(三)Index 物件重要方法

- ▶ 建構在 Index 物件上的方法
 - Function : name(): ◆ 語法 def name(self):

return _QuantLib.Index_name(self)

- ◆ 使用
- Function : fixingCalendar():
 - ◇ 語法

◆ 使用

```
def fixingCalendar(self):
return QuantLib.Index fixingCalendar(self)
```

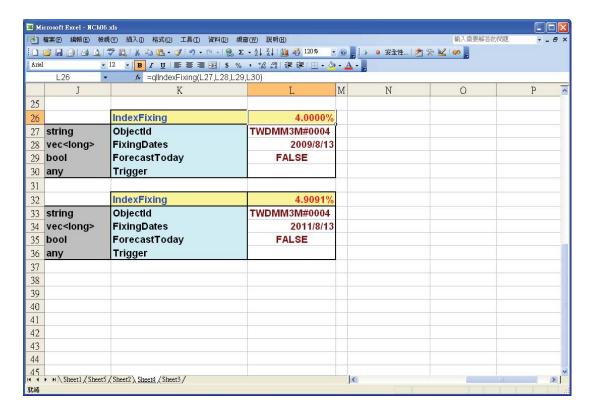
Andy M. Dong

◆ 比價利率的增刪

- Function : fixing():
 - ◇ 語法

```
def fixing(self, fixingDate, forecastTodaysFixing=False):
return _QuantLib.Index_fixing(self, fixingDate, forecastTodaysFixing)
```

- ◆ 使用
 - ✓ 根據輸入的比價日向量,傳回相對應的比價向量,如果事先沒有輸入該日的資料,則以 Index 物件中的利率曲 線物件來估計。
 - ✓ 第三個參數是控制如果沒有今日的比價日資訊,是否要以 Index 物件中的利率曲線物件來估計,預設值為 False •
 - ✓ 下圖中,我們要求 2009/8/13 的比價利率,傳回 4.00%。這是我們之前已經加入的比價日資訊。當我們要求 2011/8/13 的比價利率,則傳回 4.9091%。這是根據 Index 物件中的利率曲線物件來估計。
 - ✓ 雖然我們利率曲線是以 5% Flat Forward 建立的物件,但是由於 Index 的複利方式與利率曲線的複利方式不同, 因此有一些差距。



Andy M. Dong 2

- Function : isValidFixingDate():
 - ◇ 語法

```
def isValidFixingDate(self, fixingDate):
 return _QuantLib.Index_isValidFixingDate(self, fixingDate)
```

- ◆ 使用
 - ✔ 根據輸入的比價日向量,傳回比價日向量是否為有效日期的向量。
- Function : clearFixings():
 - ◆ 語法

```
def clearFixings(self):
 return _QuantLib.Index_clearFixings(self)
```

- ◆ 使用
 - ✓ 輸入的 Index 物件的所有比價資料清除。

- Function : addFixing():
 - ◇ 語法

def addFixing(self, fixingDate, fixing, forceOverwrite=False):
 return _QuantLib.Index_addFixing(self, fixingDate, fixing, forceOverwrite)

- ◆ 使用
- ✓ 輸入的比價日向量以及比價利率大小,加到輸入的 Index 物件之內。第四個參數控制是否要把已有日期的比價 利率以新的資料覆蓋上去。
- Function : addFixings():
 - ◇ 語法

def addFixings(self, fixingDates, fixings, forceOverwrite=False):
 return _QuantLib.Index_addFixings(self, fixingDates, fixings, forceOverwrite)

- ◆ 使用
 - ✓ 輸入的比價日向量以及比價利率大小,加到輸入的 Index 物件之內。第四個參數控制是否要把已有日期的比價 利率以新的資料覆蓋上去。
 - ✓ 在此例中,我們加入一些已經比價的資料,2009/8/13、2009/11/13、2010/2/13、2010/5/13、...、2011/2/13,他們的比價利率皆設為4.0%。注意,由於我們評價日為2011/7/11,只有此日前的日期有比價資料。

Andy M. Dong 27

