就平仓止损
                        elif self.quote.last_price >= self.state["last_price"] + 2 * self.n:
                            print("止损:卖出全部头寸")
                            self.set_position(0)
                        # 止盈策略：如果是空仓且行情最新价升破了10日唐奇安通道的上轨，就清空所有头寸结束策略,离场
                        if self.quote.last_price >= max(self.klines.high[-self.donchian_channel_stop_profit - 1:-1]):
                            print("止盈:清空所有头寸结束策略,离场")
                            self.set_position(0)

    def strategy(self):
        """海龟策略"""
        print("等待K线及账户数据...")
        deadline = time.time() + 5
        while not self.recalc_paramter():
            if not self.api.wait_update(deadline=deadline):
                raise Exception("获取数据失败，请确认行情连接正常并已经登录交易账户")
        while True:
            self.try_open()
            self.try_close()


turtle = Turtle("SHFE.hc1901")
print("策略开始运行")
try:
    turtle.state = json.load(open("turtle_state.json", "r"))  # 读取数据：本策略目标净持仓数,上一次开仓价
except FileNotFoundError:
    pass
print("当前持仓数：%d，上次调仓价：%f" % (turtle.state["position"], turtle.state["last_price"]))
try:
    turtle.strategy()
finally:
    turtle.api.close()
    json.dump(turtle.state, open("turtle_state.json", "w"))  # 保存数据
```

点击查看天勤量化（tqsdk） (https://www.shinnytech.com/tianqin/)

策略参考：https://www.douban.com/group/topic/104900172/ (https://www.douban.com/group/topic/104900172/)

(https://www.shinnytech.com/blog/r-breaker/) (https://www.shinnytech.com/blog/random_fo