Version 8.1 - 2016/07/29

Correction N.1
Page - 134
Line - Equation 6.25
Description - Substitute Equation 6.25 with

$$\rho_{k,l} = \rho_\infty + (1 - \rho_\infty)\, e^{-\beta |T_k - T_l| e^{-min(T_k, T_l)\alpha}}$$

Correction N.2
Page - 135
Line - Table 6.4
Description - Substitute Table 6.4 with the following one:

```
import numpy

def generateParametricCorrelationMatrix(alpha, beta,
rho_inf, maturity_grid):
    '''
    Function which generates a correlation matrix
    using the double exponential parameterization.

    @var alpha: alpha parameter
    @var beta: beta parameter
    @var rho_inf: rho_inf parameter
    @var maturity_grid: represents a list containing the
    forward rate maturities which we are going to model.
    For example, if we want to model the semi-annual
    forward rates maturing between 1 and 5 years from now,
    we will set:
    maturity_grid = [1, 1.5, 2, 2.5, 3, 3.5, 4, 4.5, 5]
    '''
    grid = len(maturity_grid)
    corr_matrix = numpy.zeros((grid, grid))
    for i in range(grid):
        for j in range(grid):
            first_e = numpy.exp(-beta *\
            abs(maturity_grid[i] - maturity_grid[j])\
            * numpy.exp(-alpha *\
            min(maturity_grid[i], maturity_grid[j])))
            corr_matrix[i,j] = (rho_inf + (1.0 - rho_inf)
                                * first_e)

    return corr_matrix
```

Correction N.3

Page - 181

Line - 1

Description - Substitute the following text:

"A drawback of this approach is that when employed, the swap approximation introduced by Rebonato and White (2009) and presented in Section 7.5.3, does not work. This is however not a problem when the SABR LMM is used, as in the context of this book, for pricing and hedging LIBOR exotics (as this approximation is unecessary). We anyhow believe that also the reader interested in using the SABR LMM to price swap dependent instruments can benefit from the simplifications brought by employing a null forward-volatility correlation sub-matrix.",

with:

"A drawback of this approach is that when employed, the swap approximation introduced by Rebonato and White (2009) and presented in Section 7.5.3, does not work for high $\beta_{m,n}$. This is however not a problem when the SABR LMM is used, as in the context of this book, for pricing and hedging non-callable LIBOR exotics (as this approximation is unecessary). We anyhow believe that also the reader interested in using the SABR LMM to price swap dependent instruments, in markets characterized by high $\beta_{m,n}$, can benefit from the simplifications brought by employing a null forward-volatility correlation sub-matrix. "

Correction N.4

Page - 198

Line - 5 from below

Description - Substitute: "The forward swap rate at T_m is obtained through a weighted sum of forward rates and it can be approximated as

$$S_{m,n}(T_m) \approx \sum_{k=m+1}^{n} \omega_k(0) F_k(T_m).$$

"

with: "Alternatively, the forward swap rate at T_m is obtained through a weighted sum of forward rates and it is given by

$$S_{m,n}(T_m) = \sum_{k=m+1}^{n} \omega_k(T_m) F_k(T_m).$$

"

Correction N.5

Page - 201

Line - The first equation from below should be substituted with

$$\Omega_{k,l} = \frac{2\rho_{k,l}\vartheta_{k,l}\mathbf{W}_k^{m,n}\mathbf{W}_l^{m,n} s_k(0)\, s_l(0)}{(\nu_{m,n}\alpha_{m,n}T_m)^2}$$

$$\cdot \int_0^{T_m} g\left(T_{k-1}-t\right) g\left(T_{l-1}-t\right) \int_0^t h\left(T_{k-1}-s\right) h\left(T_{l-1}-s\right) dsdt,$$

Correction N.6
Page - 202
Line - Footnote 4
Description - Substitute the following text: "does not work when using the null forward-volatility correlation approach.", with: "does not work for high $\beta_{m,n}$."

Correction N.7
Page - 196
Line - Table 7.7
Substitute the following line of code:

```
drift_correction[fwd_k - 1] = corr_fwd_fwd[fwd_k - 1,
    fwd_k] * (tau * g_t * s_t[fwd_k] * F_beta_t) / (1 +
    tau * F_t[fwd_k])
```

with:

```
drift_correction[fwd_k - 1] = (tau * g_t * s_t[fwd_k] *
    F_beta_t) / (1 + tau * F_t[fwd_k])
```

Correction N.8
Page - 197
Line - Table 7.7
Substitute the following line of code:

```
shared_drift_part[fwd_k - 1] += drift_correction[l - 1]
```

with:

```
shared_drift_part[fwd_k - 1] += corr_fwd_fwd[fwd_k - 1, l
    ] * drift_correction[l - 1]
```