

# Bachelier vs. Black

You will learn here how to simulate a stock price using Java, and how to show it using JfreeChart library. It starts with more complex concepts (don't be afraid) and goes down towards simpler things.

## 1. Price of binary option with very high volatility

A [binary call option](#) pays a fixed amount if asset price at maturity is over a strike price (a reference level for the asset). Intuitively, the price under very high volatility of a binary option paying 1\$ should be 0.5\$ whether it is a call or a put. This is because under very high volatility, the spot at expiration can be anything (0.01\$ or 1000\$).

However when one looks at the analytic formula derived from Black and Scholes model, the price of a binary call is 0\$ (the price of a binary put is 1\$).

$$c = e^{-rT} N(d) \quad \text{and} \quad p = e^{-rT} N(-d)$$

where  $d = \ln(S/X) + (b - \text{vol}^2/2)T$

c = call option price, p = put option price, T = time to maturity, S = Spot, X = strike, r = interest rate, b = interest - dividend rate, vol = volatility, N = [cumulative normal distribution](#).

So when vol is very high and S near X,  $d \approx -T \text{vol}^2/2$

If you look at the wikipedia link at the graph for the cumulative distribution function, you will see that  $N(-d) \rightarrow 0$  when d grows, and  $N(d) \rightarrow 1$

This is very different from our intuitive 0.5\$ for put and call. If you reverse the definition for the call, if the price is 0, it means we are sure to always be under the strike level, whatever the initial asset price. This means the asset price is always going to 0 under very high volatility. If the asset price is going to 0 then the digital put price is 1, which is consistent with the formula.

## 2. Why does an asset price go toward 0 under very high volatility

It took me a while to figure out but the answer is very simple. If you go back to the roots of Black and Scholes model, the main hypothesis is that the asset price obeys the following equation:

$$dS/S = \mu dt + \text{vol} dZ \quad [1]$$

where Z is a process following a normal distribution with mean 0 and unit variance,  $\mu$  is a growth rate, and vol corresponds to a volatility parameter.

Using Ito transform, this gives

$$S(t) = S(t_0) e^{(\mu - \text{vol}^2/2)(t-t_0) + \text{vol} \sqrt{(t-t_0)} Z}$$

When vol grows,  $S \rightarrow 0$  because  $\exp(-X) \rightarrow 0$  when X grows.

I wrote a small Java program to display the path followed by an asset initially at 100. I relied on Jakarta common maths library for a few functions, and the NormalizedRandomGenerator. Each horizontal number represents a day in the future ( $t - t_0 = 1$  day).

```
NormalizedRandomGenerator gen = new GaussianRandomGenerator(new
JDKRandomGenerator());
double blackVol = 0.157*30d;
```

```

    double initialSpot = 100;
    int days = 248;
    double[] normalSerie = AbstractPathGenerator.makeNormalSerie(gen, days);
    System.out.println("average="+StatUtils.mean(normalSerie)+" ,
variance="+StatUtils.variance(normalSerie));
    BlackGenerator blackGen = new BlackGenerator(blackVol, initialSpot);
    double[] blackSpots = blackGen.computeSpots(normalSerie, days);
    System.out.println(Arrays.toString(normalSerie));
    System.out.println(Arrays.toString(blackSpots));

    XYDataset blackDataset = createDataset("Black", days, blackSpots);

    XYDataset dataset = new CombinedDataset(new SeriesDataset[] {
        blackDataset,
    });

    JFreeChart chart = ChartFactory.createXYLineChart(
        "Black Simulation", "Day", "Spot",
        dataset, PlotOrientation.VERTICAL, true, false, false);
    NumberAxis axis = (NumberAxis) chart.getXYPlot().getRangeAxis();
    axis.setAutoRangeIncludesZero(false);
    axis.setAutoRangeMinimumSize(1.0);

    ChartUtilities.saveChartAsPNG(new File("/tmp/black_simulation.png"),
chart, 640, 480);

```

```

public abstract class AbstractPathGenerator {

    public static final int DAYS_ACT_365 = 248;
    protected double vol;

    protected double initialSpot;

    protected double dayInc = 1d/DAYS_ACT_365;

    public AbstractPathGenerator(double vol, double initialSpot) {
        this.dayInc = 1d/DAYS_ACT_365;
        this.vol = vol;
        this.initialSpot = initialSpot;
    }

    protected abstract double nextSpot(double spot, double normalRandom);

    public double[] computeSpots(double[] normalizedSerie, int days) {
        double[] spots = new double[days+1];
        spots[0] = initialSpot;
        for (int i=0;i<days;i++) {
            spots[i+1] = nextSpot(spots[i], normalizedSerie[i]);
        }
        return spots;
    }

    public double[] computeSpots(NormalizedRandomGenerator gen, int days) {
        double[] normalizedSerie = makeNormalSerie(gen, days);
        return computeSpots(normalizedSerie, days);
    }
}

```

```

public static double[] makeNormalSerie(NormalizedRandomGenerator gen, int
days) {
    double[] normalizedSerie = new double[days];

    for (int i=0;i<days;i++) {
        normalizedSerie[i] = gen.nextNormalizedDouble();
    }
    return normalizedSerie;
}
}

```

```

public class BlackGenerator extends AbstractPathGenerator {

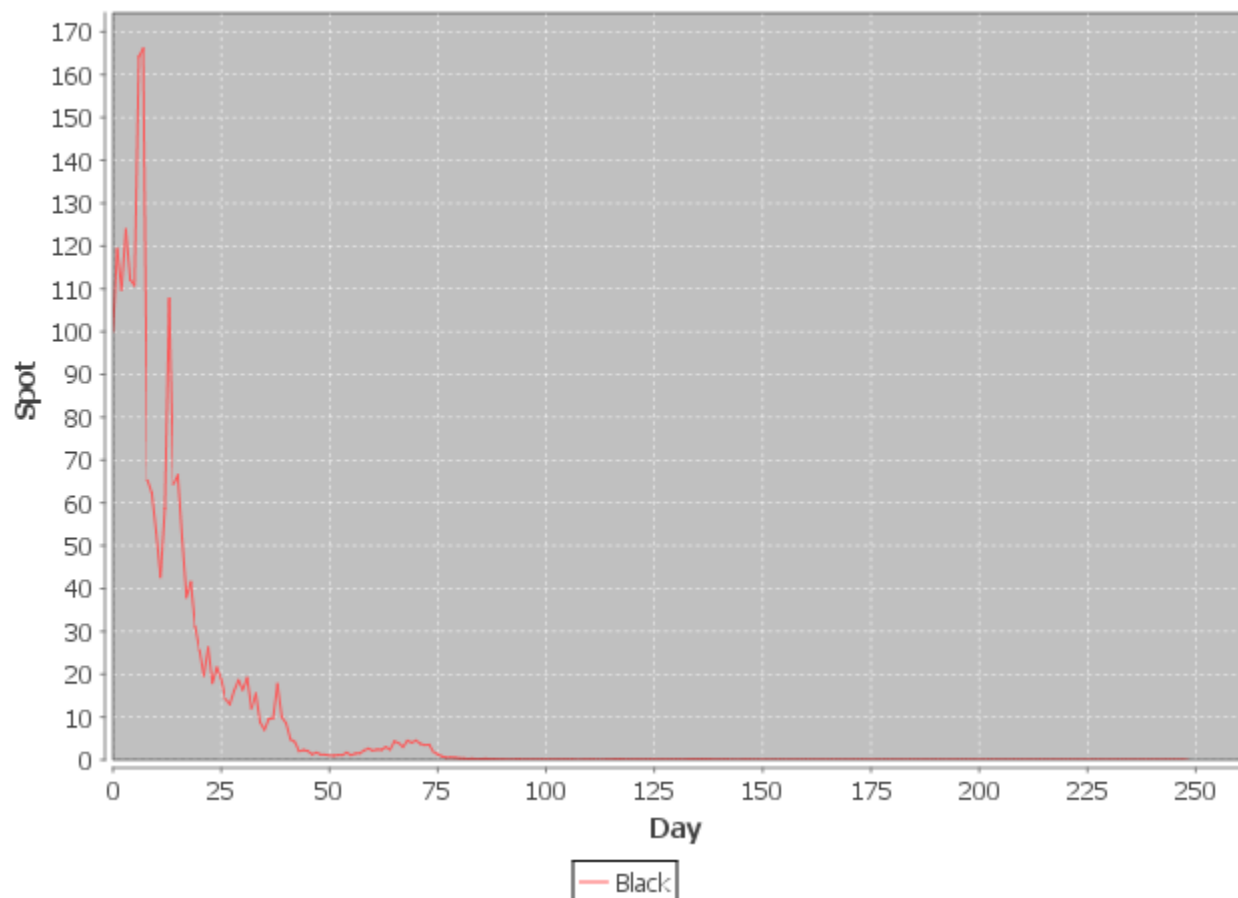
    private double alpha = 0.5;
    public BlackGenerator(double vol, double initialSpot) {
        super(vol, initialSpot);
    }

    protected double nextSpot(double spot, double normalRandom) {
        double nextSpot = spot * Math.exp(-alpha*vol*vol*dayInc +
vol*Math.sqrt(dayInc)*normalRandom);

        return nextSpot;
    }
}

```

## Black Simulation



The root of the intuitive fact that the asset price becomes 0 when volatility grows lies in “lognormality” of the asset process.

### 3. Bachelier

Before Black and Scholes in the sixties, a 19<sup>th</sup> century French economist, Bachelier, invented a method to price options. The main difference is that the asset does not follow a lognormal process but just a normal process:

$$dS = \mu * dt + \text{vol} * dZ \quad [2]$$

Integrating it gives:

$$S(t) = S(t_0) + \mu(t - t_0) + \text{vol} \sqrt{(t - t_0)} Z$$

```
public class BachelierGenerator extends AbstractPathGenerator {  
  
    public BachelierGenerator(double vol, double initialSpot) {  
        super(vol, initialSpot);  
    }  
  
    @Override  
    protected double nextSpot(double spot, double normalRandom) {  
        double nextSpot = spot + vol*Math.sqrt(dayInc)*normalRandom;  
  
        return nextSpot;  
    }  
}
```

I use  $\mu = 0$  in the simulation.

## Bachelier vs. Black



The blue line is represented using the same vol as black. But this is wrong, if you compare [1] and [2], you quickly see that bachelier vol =  $S \cdot \text{black Vol}$ . The green line is bachelier with a vol = initial Spot \* blackvol.

We see here what people criticized about Bachelier, the negative price that does not make sense. However the price keeps on moving a lot and the concept of volatility is more natural. A binary option price under Bachelier would be 0.5\$ under very high volatility.

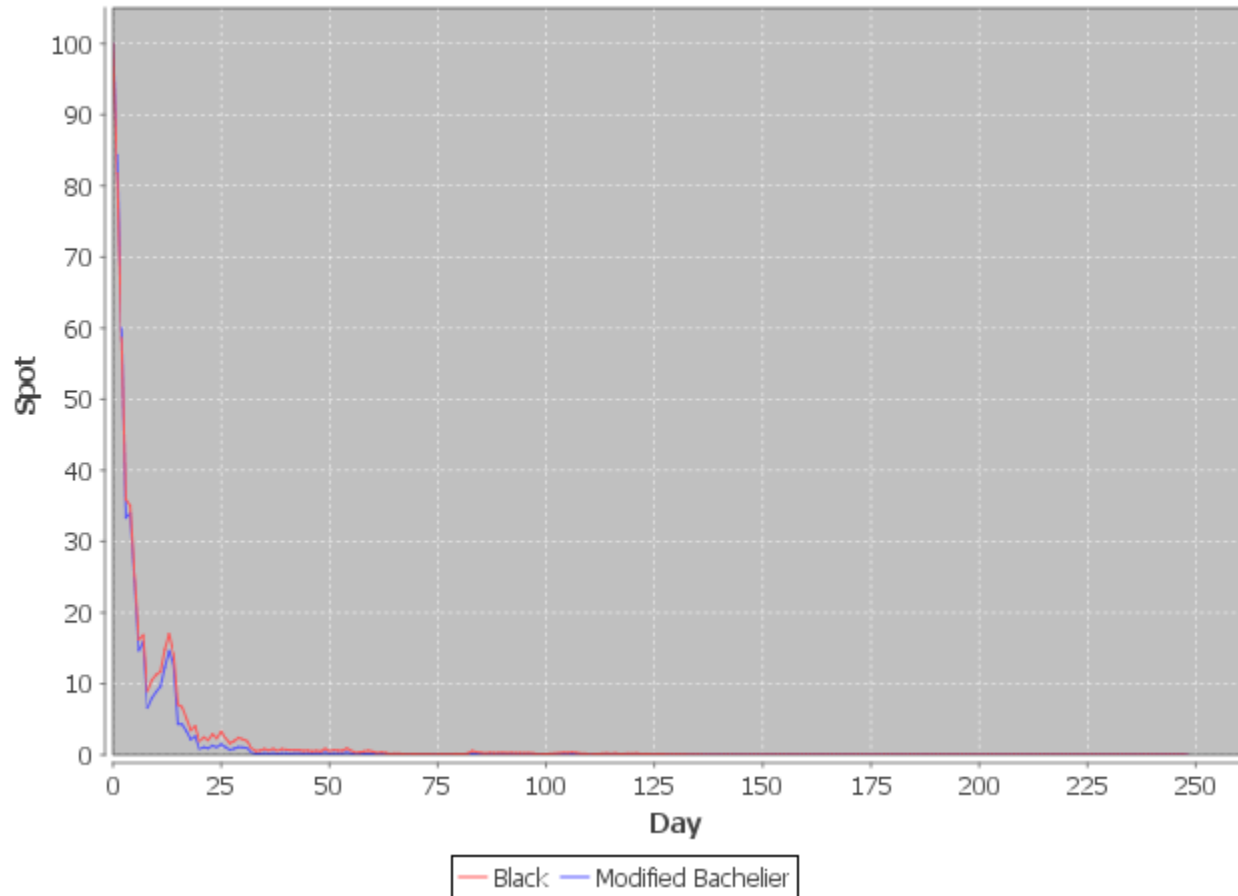
### 4. A Modified Bachelier

While Black volatility is said constant, compared to Bachelier vol, it is not really constant, as it is always the bachelier constant vol multiplied by the previous spot (see [1] and [2]). If we change slightly Bachelier formula to have the same behavior with the vol, we should have something much nearer from Black and Scholes:

$$S(t) = S(t_0) \left( 1 + \mu(t - t_0) + \text{vol} \sqrt{(t - t_0)} Z \right) \quad [3]$$

Using this formula, under very high vol, we are very near Black and Scholes. This is actually because [3] is a quite good approximation of the exponential term. We move back to a geometric change in prices and an asset price going to 0.

## Bachelier vs. Black



If you use a more standard volatility, for example 31.4% (corresponds to 2% a day), then you have much less difference between Bachelier and Black. We can clearly see a drift between Bachelier and Black corresponding to the price change. If the price goes lower, Bachelier price will be a bit higher than the Black price, if the price goes higher, Bachelier price will be a bit lower. And the modified Bachelier is indistinct of Black.

## Bachelier vs. Black



### 5. Solution of the binary option paradox

A volatility parameter that “includes” the spot price changes is obviously a bit more accurate. Why then the digital price under very high vol is inconsistent and yet everybody uses the Black model? This is because a vol of 4.71 (what I used in the graphs under high vol) is not realistic.

A volatility of 4.71 per year means a standard deviation for one day of  $4.71 \cdot \sqrt{1/248}$  assuming 248 trading day per year = 30%. A one standard deviation move of a stock at 100 pushes it to 130 in one day, something that never happens everyday over a year.

But then Bachelier on standard volatility is not that different from Black...