Janvier 2026

Telecom Sud Paris – Risk Management

Abdelkader BOUSABAA

Projet 1 : Approche RaRoC

Objectif : Mettre en œuvre un outil de tarification de type RaRoC sur une opération de crédit. L'outil devra traiter une approche au niveau du crédit, une approche au niveau du client et une approche au niveau d'un portefeuille de crédits/clients. Prendre en compte les garanties et les garants, ainsi que le risque pays.

Base de données : Utiliser la base de données sous Excel "Credit_Portfolio.xls" qui vous est fourni en pièce jointe.

Interface : Le projet doit être réalisé en Python avec une interface professionnelle (formulaires sous Excel ou sous Python ou en Web application). Tous les calculs doivent être réalisés en Python et non dans Excel. Seuls les graphiques pourront être présentés sous Excel si vous ne pouvez pas faire autrement.

Organisation : Les projets doivent être réalisés individuellement.

Toutes les informations nécessaires (corrélation, ratings, structure par terme des probabilités de défaut, …) sont fournies dans la table Excel.

Résultats minimaux attendus :

1. Des formulaires afin de pouvoir
    a. consulter/modifier les paramètres de calcul
    b. visualiser les 10 premières lignes de la table "Portfolio"
    c. lancer les calculs et visualiser les résultats
    d. afficher les résultats d'une opération comme un mini-compte de résultat
2. Un code propre (i.e. lisible et bien indenté), bien structuré (modules, classes, …) et commenté
3. Un document ".pdf" présentant l'objectif de votre projet de manière exhaustive

Projet 2 : Gestion de portefeuilles

Objectif : Mettre en œuvre un outil de simulation de type Monte Carlo pour le calcul d'indicateurs de risque (Value at Risk, Expected ShortFall, Expected Loss, …) sur un portefeuille de crédit. Nous attendons une partie "Gestion de portefeuilles" et une partie "Dérivés de Crédit" (au minimum tranche de CDO).

Base de données : Utiliser la base de données sous Excel "Credit_Portfolio.xls" qui vous est fourni en pièce jointe.

Interface : Le projet doit être réalisé en Python avec une interface professionnelle (formulaires sous Excel ou sous Python ou en Web application). Tous les calculs doivent être réalisés en Python et non dans Excel. Seuls les graphiques pourront être présentés sous Excel si vous ne pouvez pas faire autrement.

Organisation : Les projets doivent être réalisés individuellement.

Modèle de défaut : Le modèle de défaut est un modèle gaussien à deux facteurs (MG2F) dont la diffusion est donnée par

$$Z_i = \sqrt{\rho}\, X + \sqrt{\rho_S - \rho}\, X_S + \sqrt{1 - \rho_S}\, \varepsilon_i$$

Toutes les variables sont gaussiennes et les barrières de défaut sont données par la relation suivante :

$$B_i = \Phi^{-1}(PD_i)$$

Toutes les informations nécessaires (corrélation, corrélations sectorielles, ratings, structure par terme des probabilités de défaut, …) sont fournies dans la table Excel.

Résultats minimaux attendus :

1. Des formulaires afin de pouvoir
    a. consulter/modifier les paramètres de calcul
    b. visualiser les 10 premières lignes de la table "Portfolio"
    c. lancer les calculs et visualiser les résultats
2. Un graphique de convergence de la perte moyenne du portefeuille en fonction du nombre de simulations
3. Un code propre (i.e. lisible et bien indenté), bien structuré (modules, classes, …) et commenté
4. Un document ".pdf" présentant l'objectif de votre projet de manière exhaustive

Projet 3 : Pricing d'options

Objectif : Mettre en œuvre un outil de simulation de type Monte Carlo pour le pricing d'options sur actions : Vanille, Tunnel, Himalaya, Napoléon, … Nous attendons le prix de l'option, l'erreur à 99% et le graphique de convergence.

Base de données : Aucune base de données n'est fournie pour ce projet. Vous rappellerez les caractéristiques de chaque option.

Interface : Le projet doit être réalisé en Python avec une interface professionnelle (formulaires sous Excel ou sous Python ou en Web application). Tous les calculs doivent être réalisés en Python et non dans Excel. Seuls les graphiques pourront être présentés sous Excel si vous ne pouvez pas faire autrement.

Organisation : Les projets doivent être réalisés individuellement.

Modèle de défaut : Le modèle de diffusion utilisé est donné par

$$S_t = S_{t-1}\exp\left[\left(\mu - \frac{\sigma^2}{2}\right)dt + \sigma\sqrt{dt}W\right]$$

Le processus W suit une loi normale centrée réduite.

Résultats minimaux attendus :

1. Des formulaires afin de pouvoir
    a. consulter/modifier les paramètres de calcul
    b. lancer les calculs et visualiser les résultats
2. Un graphique de convergence du prix de l'option en fonction du nombre de simulations
3. Un code propre (i.e. lisible et bien indenté), bien structuré (modules, classes, …) et commenté
4. Un document ".pdf" présentant l'objectif de votre projet de manière exhaustive

Projet 4 : EDP

Objectif : L'objet du projet est de présenter un outil de résolution d'EDP. Vous devez utiliser l'algorithme présenté dans les slides au chapitre "EDP - Application à la finance". Cet algorithme générique permet de résoudre les EDP dont le format est compatible avec l'équation fondamentale de la finance.

Interface : Le projet doit être réalisé en Python avec une interface professionnelle sous Excel ou sous Jupyter. Tous les calculs doivent être réalisés en Python (avec algorithme de Thomas) et non dans Excel. Seuls les graphiques pourront être présentés sous Excel/Jupyter.

Organisation : Les projets doivent être réalisés individuellement.

Modèle de pricing : Les modèles à implémenter sont au minimum de 3 :

- Black & Scholes
- CIR
- Merton

Cf. annexe

Résultats attendus : Vous devez être en mesure de reproduire les 3 exemples étudiés en cours et dont les caractéristiques sont en annexes.

- Un code propre (i.e. lisible et bien indenté), bien structuré (modules, classes, …) et commenté
- Un document ".pdf" présentant l'objectif de votre projet de manière exhaustive

Annexe

Modèle de Cox, Ingersoll et Ross [1995]

```
kappa = 0.8
theta = 0.10
sigma = 0.5
lambda = 0.05

mu(t,x) = kappa * (theta - x)
sigma(t,x) = sigma * racine(x)
lambda(t,x) = lambda * racine(x) / sigma
r(t,x) = x

aProc(t,x) = 0.5 * sigma(t,x)^2
bProc(t,x) = mu(t,x) - lambda(t,x)*sigma(t,x)
cProc(t,x) = r(t,x)
dProc(t,x) = 0
tminBound(t,x) = 1

tmin = 0
tmax = 5
Nt = 101
xmin = 0
xmax = 1
Nx = 51

theta = 0.5       // Résolution numérique (algorithme de Crank-Nicholson)
```

Modèle de Vasicek

```
a = 0.95
b = 0.10
sigma = 0.2
lambda = 0.05
bprime = b - lambda*sigma/a
r(t,x) = x

aProc(t,x) = 0.5*(sigma^2)
bProc(t,x) = a*(bprime-x)
cProc(t,x) = r(t,x)
dProc(t,x) = 0
tminBound(t,x) = 1

tmin = 0
tmax = 5
```

Nt = 1001
xmin = -1
xmax = 1
Nx = 101

theta = 0, 0.25, 0.50, 0.75, … , 5

Modèle de Black & Scholes

K = 100
sigma = 0.20
tau = 0.25
r = 0.08
b = -0.04

aProc(t,x) = 0.5*(sigma^2) * (x^2)
bProc(t,x) = b * x
cProc(t,x) = r
dProc(t,x) = 0

tminBound(t,x) = max(x – K,0)
xminBound(t,x) = 0
xmaxBound(t,x) = max(x – K,0)
DxmaxBound(t,x) = 1

xmin = 50
xmax = 150
Nx = 201
tmin = 0
tmax = tau
Nt = 1000