

**Exercise 1**

Let  $S^1 = (S_t^1)_{t \in [0, T]}$  and  $S^2 = (S_t^2)_{t \in [0, T]}$  be two assets following the risk-neutral dynamics

$$dS_t^1 = rS_t^1 dt + \sigma_1 S_t^1 dW_t^1, \quad 0 \leq t \leq T, \quad (1)$$

$$dS_t^2 = rS_t^2 dt + \sigma_2 S_t^2 dW_t^2, \quad 0 \leq t \leq T, \quad (2)$$

for risk-free rate  $r > 0$ , constant volatilities  $\sigma_1, \sigma_2 > 0$  and correlated Brownian motions  $\langle W^1, W^2 \rangle_t = \rho t$ ,  $\rho \in [-1, 1]$ .

An exchange option for  $S^1$  to  $S^2$  with maturity  $T$  is a product that pays  $(S_T^1 - S_T^2)^+$  at time  $T$ . It can be seen that the value at time 0 of the exchange option is

$$C^{BS}(0, S_0^1, S_0^2, \sigma, T), \quad (3)$$

where  $\sigma = \sqrt{\sigma_1^2 - 2\rho\sigma_1\sigma_2 + \sigma_2^2}$  and  $C^{BS}(r, S, K, \sigma, T)$  is the discounted Black Scholes value of a Call option of risk-free rate  $r$ , spot price  $S$ , strike  $K$ , volatility  $\sigma$  and maturity  $T$ .

Give the implementation of a class `ExchangeOption` that extends

```
net.finmath.montecarlo.assetderivativevaluation.products.AbstractAssetMonteCarloProduct.
```

In particular, the method

```
getValue(double evaluationTime, AssetModelMonteCarloSimulationModel model)
```

has to be implemented in such a way that returns the (discounted) payoff of an exchange option, based on Monte-Carlo.

**Hint:** in order to give a general implementation, you can suppose the argument `model` of type `AssetModelMonteCarloSimulationModel` to represent an  $n$ -dimensional process, for general  $n$ , and that the processes in (1) and (2) are the  $i$ -th and  $j$ -th component of the process represented by `model`, identified by their index. Look at the methods of `AssetModelMonteCarloSimulationModel` that you can use for your scope.

**Exercise 2**

Write a `JUnit` test class with the following three methods:

- A method which checks if the value of the exchange option with an underlying constructed with a seed at your choice approximates (3) up to a tolerance of 2%.
- A method which checks that, out of 500 computations of the value of an exchange option whose underlying is constructed with a random seed, the price approximates (3) up to a tolerance of 2% at least in the 90% of cases.
- A method where you check if the price of the exchange option increases or decreases with respect to the correlation  $\rho$ : what do you expect?

Use parameters' values of your choice, as long as you think they make sense.

**Hints:**

- The main point of this exercise is how to construct the object that you have to pass to `getValue`. A *possible* choice is to use a constructor of

```
net.finmath.montecarlo.assetderivativevaluation.MonteCarloMultiAssetBlackScholesModel.
```

In this case, you have to focus in particular on how to construct the object representing the Brownian motions driving the model and the correlation matrix.

- The value in (3) can be computed using the method `computeAnalyticalValue` that you can find in `net.finmath.functions.AnalyticFormulas`.