* **系統目的**

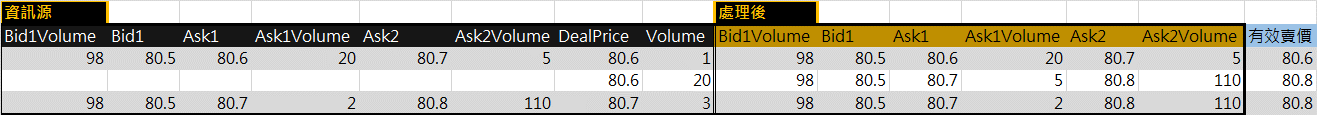
根據現貨掛期貨委託單的成交方式來做期現貨套利，期貨掛單價格會根據現貨有效價格

* **S相關**

|  |  |
| --- | --- |
| **S** | **定義** |
| Valid Lots | 有效張數，此值By各Pid設定 |
| 現貨有效買價 | 從現股Bid1價格的委託量起算，往後累計加總張數> Valid Lots的價格 |
| 現貨有效賣價 | 從現股Ask1價格的委託量起算，往後累計加總張數> Valid Lots的價格 |

* 現貨穿價時，有效價量取代方式：

1. 當成交價>前一次行情Ask1 -> 用成交價取代Ask1，委託量Ask1Volume = 前一次行情值 - 同價的成交量
2. 當成交價<前一次行情Bid1 -> 用成交價取代Bid1，委託量Bid1Volume = 前一次行情值 - 同價的成交量
3. 以上邏輯後要再做ValidLots的判斷，才是此時的現貨有效買價/賣價。

[ 範例 ]：Valid Lots = 10

* **下單及開關**

1. 期貨下單

* 下單口數：期貨買賣下單設定口數By各Pid設定FLots
  + 期貨賣單下單口數 = Flots
  + 期貨買單下單口數 =   
     若股票賣出選下現股：期貨買單下單口數 = min( FLots , 現貨有效買價之買單量/個股期貨契約乘數 )

若股票賣出選下借券：期貨買單下單口數 = min( FLots , 借券張數/個股期貨契約乘數, 借券可賣餘額/個股期貨契約乘數)  
\*以上期貨下單口數皆無條件捨去致整數位

* + 期貨買單下出時就將現股/借券庫存預扣住，成交時再釋放做下單
  + Flots防呆：Flots不能超過validLots/個股期貨契約乘數，超過不予修改，DB載入時若不符合此規則，一律改成Flots=1
* 期貨下單Account、現貨下單Account&TID 於DB中可設定多組，下哪個帳號By Pid設定
* 單種皆為ROD
* 期貨最後判斷的出價若期貨買單出價高於期貨漲停價則下期貨漲停價，期貨賣單出價低於跌停價則下期貨跌停價；若期貨賣單高於期貨漲停價則暫不下單，期貨買單低於期貨跌停價則暫不下單。

1. 現貨下單
2. 下單價格：期貨下單時的現貨有效價格，期貨一成交即以現貨有效價格下出委託。
3. 下單張數：期貨成交口數\*個股期貨契約乘數
4. 單種皆為ROD
5. 以下情況Pid開關關閉，並訊息揭示
   1. 處置股票 (T30裡mark\_w欄位 <> 0)
   2. 期貨超過漲跌停價
6. 以下情況該Pid期貨暫不下單：
7. 現貨尚未開盤
8. 現貨暫緩搓合
9. 現貨尚未收到成交價
10. 漲幅>S%暫時不下期貨賣單
11. 漲幅<-S%暫時不下期貨買單
12. 現貨下出後+N秒寬限時間後，期貨+現貨Delta不為0時暫不再下單，直到Delta回復為0時再開始下單
13. 若有人掛單在內層以內(買賣開關亮黃燈提示)
14. 買/賣邊開關Off：
    1. 選下現貨或借券庫存皆為0時→關閉期貨買開關 (訊息提示)
    2. 期貨成交NLots達到Match口數 (訊息提示)

* **基本設定**

1. 上市櫃股票及期貨行情：KMD和元大資訊源互為備援
2. Broadcast連線：NB52：暫停交易 NB54：證交行情延遲
3. 買內層寬度B\_In%需<買外層寬度B\_Out%、賣內層寬度S\_In%需>賣外層寬度S\_Out%；否則不予修改

* **參數、防呆及畫面**

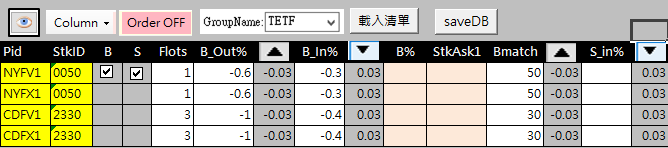
全域參數：

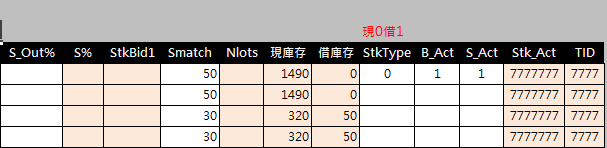
|  |  |
| --- | --- |
| 每筆最大下單口數 | FLots過此數字則以此數字替代 |
| SpotStop秒 | 經過spotstop秒都沒有新的行情進來，則開關關閉 |
| SpotJump(%) | 買賣共用同一個SpotJump，以前一筆買(賣)價對新買(買)的價格變動來判斷是否碰觸spotjump，例如：abs((Bidt/ Bidt-1)-1)> SpotJump，則買單開關關閉，訊息揭露 |
| S%Foolproof | 上下界防呆。期貨買價不得高於現貨成交價\*(1+ S%Foolproof)，期貨賣價不得低於現貨成交價\*(1- S%Foolproof) |
| S% | 現股超過此漲跌幅設定則對應方向之期貨開關關閉 (漲幅>S%暫時不下期貨賣單，漲幅<-S%暫時不下期貨買單) |
| Forward競價(進)毫秒 | 須改價往前競價時，等Forward後改價重下 |
| Backward競價(退)毫秒 | 須改價往後時，等Backward後改價重下 |
| N搶單秒 | 期貨有在內層以外成交，搶單下單價格為成交價，停留N秒後取消，下單口數為Flots |
| SwitchOffT | 總下單開關自動關閉時間 |
| ShiftUnit | 每次點調整 內層寬度(B\_In%、S\_in%)的移動單位，於ini中設定 |
| N秒Delta寬限時間 | 現貨下出後+N秒寬限時間後，期貨+現貨Delta不為0時暫不再下單，直到Delta回復為0時再開始下單 |
| OrderStartT | 開始下單時間，總開關可以提早啟動，但時間到才開始下，但是還是須等到該現貨商品開盤後才能下 |
| SwitchOffT | 總下單開關自動關閉時間 |

* **UI介面：**
* 主控畫面分做上方功能列和下方2種不同欄位屬性，上方功能包括：

1. 精簡模式/全模式：按下精簡則僅顯示有勾選之欄位
2. ：勾選之欄位為精簡模式下會顯示之欄位
3. ：總下單開關
4.  / 載入清單按鈕：選擇要載入的清單名稱後按下Load載入，若在總下單開關開啟的情況下按載入清單，則先將總下單開關OFF
5. SaveDB：將參數設定欄位和買賣開關回寫DB，AP關閉時也自動執行

* 下方欄位分做兩個部分，左邊為開關By Group&ValidLots設定，右邊為Pid清單：





商品參數：

|  |  |
| --- | --- |
| Pid | 使用者於DB建立期貨商品清單 |
| StkID | 使用者於DB建立標的股票代碼 |
| B | 期貨買開關 |
| S | 期貨賣開關 |
| FLot | 期貨下單口數 |
| Valid | 現貨有效口數(Valid Lots) |
| B\_Out% | 期貨買外層寬度 |
|  | B\_In% 向上移動ShiftUnit，此Button上揭示ini設定之ShiftUnit |
| B\_In% | 此值設定期貨買內層寬度 |
| Adj\_B\_In% | 經AI進退計算後之期貨買內層寬度 |
|  | B\_In% 向下移動ShiftUnit，此Button上揭示ini設定之ShiftUnit |
| B% | B% = (( 買價\_F / 買價\_S ) -1 )/100 ->即實際買期貨之Basis幅度 |
| StkAsk1 | 現股有效賣價 |
| S\_In% | 此值設定期貨賣內層寬度 |
| Adj\_S\_In% | 經AI進退計算後之期貨賣內層寬度 |
|  | S\_In% 向上移動ShiftUnit，此Button上揭示ini設定之ShiftUnit |
| S\_Out% | 期貨賣外層寬度 |
|  | S\_In% 向下移動ShiftUnit，此Button上揭示ini設定之ShiftUnit |
| S% | S% = (( 賣價\_F / 賣價\_S ) -1 )/100 ->即實際賣期貨單之Basis幅度 |
| StkBid1 | 現股有效買價 |
| AI\_S% | 進退參數，每買進/賣出n \* AI口口數的期貨，買內層減/增 n \* AI\_S% (即Adj\_B\_In%)、賣內層增/減 n \* AI\_S% (即Adj\_S\_In%) |
| AI口 |
| StkType | 0：股票賣出選下現股；1：股票賣出選下借券 |
| Bmatch | 期貨買成交match口數 |
| Smatch | 期貨賣成交match口數 |
| NLots | 現貨淨買賣張數(現股+借券) |
|  | Reset NLots，若按Reset時NLots不為0則同步關閉該Pid開關 |
| 現庫存 | 現貨庫存張數 (為Stk\_Act現貨下單帳號之下的庫存張數) |
| 借庫存 | 借券庫存張數 (為Stk\_Act現貨下單帳號之下的庫存張數) |
| B\_Act | 期貨買下單帳號 |
| S\_Act | 期貨賣下單帳號 |
| Stk\_Act | 現貨下單帳號 |
| TID | 現貨下單TID |

* 庫存計算

1. 現貨
2. 可當沖：整股現貨庫存=昨日庫存-控管股數+整股買進-整股賣出-零股賣出
3. 不可當沖：整股現貨庫存=昨日庫存-控管股數-整股賣出-零股賣出

可當沖股票可賣庫存=昨日庫存-控管股數-今日賣出+今日買進-在途賣出

不可當沖股票可賣庫存=昨日庫存-控管股數-今日賣出-在途賣出

* **市價單處理**

當出現市價單時(資訊源價格為0但有委託量)

1. Bid1出現市價單：則認定Bid1為漲停價
2. Ask1出現市價單：則認定Ask1為漲停價

* **訊息視窗**

1. 競價行為訊息視窗
2. 開關關閉訊息視窗

* **內外層價格計算**

1. 期貨內層買單價格：tickunit(現貨有效買價\*(1- Adj\_B\_In %)，向下取)
2. 期貨內層賣單價格：tickunit(現貨有效賣價\*(1+Adj\_S\_In%，向上取)
3. 期貨外層買單價格：tickunit(現貨有效買價\*(1-B\_Out %)，向下取)
4. 期貨外層賣單價格：tickunit(現貨有效賣價\*(1+S\_Out %)，向上取)
5. 現貨有效價格改變時重算內外層價

* **競價邏輯**

1. **起始下單**
   1. 若無人掛單，則掛在外層。
   2. 若有人掛單在內層外(不含內層)&外層內(含外層)，則掛在前一個tick，但不得窄於內層。
   3. 若有人掛單在內層以內(含內層)且第n檔也在內層以內(含內層)，則掛在內層(買賣開關黃燈提示)，若第n+1檔也在內層以內(含內層)，則暫時抽單不下。  
      \* n為config中設定參數

**[舉例]：**  
**n=2，買內層= -0.3，買外層= -0.6，市場現貨有效買價=95.5，則買內層價=95.213(無條件捨去至交易所Tick=95.2)，買外層價=94.927(無條件捨去至交易所Tick=94.9)**

Case1：若期貨市場無人掛買單，則掛在94.9

Case2：若期貨市場有人掛買單在95，則掛在95.1

Case3：若期貨市場有人掛買單在95.3，則掛在95.2(買賣開關黃燈提示)；若期貨市場有人掛買單在95.4，則暫時抽單不下。

**n=2，賣內層= 0.3，賣外層= 0.6，市場現貨有效賣價=95.5，則賣內層價=95.787(無條件進位至交易所Tick=95.8)，賣外層價=96.073(無條件進位至交易所Tick=96.1)**

Case1：若期貨市場無人掛賣單，則掛在96.1

Case2：若期貨市場有人掛賣單在96，則掛在95.9

Case3：若期貨市場有人掛賣單在95.7則掛在95.8(買賣開關黃燈提示)；若期貨市場有人掛賣單在95.6，則暫時抽單不下。

1. **改單邏輯(進)**

發現有人掛單在前面時，依據以上ii~iii邏輯判斷，利用改單秒數(進)進行改單

1. **改單邏輯(退)**

若掛單掛在內層及外層(含外層)之間，當發現原本前面的單子已抽單時，利用改單秒數(退)進行改單，而改單後的位置會因為別人掛單的位置有所不同，分別陳列如下：

* 1. 無人掛單時，掛至外層
  2. 其它人掛單在內層及外層(含外層)之間，則掛在別人掛單的位置前一個tick，超過外層則掛外層
  3. 若已掛在內層及外層(含外層)之間，但因為現貨有效價格變動，導致掛單價已觸及內層價時，先取消，等期貨行情來再判斷掛單位置

1. **抽單**
2. 當漲(跌)幅碰觸到S%則關閉期貨賣(買)單開關，並且抽掉該方向的下單
3. **搶單：**
4. 當發現市場有成交在內層以外(含內層)時，系統自動下FLots口數的ROD單，N秒後自動取消
5. 若N秒內搶單全數成交時，需等到市場又有新的成交價出現時才會再下FLots口數的新單子
6. 會偵測自己的搶單價而不會與其競價
7. N秒內因為現貨之變動，導致搶單已觸及內層時取消下單