Quantitative Finance

Pricing under Historical Distributions

Patrick Hénaff

Version: 07 Feb 2025

Contents


```
library(lubridate)
library(fExoticOptions)
library(kableExtra)
library(ggplot2)
library(stats)
library(nleqslv)
library(reshape)
```

In this note, we consider methods for pricing derivatives under a historical density for the underlying asset.

In order to motivate this approach, consider Figure 1 which represents the distribution of price for the "SoCal basis". This is the difference between the price of natural gas at Henry Hub (the delivery point in Southern Louisiana for the NYMEX Futures contract), and the price in Southern California. In general, the price difference reflects the cost of transporting gas from East to West, say less than a douzen cents per unit of volume. Once in a while, however, a disruption in supply causes the price differential to jump to $3 or $5. In that context, what is the price of a call option struck at $2?

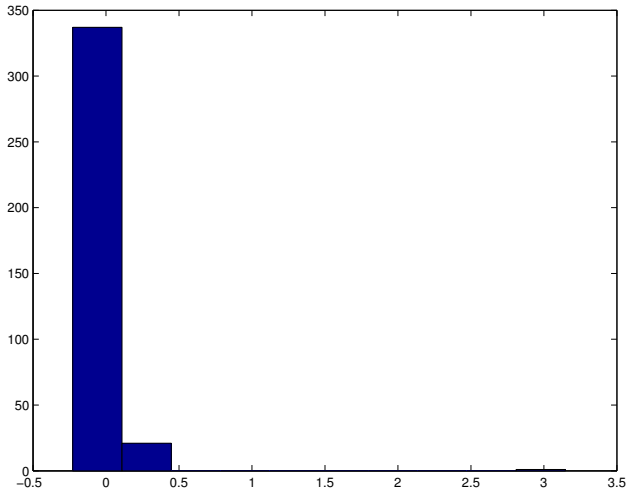

Figure 1: Distribution of So-Cal basis (USD per mmbtu)

It is tempting to make use of historical data, given the very specific nature of the distribution. The question then becomes: Can we transform an empirical density into a density that:

1. is risk-neutral,
2. is consistent with current market observations, and
3. retains the stylized facts of the historical density?

There is a abundant literature on the subject of derivative pricing under empirical densities. See, for example, the review article of Jackwerth (Jackwerth, 1999), as well as the paper by Derman and Zou (Zou, 1999). The principle of Derman's approach is to start with an empirical density for the underlying asset, and to adjust this density in order to make it risk-neutral. Under the risk-neutral density, the expected value of the settlement of a futures contract must be the current forward price, and the fair value of a derivative is the discounted expected value of the payoff.

How do we transform an empirical density into a risk-neutral one? The intuition is to adjust the empirical density in a way that "disturbs" it as little as possible. The concept of entropy is useful to quantify the notion of closeness between distribution.

1 The Derman-Zou density adjustment

1.1 Entropy

The entropy of a random variable X is defined as

$$H(X) = -\sum_i p(x_i) \ln(p(x_i))$$

Therefore, the entropy of a uniform discrete variable with probability $1/N$ associated with each value $x_i, i = 1, \ldots, N$ is $\ln(N)$.

Let P and Q be two discreet random variables. The relative entropy of P and Q measures the closeness between the probability distributions over the same set:

$$\begin{aligned} S(P,Q) &= E_Q\{\log Q - \log P\} \\ &= \sum_x Q(x) \log\left(\frac{Q(x)}{P(x)}\right) \end{aligned} \tag{1}$$

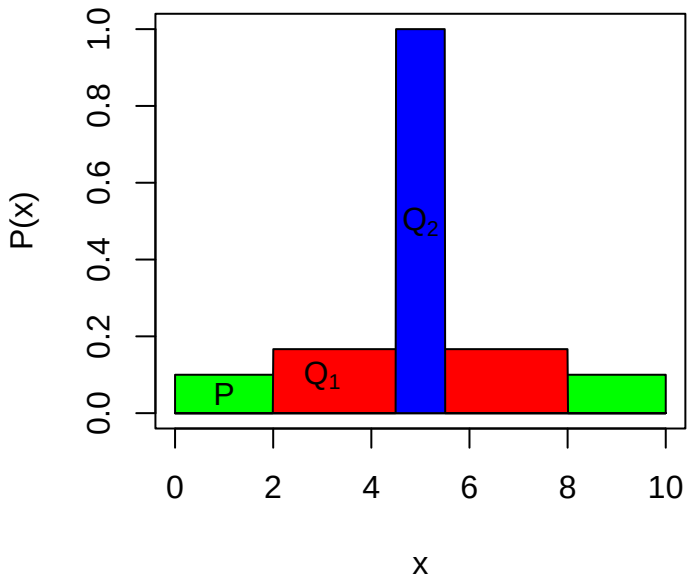

Figure 2: Uniform distributions

In order to form an intuition regarding relative entropy, consider a uniform distribution P over the set $X_N = (x_i, i = 1, \ldots, N)$ and the distribution Q, also uniform over a subset $X_M \in X_N$ of size M:

$$q(x_i) = \begin{cases} \frac{1}{M} & \text{if } x_i \in X_M \\ 0 & \text{otherwise} \end{cases}$$

The distributions are represented in figure 2. Then

$$S(Q,P) = \ln(N) - \ln(M)$$

Now write $M = N^\alpha$, the relative entropy can be expressed by:

$$S(Q,P) = \ln(N)(1-\alpha)$$

The relative entropy is inversely proportional (in a log scale) to the number of elements that are common to X_M and X_N.

A natural procedure to compute the adjusted density Q is to minimize the relative entropy $S(P, Q)$ between the historical density P and Q under the constraint that Q is risk-neutral. We can expand upon this idea by computing Q so that Q satisfies an arbitrary number of additional constraints, in particular that Q prices exactly a number of benchmark derivatives. Next section describes a method for performing this calculation.

1.2 Computation of the risk-neutral density

Consider a prior density P evaluated over n intervals, so that:

$$p_i = P(x_i \leq X \leq x_{i+1})$$

We look for the discrete density Q defined over the sample sampling, which solves the following minimization problem:

$$\begin{array}{ll} \max & -\sum_{i=1}^{n} q_i \log \frac{q_i}{p_i} \\ \text{such that} & \\ & \sum_{i=1}^{n} q_i = 1 \\ & \sum_{i=1}^{n} q_i A_j(x_i) = b_j, \;\; j = i, \ldots, m \end{array}$$

The second set of constraints may be used to calibrate the risk-neutral density to match a variety of market information. For example, to match a given at-the-money volatility (σ), set:

$$\begin{aligned} A_j(x) &= x^2 \\ b_j &= \sigma^2 + \bar{x}^2 \end{aligned} \tag{2}$$

To match the price s of a straddle of strike K, set:

$$\begin{aligned} A_j(x) &= \mid x - K \mid \\ b_j &= s e^{r(T-t)} \end{aligned} \tag{3}$$

Let's write the Lagrangian and the first-order conditions:

$$L = -\sum_i q_i \log \frac{q_i}{p_i} + \sum_j \lambda_j (b_j - \sum_i q_i A_j(x_i)) + \mu (1 - \sum_i q_i)$$

$$\frac{\partial L}{\partial q_i} = -\ln(\frac{q_i}{p_i}) - 1 - \sum_j \lambda_j A_j(x_i) - \mu = 0 \tag{4}$$

$$\frac{\partial L}{\partial \lambda_j} = b_j - \sum_i p_i A_j(x_i) = 0 \tag{5}$$

$$\frac{\partial L}{\partial \mu} = 1 - \sum_i q_i = 0 \tag{6}$$

Using (4), we obtain,

$$\begin{aligned} q_i &= p_i e^{\left(-1-\sum_j \lambda_j A_j(x_i)-\mu\right)} \\ &= p_i e^{(-1-\mu)} e^{-\sum_j \lambda_j A_j(x_i)} \end{aligned} \tag{7}$$

Divide by $\sum_i q_i$ to eliminate μ and the constant term:

$$q_i = \frac{p_i e^{-\sum_j \lambda_j A_j(x_i)}}{\sum_k p_k e^{-\sum_j \lambda_j A_j(x_k)}}$$

To calculate λ_j, use (5):

$$b_j = \sum_i q_i A_j(x_i)$$

and substitute the expression for q_i:

$$b_j = \frac{\sum_i p_i e^{-\sum_j \lambda_j A_j(x_i)} A_j(x_i)}{\sum_i p_i e^{-\sum_j \lambda_j A_j(x_i)}}$$

To summarize, the values for the adjusted probabilities q_i are found to be:

$$q_i = \frac{p_i e^{-\sum_j \lambda_j A_j(x_i)}}{\sum_k p_k e^{-\sum_j \lambda_j A_j(x_k)}}$$

with λ_j solving the system:

$$b_j = \frac{\sum_i p_i e^{-\sum_j \lambda_j A_j(x_i)} A_j(x_i)}{\sum_k p_k e^{-\sum_j \lambda_j A_j(x_k)}},\ j = 1, \ldots, m$$

1.3 Illustration

We first generate an empirical density at horizon 1 month by bootstrapping a series of daily returns, and compute a smoothed density from the sample points. Figure 3 shows the density and the cumulative distribution. This is our prior density P.

```
ts.zc <- get.ts(folder="SBF120", ticker="zc.pa")
nb.samples <- 500
nb.days <- 22
boot.samples <- matrix(sample(ts.zc, size=nb.samples*nb.days, replace=TRUE), nb.samples, nb.days
monthly.means <- apply(boot.samples,1,sum)
adj = 1
eps <- .2 * diff(range(monthly.means))
dens <- density(monthly.means, adjust = adj, from=min(monthly.means)-eps, to=max(monthly.means)+
                n=500)
dens <- data.frame(x=dens$x, y=dens$y)
```

We compute a smooth density from the sample points:

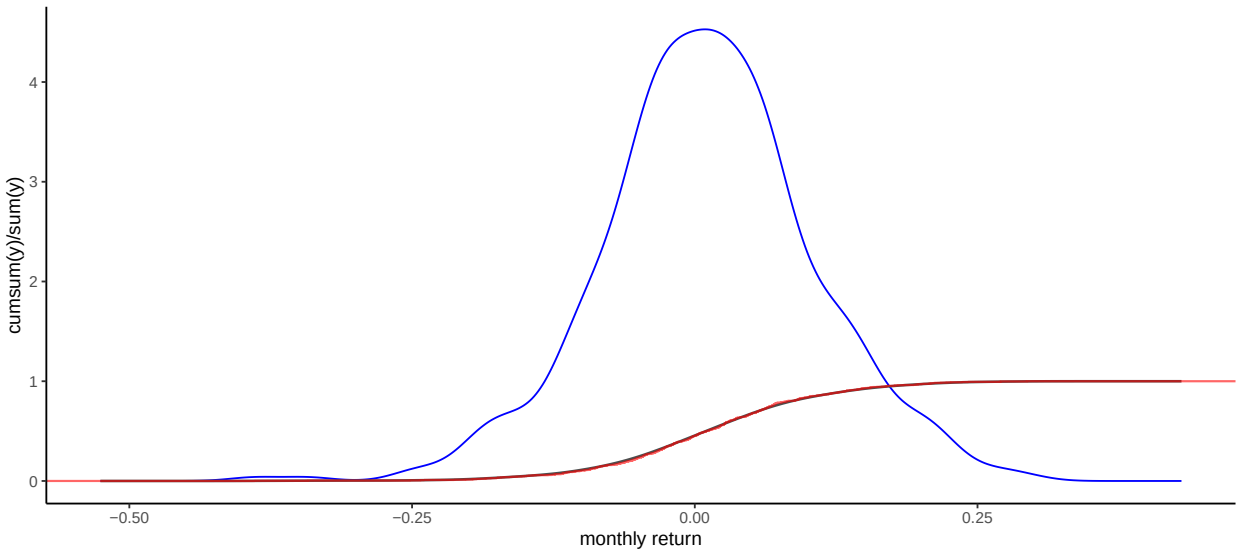

Figure 3: Historical monthly return - Zodiac Aerospace

The historical annualized expected return is 1.1 %, and the annualized volatility is 32.7 %. Assume, however, the risk-free rate is 3% and that ATM straddles are priced at a volatility of 30%. We now adjust the density to be consistent with this information.

The market price of an ATM straddle

```
S.0 <- 100
K <- 100
TTM = 1/12
```

```
r = .03
sigma = .3
riskfree.df <- exp(-r*TTM)
s <- CRRBinomialTreeOption(TypeFlag="ce", S=S.0, X=K, Time=TTM, r=r,
                           b=r, sigma=sigma, n=200)@price +
     CRRBinomialTreeOption(TypeFlag="pe", S=S.0, X=K, Time=TTM, r=r,
                           b=r, sigma=sigma, n=200)@price
```

The A matrix has two rows: row 1 to match the expected return and row 2 to match the market price of the ATM straddle.

```
x = dens$x
S.T <- S.0 * exp(x)
p = dens$y / sum(dens$y)
A <- matrix(c(S.T, abs(S.T-K)), nrow=2, byrow=TRUE)
b <- c(S.0/riskfree.df, s/riskfree.df)
```

Solve for λ_j and compute the q_i:

```
obj.func <- function(l) {
lambda <- matrix(l, nrow=1)
n.1 <- exp(-lambda %*% A)
denom = sum(p*n.1)
num <- vector(mode="numeric", length=2)
num[1] = sum(p * n.1 * A[1,])
num[2] = sum(p * n.1 * A[2,])

err <- b - num / denom
err
}

sol <- nleqslv(c(1,1), obj.func)
lambda <- matrix(sol$x, nrow=1)

n.1 <- exp(-lambda %*% A)
denom <- sum(p*n.1)
q = p*n.1 / denom

df <- data.frame(x=x, p=p, q=as.vector(q))
```

Figure 4 shows the result of the minimum entropy adjustment to the sample distribution, matching the expected risk-free rate and the market price of the ATM straddle.

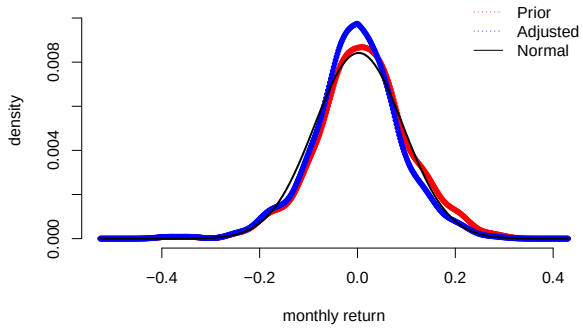

Figure 4: Adjusted densities calibrated to the risk-free rate and to the ATM straddle.

1.4 Recovering the smile

We can now compute the smile implied by the shape of the empirical distribution. The implied volatility as a function of strike is shown in Figure 5.

```
KK <- seq(from=90, to=110, by=2)
c.vol <- vector(mode="numeric", length=length(KK))

for(i in seq_along(KK)) {
  K <- KK[i]
  c.price = sum(df$q * pmax(S.T-K, 0)) * exp(-r*TTM)
  c.vol[i] = GBSVolatility(price=c.price, TypeFlag="c", S=S.0, X=K, Time=TTM, r=r,
                           b=r)
}
```

2 Monte-Carlo pricing with historical paths

The principle here is to use actual historical paths in a Monte-Carlo simulation. In this section, we summarize the model developed by Potters *et al* (**Marc2001?**).

2.1 The hedged Monte-Carlo algorithm

Let's first introduce some notation:

x_k value of underlying asset at step k
$C_k(x_k)$ value of the derivative
$\phi_k(x_k)$ hedge ratio for derivative C_k

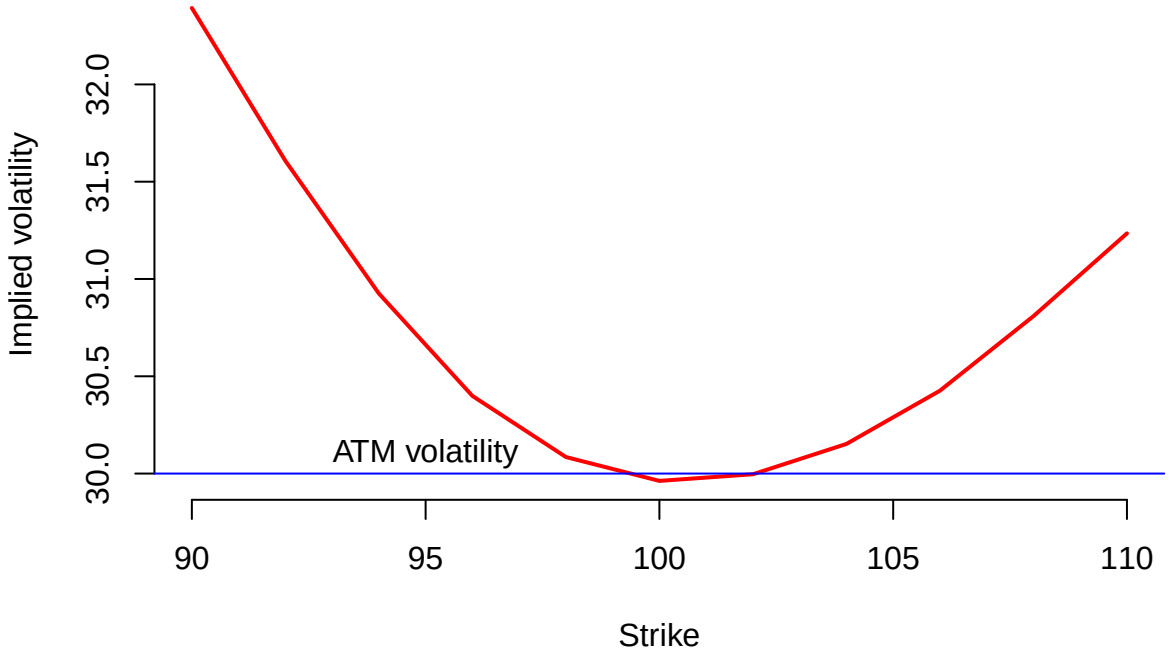

Figure 5: Implied volatility from the adjusted empirical density

Define the local risk R_k as:

$$E^P\left[(C_{k+1}(x_{k+1}) - C_k(x_k) - \phi_k(x_k)[x_{k+1} - x_k])^2\right]$$

where $E^P[]$ is the expectation under the objective probability measure.

We look for the pricing function $C_k(x)$ that minimizes the residual hedging risk.

The functions $C(x)$ and $\phi(x)$ are approximated by a set of basis functions:

$$C_k(x) = \sum_{a=1}^{M} \gamma_a^k C_a(x) \tag{8}$$

$$\phi_k(x) = \sum_{a=1}^{M} \gamma_a^k \frac{\partial C_a(x)}{\partial x} \tag{9}$$

Splines provide a convenient set of basis functions: given a set of knots $t_i, i = 1 \ldots, k$, the polynomial spline of degree n is defined by:

$$C(x) = \sum_{j=0}^{n} b_{0,j} x^j + \sum_{i=1}^{k} \sum_{j=0}^{n} b_{i,j} (x - t_i)_+^j$$

Thus, a spline of degree n with k knots is a linear combination of $m = (k+1)(n+1)$ basis functions. The derivative of $C(x)$ with respect to x is readily computed. To simplify notation, let:

$$C(x) = \sum_{a=1}^{m} \beta_a F_a(x)$$

$$C'(x) = \sum_{a=1}^{m} \beta_a F_a'(x)$$

where $F_a(x)$ are the elementary basis functions. At each step t in the hedged Monte-Carlo simulation, we obtain the price function by solving for β the following optimization problem (formulation for a call):

$$\min \sum_{l=1}^{N} [e^{-\rho} C_{t+1}(x_{t+1}^l) - \sum_{a=1}^{M} \beta_a (F_a(x_t^l) + F_a'(x_t^l)(x_{t+1}^l e^{-\rho} - x_t^l))]^2$$

A simple modification of the model enables us to account for the bid-ask spread on transactions. We assume that the price paths are mid-market prices, and that transactions are negotiated at the bid or ask price. Let ϵ be the bid-ask spread. The local risk becomes:

$$(C_{k+1}(x_{k+1}) - C_k(x_k) - \phi_k(x_k)(e^{-\rho} x_{k+1} - x_k - \delta\epsilon/2))^2$$

where

$$\delta = \begin{cases} -1 & (x_k - e^{-\rho} x_{k+1}) >= 0 \\ 1 & (x_k - e^{-\rho} x_{k+1}) < 0 \end{cases}$$

So far, we have only considered contracts with a single payoff at expiry. A simple extension of the model can accommodate arbitrary contingent cash flows at each step of the simulation. Assume that as each time step, the contract generates a cash flow $F(x_k)$. We then define the price function $C(x_k)$ as the contract value ex cash flow. The local risk function is then:

$$(C_{k+1}(x_{k+1}) + F_{k+1}(x_{k+1}) - C_k(x_k) + \phi_k(x_k)(x_k - e^{-\rho} x_{k+1} + \delta\epsilon/2))^2$$

With contingent cash flows at each period, the constraints defined by equation (**??**) need to be reformulated. Consider first the constraint that the value of the contract must be greater or equal to the intrinsic value of the European option. With multiple cash flows, the equivalent constraint is that the contract value at a given time step and state must be greater than the sum of the expected discounted cash flows, under the conditional probability of this time step and state. The rest of the algorithm is left unchanged.

2.2 Illustration

To demonstrate the usefulness of this algorithm, we reproduce some results from (**Marc2001?**). The first experiment compares the Hedged MC algorithm to the classical binomial model for pricing European options, and to a unhedged Monte-Carlo pricing method. The first step is togenerate geometric brownian paths:

```
GBMPathSimulator <-
  function(nbObs, nbSim, S0, mu, sigma, TTM, center = 'Mean') {
    delta.t <- TTM / nbObs
    if (center == 'Mean') {
      Z <- rnorm(nbObs * nbSim,mean = 0,sd = 1)
      dim(Z) <- c(nbObs, nbSim)
      Z <- Z - mean(Z)
    } else {
      Z <- rnorm(nbObs * nbSim / 2,mean = 0,sd = 1)
      dim(Z) <- c(nbObs, nbSim / 2)
      Z <- cbind(Z,-Z)
    }

    path = (mu - sigma * sigma / 2) * delta.t + sigma * sqrt(delta.t) * Z
    S <- S0 * rbind(rep(1, nbSim), exp(apply(path, 2, cumsum)))
    S
  }
```

For the sake of simplicity, the basis functions are truncated power functions with one knot located at the strike.

```
  basis <- function(k, S, K, truncate=FALSE) {
    truncated.S <- if(truncate) pmax(S - K, 0) else S
    F <- truncated.S ^ (k-1)
    F.prime <- (k-1)*truncated.S^(k-2)
    cbind(F, F.prime)
 }
```

The hedged Monte-Carlo algorithm is implemented as follows:

```
nb.paths <- 500  # number of simulated paths
M <- 8           # number of splines
X <- matrix(0, nrow=nb.paths, ncol=M+1)
Z <- matrix(0, nrow=nb.paths, ncol=M+1)
american.ex <- FALSE

hedged.mc <- function(payoff, K) {
  # continuation value at last time step is exercice value
  CV <- payoff(S[,(nb.steps+1)])
  t.last=1
  for (t in nb.steps:t.last) {
    if(american.ex) {
      exercice.value <- payoff(S[,t+1])
      CV <- pmax(CV, exercice.value)
    }
    # discounted continuation value
    disc.CV <- CV * df

    # compute all polynomial terms
    for(k in seq(M+1)) {
      tmp <- basis(k, S[,t], K, truncate=FALSE)
      C.alpha <- tmp[,1]
      F.alpha <- tmp[,2]
      X[, k] <- C.alpha + F.alpha * (S[, t+1]*df - S[, t])
      Z[, k] <- C.alpha
    }

    # Try polynomial regression until all coefficients OK

    reg.ok <- FALSE
    MM <- M
    MM.min <- if(t>1) 2 else 1
    while(!reg.ok & MM>=MM.min) {
      # add truncated term
      tmp <- basis(MM, S[,t], K, truncate=TRUE)
      C.alpha <- tmp[,1]
```

```
      F.alpha <- tmp[,2]
      X[, MM+1] <- C.alpha + F.alpha * (S[, t+1]*df - S[, t])
      Z[, MM+1] <- C.alpha

      # at the origin, the independent variable is the constant S.0
      if(t>1) {
        reg <- lm(disc.CV ~ X[,seq(MM+1)] -1, singular.ok = TRUE)
      } else {
        reg <- lm(disc.CV ~ X[,seq(MM)] -1, singular.ok = TRUE)
      }
      reg.ok <- (any(is.na(reg$coefficients)) == FALSE)
      if(!reg.ok) MM <- MM-1
    }
    if(!reg.ok) {
      stop(paste("Regression error at t:", t))
    }

    if(t>1) {
      CV <- Z[,seq(MM+1)] %*% matrix(reg$coefficients, ncol=1)
    } else {
      CV <- Z[,seq(MM)] %*% matrix(reg$coefficients, ncol=1)
    }
  }
 mean(CV)
}
```

The first experiment reported in Potters's paper involves a 3-months Call option, strike 100.

```
  K.call <- 100
  call.payoff <- function(S) pmax(S-K.call, 0)
```

The other parameters of the experiment are as follows:

```
S.0 <- 100    # spot
r <- 0.05     # risk-free interest rate
div <- 0      # dividend yield
TTM <- 1/4    # time to maturity, in years
sigma <- 0.30 # volatility

nb.steps <- 20   # number of time steps in simulation
nb.trials <- 500 # number of replications

dT <- TTM/nb.steps
# discount factor for one time step
df <- exp(-r*dT)
```

Table 1: Pricing an ATM European option maturity 3 months. $S_0 = 100$

	Black-Scholes	Unhedged MC	Hedged MC
Mean	6.58	6.57	6.56
SD	NA	0.23	0.08

We compare in Table 1 the exact Black-Scholes value of the option to the estimates obtained by unhedged Monte-Carlo simulations and by hedged simulations.

```
hedged.mc.price <- vector(mode="numeric", length=nb.trials)
unhedged.mc.price <- vector(mode="numeric", length=nb.trials)

for(i in seq(nb.trials)) {
  # simulated paths for the underlying asset
  S <- t(GBMPathSimulator(nb.steps, nb.paths, S.0, mu=r-div,

  unhedged.mc.price[i] <- mean(call.payoff(S[, nb.steps+1])) * exp(-r*TTM)
  hedged.mc.price[i] <- hedged.mc(call.payoff, K.call)
}

opt <- CRRBinomialTreeOption(TypeFlag="ce", S=S.0, X=K.call, Time=TTM, r=r,
                             b=r, sigma=sigma, n=200)@price
```

The results show a remarkable consistency between the Hedged MC method and the Black-Scholes value, with a standard deviation reduced by x% with respect to the unhedged MC algorithm.

The true value of the Hedged MC algorithm is its ability to price an asset under a density that is not risk-neutral. This is demonstrated by the next experiment, where the drift of the geometric brownian motion is set to 30%. Everything else is left as in the previous experiment:

```
drift <- 0.3
for(i in seq(nb.trials)) {
  # simulated paths for the underlying asset
  S <- t(GBMPathSimulator(nb.steps, nb.paths, S.0, mu=drift-div,

  unhedged.mc.price[i] <- mean(call.payoff(S[, nb.steps+1])) * exp(-r*TTM)
  hedged.mc.price[i] <- hedged.mc(call.payoff, K.call)
}
```

The results in Table 2 show that the Hedged MC correctly prices the option under a geometric brownian motion with an arbitrary drift.

In order to evaluate an American put option, the estimated value $C_{k+1}(x_{k+1})$ is replaced by $\max(C_{k+1}(x_{k+1}), K - x_{k+1})$, where K is the strike.

Table 2: Pricing an ATM European option when the scenarios are not risk-neutral. $S_0 = 100$

	Black-Scholes	Unhedged MC	Hedged MC
Mean	6.58	10.73	6.55
SD	NA	0.25	0.08

Following Potters's paper, the other parameters are as follows:

```
S.0 <- 40     # spot
r <- 0.06     # risk-free interest rate
div <- 0      # dividend yield
TTM <- 1      # time to maturity, in years
sigma <- 0.20 # volatility

nb.steps <- 20   # number of time steps in simulation
nb.paths <- 500  # number of simulated paths
nb.trials <- 500 # number of replications

K.put <- 40
put.payoff <- function(S) pmax(K.put-S, 0)

dT <- TTM/nb.steps
# discount factor for one time step
df <- exp(-r*dT)
```

```
american.ex <- TRUE
for(i in seq(nb.trials)) {
  # simulated paths for the underlying asset
  S <- t(GBMPathSimulator(nb.steps, nb.paths, S.0, mu=r-div,

  hedged.mc.price[i] <- hedged.mc(put.payoff, K.put)
}

opt <- CRRBinomialTreeOption(TypeFlag="pa", S=S.0, X=K.put, Time=TTM, r=r,
                             b=r, sigma=sigma, n=200)@price
```

Table 3 summarizes the results. The estimated price by hedged MC is higher than reported, but qualitatively consistent with the binomial price.

References

Jackwerth, J. C. (1999). Option-Implied Risk-Neutral Distribution and Implied Binomial Trees: A Literature Review. *The Journal of Derivatives*, 66–82.

Table 3: Pricing an ATM American option, maturity 1 year. $S_0 = 40$

	Binomial	Hedged MC
Mean	2.32	2.35
SD	NA	0.03

Zou. (1999). Strike-Adjusted Spread: A New Metric for Estimating the value of Equity Options. *Quantitative Strategies Research Note, Goldman Sachs*.