Errata to Option Pricing and Volatility Using Excel/VBA
Fabrice Douglas Rouah and Gregory Vainberg

Please visit www.Volopta.com for free derivatives pricing code written in C++, Matlab, VBA and more!

- Page 9. The update step in the VBA function NewtRapNum() should be changed to

  cur_x = cur_x − (fx_delta_x / dx).

  Thanks to *Mikael Petitjean.*

- Page 9. The equation in the middle of the page should be changed to

  $$f'(x) = \frac{f(x) - f(x - dx)}{dx}$$

  and the phrases preceding the equation should read

  "...joining the function at x and $x - dx$..."

  "...Taylor series expansion for $f(x)$ about $f(x - dx)$."

  Thanks to *Mikael Petitjean.*

- Page 28, fifth to last line. The statement

  l(n) = 1: z(n) = 0: c(n) = 0

  should be

  l(n) = 1: z(n) = 0: u(n) = 0

  The VBA function NSpline() in the Excel file Chapter1NatSpline should be updated to reflect this change. Please note that this typo does not affect the functionality of the NSpline() function. Thanks to *Josua Müller.*

- Page 32, Exercise 1.4. The matrix inversion in the expression for the hat matrix has been omitted. The correct expression for the hat matrix is $X\left(X^{T}X\right)^{-1}X^{T}$. Note that the VBA code in the solution is correct. Thanks to *Pierluigi Cerruti.*

- Page 51, middle of the page. The line should be changed from

  ```
  BOOLEnumint = (1 / 45) * (14 * y(ind) + 64 * y(ind + 1)
               + 24 * y(ind + 2) + 64 * y(ind + 3)
                + 14 * y(ind + 4)) * deltax
  ```

  to

  ```
  BOOLEnumint = BOOLEnumint + (1 / 45) * (14 * y(ind)
                + 64 * y(ind + 1) + 24 * y(ind + 2)
                + 64 * y(ind + 3) + 14 * y(ind + 4)) * deltax
  ```

  Please note that the VBA code is correct in the Excel file in which this code appears (Chapter2Newton). Thanks to *Hyosik Jeon*.

- Page 76, second to last line. "Term1" should be changed to "Term2". Hence, the line should be changed from

  ```
  Term1 = (d2 / (n + 1 / 3 - (1 - Method) * 0.1
  ```

  to

  ```
  Term2 = (d2 / (n + 1 / 3 - (1 - Method) * 0.1
  ```

  Please note that this change must also be done to the LRCash() function in the Excel file Chapter8CashorNothing.

- Page 77, first line. "Term1" should be changed to "Term2". Hence, the line should be changed from

  ```
  p = 0.5 + Sgn(d2) * 0.5 * Sqr(1 - Exp(-Term1))
  ```

  to

  ```
  p = 0.5 + Sgn(d2) * 0.5 * Sqr(1 - Exp(-Term2))
  ```

  Please note that this change must also be done to the LRCash() function in the Excel file Chapter8CashorNothing.

- Page 105, Equation 3.19. The D_i inside the summation should be D_z. Thanks to *Pierluigi Cerruti*.

- Page 141. The expression for $\ln(v_t)$ in the VBA code should be

  ```
  lnvt = lnvt + (1/curv) * (kappa * (theta − curv) − 0.5 * sigmav^2)
      * deltat + sigmav * sqr(deltat/curv) * ev
  ```

  This will make the code consistent with Equation (5.9) on page 140. The code in the Excel file Chapter5Simulate should be changed to reflect this typo. Thanks to *Soon-Kian Phang*, *Philippe Ledent* and *Lorenz Schneider*.

- Page 142. *Brian Byrne* notes that the HestonMC() function for simulating the Heston (1993) call price that appears in the Chapter5HestonMC seems to break down for large values of the *daynum* parameter. Decreasing the number of simulations will fix the problem, and the program will run fine with long maturities. VBA seems to break down when there are too many iterations. Maybe there is a global setting in VBA that needs to be changed. Any insight from readers would be appreciated. Please note that this program has been reproduced in C++ and runs fine for long maturities and a large number of simulations. The C++ code is available at **www.Volopta.com**. Update: *Lorenz Schneider* explains that this is especially true when σ (sigmav) is greater than one. When curv is small the term -0.5*sigmav^2/curv can drag the value of LnVt down so that curv goes quickly to zero. Division by zero will cause the algorithm to crash. He suggests adding a floor on curv in the VBA code, for example curv = max(curv,0.001).

- Page 170. In the GARCHMLE() function contained in the Excel files Chapter6GARCH and Chapter6Exercises, the condition

  If ((omega < 0) Or (alpha < 0) Or (beta < 0))

  should be replaced with

  If ((omega < 0) Or (alpha < 0) Or (beta < 0) Or (alpha + beta >= 1))

  That should ensure $\alpha + \beta < 1$. Thanks to *Nikolay Kachakliev.*

- Page 173. The phrase in the middle of the page

  "The Excel file Chapter6LongRun contains the very simple VBA function YearVol() for computing $\sigma(T)$."

  should be replaced with

  "The Excel file Chapter6GARCH contains the very simple VBA function YearVol() for computing $\sigma(T)$."

  Thanks to *Piotr Jagodowicz.*

- Page 179. Heston and Nandi (2000) parameter values. *Tim Krehbiel* writes "Model instability arises when non-zero risk free rates are introduced." He finds that for non-zero rates, the upper constraints

  if P_i >1 then P_i = 1

  for $i = 1, 2$, are binding. Any insight from readers would be appreciated.

- Page 279. In Equation (9.4) the minus sign in front of $\lambda\sigma^2_{t+1}$ should be a plus sign. It should be exactly like Equation (6.19) on page 175

  $$r_{t+1} = r + \lambda\sigma^2_{t+1} + \sigma_{t+1}z_{t+1}$$

  Thanks to *Anders Lund Lindhardsen.*

- Page 280. The expression for VARt(t) in the HNGARCHMLE() function, which appears in lines 8 and 9 on page 280 as

  VARt(t) = omega + beta*VARt(t+1) + alpha
  *(Z(t+1) - gamma*VARt(t+1))^2

must be changed to

```
VARt(t) = omega + beta*VARt(t+1) + alpha
        *(Z(t+1) - gamma*Sqr(VARt(t+1)))^2
```

This change must be made to the HNGARCHMLE() function that appears in the Excel file Chapter9HNGarch.

Please note that the formulas in the Chapter9HNGarch file must also be updated. Hence, in cell E3, the expression

```
= omega + beta*E4 + alpha*(F4 - gamma*E4)^2
```

must be changed to

```
= omega + beta*E4 + alpha*(F4 - gamma*sqrt(E4))^2
```

The updated formula can then be copied to cells E4:E252. Thanks to *Tim Krehbiel* and *Stanley Luo*.

- Page 297. The parameter checks in the VBA code should be

  ```
  If (V0 < 0) Or (kappa < 0) Or (thet < 0)
  ```

  Please note that the VBA code is correct in the Excel file in which this code appears (Chapter9Exercise2).