Volatility Calibration with American Options

Yves Achdou*
Govindaraj Indragoby †
Olivier Pironneau ‡

January 21, 2005

Abstract

In this paper, we present two methods in order to calibrate the local volatility with American put options. Both calibration methods use a least-square formulation and a descent algorithm. Pricing is done by solving parabolic variational inequalities, for which solution procedures by active set methods are discussed.
The first strategy consists in computing the optimality conditions and the descent direction needed by the optimization loop. This approach has been implemented both at the continuous and discrete levels. It requires a careful analysis of the underlying variational inequalities and of their discrete counterparts. In the numerical example presented here (American options on the FTSE index), the squared volatility is parameterized by a bicubic spline.
In the second approach, which works in low dimension, the descent directions are computed with Automatic Differentiation of computer programs implemented in C++.

Keywords: American options, variational inequalities, calibration of volatility, inverse problems, optimality conditions, finite element methods, automatic differentiation.

1 Introduction

The Black-Scholes model involves a risky asset and a risk-free asset whose price at time t is $S_t^0 = e^{rt}$, where r is the interest rate; it assumes that the price of the risky asset is a solution to the following stochastic differential equation,

$$dS_t = S_t(\mu dt + \sigma_t dB_t), \tag{1}$$

*UFR Mathématiques, Université Paris 7, Case 7012, 75251 PARIS Cedex 05, France and Laboratoire Jacques-Louis Lions, Université Paris 6. achdou@math.jussieu.fr
†Mathematics dept. University of Pondicherry, Tamil Nadu, India
‡Laboratoire Jacques-Louis Lions, Université Pierre et Marie Curie, Boˆite courrier 187, 75252 Paris Cedex 05. France. pironneau@ann.jussieu.fr

where W_t is a standard Brownian motion on a probability space $(\Omega, \mathcal{A}, \mathbb{P})$. Here σ_t is a positive number, called the *volatility*. In what follows, it will be convenient to work with the squared volatility $\eta_t = \sigma_t^2$.

A European option on the underlying risky asset, is a contract which permits to its owner a benefit $P_\circ(S_T)$ at time T The function $P_\circ$ is called the payoff function and the date T is the maturity.
With the Black-Scholes assumptions, it is possible to prove that the option's price at time t is given by

$$P_t = P_e(S_t, t) \equiv \mathbb{E}^*(e^{-r(T-t)} P_\circ(S_T)|F_t), \tag{2}$$

where the expectation $\mathbb{E}^*$ is taken with respect to the so-called risk-neutral probability $\mathbb{P}^*$ (equivalent to $\mathbb{P}$ and under which $dS_t = S_t(rdt + \sqrt{\eta_t} dW_t)$, W_t being a standard Brownian motion under $\mathbb{P}^*$ and F_t being the natural filtration of W_t). It can be seen that the pricing function P_e solves the parabolic partial differential equation:

$$\frac{\partial P_e}{\partial t} + \frac{\eta S^2}{2} \frac{\partial^2 P_e}{\partial S^2} + rS \frac{\partial P_e}{\partial S} - rP_e = 0. \tag{3}$$

In contrast with European options, American options can be exercised any time before maturity: *An American vanilla call (resp. put) option is a contract giving its owner the right to buy (resp. sell) a share of a specific common stock at a fixed price K before a certain date T*. More generally, for a payoff function $P_\circ$, the American option with payoff $P_\circ$ and maturity T can be exercised at any $t < T$, yielding the payoff $P_\circ(S_t)$.
Using the notion of strategy with consumption, the Black-Scholes model leads to the following formula for pricing an American option with payoff $P_\circ$: under the risk neutral probability,

$$P_t = P(S_t, t) \equiv \sup_{\tau \in \mathcal{T}_{t,T}} \mathbb{E}^* \left(e^{-r(\tau - t)} P_\circ(S_\tau) \middle| F_t \right), \tag{4}$$

where $\mathcal{T}_{t,T}$ denotes the set of stopping times in $[t, T]$ (see [17] for the proof of this formula). It can be seen that for an American vanilla call on a non dividend paying stock, the formula (4) coincides with (2), so American and European vanilla calls have the same price. This means that an American vanilla call should not be exercised before maturity.
It can be shown that the price P of the American put of strike K and maturity T is given as a solution to

$$\begin{array}{l} \frac{\partial P}{\partial t} + \frac{\eta S^2}{2} \frac{\partial^2 P}{\partial S^2} + rS \frac{\partial P}{\partial S} - rP \leq 0, \qquad P(t, S) \geq P_\circ(S), \quad t \in [0, T),\ S > 0, \\ (\frac{\partial P}{\partial t} + \frac{\eta S^2}{2} \frac{\partial^2 P}{\partial S^2} + rS \frac{\partial P}{\partial S} - rP)(P - P_\circ) = 0 \quad t \in [0, T),\ S > 0, \\ P(t = T, S) = P_\circ(S), \quad S > 0, \end{array} \tag{5}$$

where

$$P_\circ(S) = (K - S)_+.$$

The volatility is the difficult parameter of the Black-Scholes model. It is convenient to take it to be constant but then the computed options' prices do not match the market prices. Conversely, taking a family of options available on the market and inverting for each of them the Black-Scholes formula does not yield a constant volatility: for each option, one obtains a different *implied* volatility, and the implied volatility is often a convex function of the strike K, which is known in finance as the *smile effect*.

There are essentially three ways to improve on the Black-Scholes model with a constant volatility:

- *Use a local volatility*, i.e. assume that the volatility is a function of time and of the stock price. Then one has *to calibrate the volatility* from the market data, i.e. to find a volatility function which permits to recover the prices of the options available on the market.
- *assume that the volatility is itself a stochastic process*, see for example [11, 8].
- *generalize the Black-Scholes model by assuming that the spot price is for example a Lévy process*, see [6] and references therein.

In this paper, we deal with the first approach: the calibration problem consists in finding $\eta(S,t)$ from the observations of

- the spot price $S_\circ$ today,
- the prices $(\bar{P}_i)_{i\in I}$ of a family of options with different maturities and different strikes $(T_i,K_i)_{i\in I}$.

It has been observed by Dupire,[7], that fixing $t=0$ and $S=S_\circ$, the price of a European vanilla call with maturity τ and strike K: $P(S_\circ,0,K,\tau)$ as a function of τ and K satisfies

$$\partial_\tau P - \frac{1}{2}\eta(K,\tau)K^2\frac{\partial^2 P}{\partial K^2} + rK\frac{\partial P}{\partial K} = 0, \text{ with } P(K,0) = (K-S_\circ)_-, \tag{6}$$

If the options for all strikes and maturities where on the market, the local squared volatility would be:

$$\eta(K,\tau) = 2\frac{\partial_\tau P(K,\tau) + rK\frac{\partial P}{\partial K}(K,\tau)}{K^2\frac{\partial^2 P}{\partial K^2}(K,\tau)}. \tag{7}$$

Hence, it is tempting to solve the calibration problem by taking first a smooth K,τ-interpolation P of $(\bar{P}_i)_{i\in I}$ at $K_i,T_i)$, and then by using (7). The problem with this strategy is that it is unstable (a small change in $\bar{P}_i$ can produce a large change in η) and also it is not possible to constrain η to stay in an interval of sensible values (in particular to be positive). Nevertheless the argument above shows that most likely, there are infinitely many solutions to the calibration problem with European options. A more stable way to solve the calibration problem is to

$$\text{find } \eta\in\mathcal{H} \text{ minimizing} \quad J(\eta)+J_R(\eta), \qquad J(\eta)=\sum_{i\in I}|P(S_\circ,0,K_i,T_i)-\bar{P}_i|^2, \tag{8}$$

where $\mathcal{H}$ is a suitable closed subset of a possibly infinite dimensional function space, J_R is a suitable Tychonoff regularization functional, and where $P(S_\circ,0,K_i,T_i)$ is the price of the option with strike K_i and maturity T_i computed with the local volatility η.
Note that if $\mathcal{H}$ is not carefully chosen and if $J_R=0$, then the problem is unstable too. Note also that computing $J(\eta)$ requires solving $\#(I)$ different problems of the type (3) for European options or (5) for American options, so this approach is expensive.
There has been a number of valuable studies on the calibration of volatility with European options of which it is difficult to make a complete account here. In Lagnado and Osher [15, 16] a least square method is used, the volatility is discretized by splines using matlab and the computation of the gradient of J with respect to η is done either

by numerical differences or by Adol-C. In Achdou et al [2] , calibration with European option is made easier by using Dupire's equation. An alternative to least squares is the pioneering method by Avellaneda et al [5] based on the maximization of an entropy function via dynamic programming. Up to our knowledge, calibration with American options has not been very much discussed yet in the mathematical finance literature.

The paper is divided in five parts: we first give theoretical results on the variational inequality for pricing American options and on the free boundary which delimitates the region of exercise. The second part of the paper is devoted to the finite element method for pricing American options. We show in particular that under some assumptions, there is a free boundary in the discrete problem too, and we discuss two algorithms for computing the price of the American option, both based on active sets strategies. The third part is devoted to volatility calibration with American options: we consider problem (8) with

$$J_R(\eta) = \int_0^T \int_0^{\bar{S}} a(S\partial_S\eta)^2 + b(\partial_t\eta)^2 + c(S\partial^2_{tS}\eta)^2 + d(S^2\partial^2_{SS}\eta)^2 + e(\eta - \eta_g)^2.$$

The minimization problem is constrained, and optimality conditions are found for (8); it is also proved that differentiability with respect to η holds provided a strict complementarity condition is satisfied. We give an example of calibration with American options on the FTSE index, where the squared volatility is discretized with bicubic splines. The results contained in the four first parts are proved in [1, 3, 4]. In the last part of the paper, a suitable parameterization of the volatility leads to an unconstrained optimization problem, and automatic differentiation of computer codes is used in order to find descent direction in the optimization algorithm. The technique of automatic differentiation is particularly easy to implement and very accurate in most cases. However, it cannot be used in large dimensions.

2 The variational inequality and the free boundary

All the proofs of the results below can be found in [1].
Calling t the time to maturity, the problem becomes

$$\begin{array}{rcl} \dfrac{\partial P}{\partial t} - \dfrac{\eta(S,t)S^2}{2}\dfrac{\partial^2 P}{\partial S^2} - rS\dfrac{\partial P}{\partial S} + rP & \geq & 0, \quad P \geq P_\circ, \\ (\dfrac{\partial P}{\partial t} - \dfrac{\eta(S,t)S^2}{2}\dfrac{\partial^2 P}{\partial S^2} - rS\dfrac{\partial P}{\partial S} + rP)(P - P_\circ) & = & 0, \end{array} \tag{9}$$

with Cauchy data

$$P|_{t=0} = P_\circ. \tag{10}$$

We focus on the case of a vanilla put, i.e. the payoff function is $P_\circ(S) = (K - S)_+$.
To write the variational formulation of (9) (10), we need to use the Sobolev space

$$V = \{v \in L^2(\mathbb{R}_+) : \ S\frac{dv}{dS} \in L^2(\mathbb{R}_+)\}, \tag{11}$$

and we call $\mathcal{K}$ the subset of V:

$$\mathcal{K} = \{v \in V, v \geq P_\circ \text{ in } \mathbb{R}_+\}. \tag{12}$$

Since the function of V are continuous, the inequality in (12) has a pointwise meaning. The set $\mathcal{K}$ is a closed and convex subset of V, because convergence in V implies pointwise convergence. We introduce the bilinear form a_t:

$$\begin{aligned} a_t(v,w) = &\int_{\mathbb{R}_+} \frac{S^2\eta(S,t)}{2}\frac{\partial v}{\partial S}\frac{\partial w}{\partial S}\,dS \\ &+ \int_{\mathbb{R}_+}\left(-r(t)+\eta(S,t)+\frac{S}{2}\frac{\partial \eta}{\partial S}(S,t)\right)S\frac{\partial v}{\partial S}w\,dS \\ &+ r\int_{\mathbb{R}_+} vw\,dS \end{aligned} \tag{13}$$

We make the assumptions: there exist two positive constants, $\underline{\eta}$ and $\overline{\eta}$ such that for all $t \in [0,T]$ and all $S \in \mathbb{R}_+$,

$$0 < \underline{\eta} \leq \eta(S,t) \leq \overline{\eta}. \tag{14}$$

There exists a positive constant C_η such that for all $t \in [0,T]$ and all $S \in \mathbb{R}_+$,

$$|S\frac{\partial \eta}{\partial S}(S,t)| \leq C_\eta. \tag{15}$$

These imply that the bilinear form a_t is continuous on V uniformly in t, and Gårding's inequality : for a non negative constant λ depending only on $\bar{\eta}$, $\underline{\eta}$ and C_η,

$$a_t(v,v) \geq \frac{\eta}{4}|v|_V^2 - \lambda\|v\|^2_{L^2(\mathbb{R}_+)}, \quad \forall v \in V. \tag{16}$$

The weak form of (9) is to

find $P \in \mathcal{C}^0([0,T];L^2(\mathbb{R}_+)) \cap L^2(0,T;\mathcal{K})$ such that $\frac{\partial P}{\partial t} \in L^2(0,T;V')$, satisfying $P_{|t=0} = P_\circ$, and

$$\forall v \in \mathcal{K}, \quad \left(\frac{\partial P}{\partial t}(t), v - P(t)\right) + a_t(P(t), v - P(t))) \geq 0. \tag{17}$$

Theorem 1 *With η satisfying assumptions (14) and (15), the problem (17) has a unique solution P which belongs to $\mathcal{C}^0([0,T]\times[0,+\infty))$ with $P(0,t) = K, \forall t \in [0,T]$, and is such that*
$S\frac{\partial P}{\partial S}, \frac{\partial P}{\partial S} \in L^2(0,T;V)$, $S\frac{\partial P}{\partial S} \in C^0([0,T];L^2(\mathbb{R}_+))$ and $\frac{\partial P}{\partial t} \in L^2(0,T;L^2(\mathbb{R}_+))$.
The function P is also greater than or equal to P_e, the price of the vanilla European put.
The quantities $\|P\|_{L^2(0,T;V)}$, $\|P\|_{L^\infty(0,T;L^2(\mathbb{R}_+))}$, $\|S\frac{\partial P}{\partial x}\|_{L^2(0,T;V)}$, $\|\frac{\partial P}{\partial S}\|_{L^2(0,T;V)}$, $\|S\frac{\partial P}{\partial S}\|_{L^\infty(0,T;L^2(\mathbb{R}_+))}$, $\|\frac{\partial P}{\partial t}\|_{L^2(0,T;L^2(\mathbb{R}_+))}$, are bounded by constants depending only on K, $\bar{\eta}$, $\underline{\eta}$ and C_η.
We have that

$$-1 \leq \frac{\partial P}{\partial S} \leq 0, \quad \forall t \in (0,T],\ a.a.\ S > 0. \tag{18}$$

There exists a function $\gamma : (0,T] \to [0,K)$, such that $\forall t \in (0,T)$, $\{S$ s.t. $P(S,t) = P_\circ(S)\} = [0,\gamma(t)]$. The function γ is upper semi-continuous, right continuous in $[0,T)$, and, for each $t \in (0,T]$, γ has a left-limit at t.
Calling μ the function $\mu = \frac{\partial P}{\partial t} + A_t P$, where A_t is the linear operator: $V \to V'$; for all $v, w \in V$, $A_t v = -\frac{\eta(S,t)S^2}{2}\frac{\partial^2 v}{\partial S^2} - rS\frac{\partial v}{\partial S} + rv$, we have

$$\mu = rK1_{\{P=P_\circ\}}. \tag{19}$$

In other words, a.e., one of the two conditions $P = P_\circ$ and $\mu = 0$ is not satisfied: there is strict complementarity in (9).
Finally, there exists $\gamma_0 > 0$ depending only on $\bar{\eta}$ and K such that

$$\gamma(t) \geq \gamma_0, \quad \forall t \in [0, T]. \tag{20}$$

3 Pricing American options with a finite element method

3.1 A finite element method

All the proof of the results below are given in [3].
We localize the problem on $(0, \bar{S})$, so V becomes

$$V = \{v \in L^2((0, \bar{S}); S\frac{\partial v}{\partial S} \in L^2((0, \bar{S}); v(\bar{S}) = 0\}$$

(where $\bar{S}$ is large enough so that $P_\circ(\bar{S}) = 0$), and $\mathcal{K} = \{v \in V, v \geq P_\circ\}$. The variational inequality is (17) with new meanings for V, $\mathcal{K}$, and a_t.
Moreover, if $\gamma_0 \in (0, K)$ as in (20) is known, one can focus on the smaller interval $[\underline{S}, \bar{S}]$, with $0 \leq \underline{S} < \gamma_0$ and obtain the equivalent weak formulation:

$$\text{find } P \in L^2((0, T, \mathcal{K}) \cap \mathcal{C}^0([0, T]; L^2(\Omega)), \text{ with } \frac{\partial P}{\partial t} \in L^2(0, T; V')$$

such that $P(t = 0) = P_\circ$ and (17)for all $v \in \mathcal{K}$, with the new definition of the closed set $\mathcal{K}$:

$$\mathcal{K} = \{v \in V, v \geq P_\circ \text{ in } (0, \bar{S}], P = P_\circ \text{ in } (0, \underline{S}]\}. \tag{21}$$

We introduce a partition of the interval $[0, T]$ into subintervals $[t_{n-1}, t_n]$, $1 \leq n \leq N$, with $\Delta t_i = t_i - t_{i-1}$, $\Delta t = \max_i \Delta t_i$ and a partition of the interval $[0, \bar{S}]$ into subintervals $\omega_i = [S_{i-1}, S_i]$, $1 \leq i \leq N_h + 1$, such that $0 = S_0 < S_1 < \cdots < S_{N_h} < S_{N_h+1} = \bar{S}$. The size of the interval ω_i is called h_i and we set $h = \max_{i=1,\ldots,N_h+1} h_i$. The mesh $\mathcal{T}_h$ of $[0, \bar{S}]$ is the set $\{\omega_1, \ldots, \omega_{N_h+1}\}$. In what follows, we will assume that both the strike K and the real number $\underline{S}$ coincide with nodes of $\mathcal{T}_h$: there exist $\alpha < \kappa$, $0 \leq \alpha < \kappa < N_h + 1$ such that $S_\kappa = K$ and $S_{\alpha-1} = \underline{S}$. We define the discrete space V_h by

$$V_h = \left\{v_h \in V, \ \forall \omega \in \mathcal{T}_h, v_{h|\omega} \in \mathcal{P}_1(\omega)\right\}, \tag{22}$$

where $\mathcal{P}_1(\omega)$ is the space of linear functions on ω.
Since K is a node of $\mathcal{T}_h$, $P_\circ \in V_h$, and since $\underline{S}$ is also a node of $\mathcal{T}_h$, we can define the closed subset $\mathcal{K}_h$ of V_h by

$$\begin{aligned} \mathcal{K}_h &= \{v \in V_h, \ v \geq P_\circ \text{ in } [0, \bar{S}), \quad v = P_\circ \text{ in } [0, \underline{S}]\} \\ &= \{v \in V_h, \ v(S_i) \geq P_\circ(S_i), \ i = 0, \ldots, N_h + 1, \ v(S_i) = P_\circ(S_i), \ i < \alpha\}. \end{aligned} \tag{23}$$

The discrete problem arising from an implicit Euler scheme is:

$$\text{find } (P^n)_{0 \leq n \leq N} \in \mathcal{K}_h \text{ satisfying}$$

$$P^0 = P_\circ, \tag{24}$$

$$\text{and for all } n, \ 1 \leq n \leq N,$$

$$\forall v \in \mathcal{K}_h, \quad \left(P^n - P^{n-1}, v - P^n\right) + \Delta t_n a_{t_n}(P^n, v - P^n) \geq 0. \tag{25}$$

Consider λ such that Gårding's inequality (16) holds, and take $\Delta t < \frac{1}{\lambda}$, there exists a unique P^n satisfying (25).
Let $(w^i)_{i=0,\dots N_h}$ be the nodal basis of V_h, and let $\mathbf{M}$ and $\mathbf{A}^m$ in $\mathbb{R}^{(N_h+1)\times(N_h+1)}$ be the mass and stiffness matrices defined by

$$\mathbf{M}_{i,j} = (w^i, w^j), \mathbf{A}^m_{i,j} = a_{t_m}(w^j, w^i), \quad 0 \le i, j \le N_h.$$

Calling

$$U^n = (P^n(S_0), \dots, P^n(S_{N_h}))^T \text{ and } U^0 = (P_\circ(S_0), \dots, P_\circ(S_{N_h}))^T,$$

(25) is equivalent to

$$\begin{array}{rcll} (\mathbf{M}(U^n - U^{n-1}) + \Delta t_n \mathbf{A}^n U^n)_i & \ge & 0, & \text{for } i \ge \alpha, \\ U^n_i & = & U^0_i & \text{for } i < \alpha, \\ U^n & \ge & U^0, & \\ (U^n - U^0)^T(\mathbf{M}(U^n - U^{n-1}) + \Delta t_n \mathbf{A}^n U^n) & = & 0. & \end{array} \tag{26}$$

We call $\mathbf{M}_\alpha$, respectively $\mathbf{A}^n_\alpha$, the block of $\mathbf{M}$, respectively $\mathbf{A}^n$, corresponding to $\alpha \le i, j \le N_h$.

3.2 The discrete exercise boundary

One may ask if there is a well defined exercise boundary $t \to \gamma_h(t)$ also in the discrete problem. A positive answer has been given by Jaillet et al [14] in the case of a constant volatility, an implicit Euler scheme and a uniform mesh in the logarithmic variable. The main argument of the proof lies in the fact that the solution to the discrete problem is non decreasing with respect to the variable t. With a local volatility, this may not hold (see the numerical example below). The result of Jaillet et al has been completed for a local volatility in [3], in the special case when the mesh is uniform in the variable S: here too, the discrete problem has a free boundary. The proof does not rely any longer on the monotonic character of the discrete solution with respect to t but on the discrete analogue of the bounds (18), i.e. $-1 \le \frac{\partial P}{\partial S} \le 0$. This is proved by studying a suitable penalized problem (which is the discrete version of a semilinear parabolic equation with a non decreasing and convex non linearity) and by using a discrete maximum principle on the partial derivative with respect to S (for this reason, a uniform mesh is needed). We can summarize this by

Theorem 2 *Let η verify (14) and (15), and choose $\Delta t < \frac{1}{2\lambda}$, with λ given in (16). Assume that the grid $\mathcal{T}_h$ is uniform and that $\underline{S} > 0$. Assume also that the parameters h and $\frac{h^2}{\Delta t}$ are small enough so that the matrices $\mathbf{A}^n_\alpha$ and $\mathbf{M}_\alpha + \Delta t_n \mathbf{A}^n_\alpha$ are tridiagonal irreducible M-matrices for all n, $1 \le n \le N$.*
There exist N real numbers γ^n_h, $1 \le n \le N$, such that

$$\begin{array}{l} \underline{S} \le \gamma^n_h < K, \\ \gamma^n_h \text{ is a node of } \mathcal{T}_h, \\ \forall i, 0 \le i \le N_h, \quad P^n(S_i) = P_\circ(x_i) \Leftrightarrow S_i \le \gamma^n_h. \end{array} \tag{27}$$

We believe that this may be extended to somewhat more general meshes.

3.3 A front-tracking algorithm

Here, we propose an algorithm for computing the solution of (25) assuming that the free boundary is the graph of a function. In our experience, this algorithm, based on tracking the free boundary, is more robust (and slightly more expensive) than the Brennan and Schwartz algorithm (see [14]). Since the free boundary is the graph of a function, the idea is to look for γ_h^n by

- Start from $\gamma_h^n = \gamma_h^{n-1}$,
- solve the discrete problem corresponding to

$$\frac{P^n - P^{n-1}}{\Delta t_n} - \frac{\eta(S,t_n)S^2}{2}\frac{\partial^2 P^n}{\partial S^2} - rS\frac{\partial P^n}{\partial S} + rP^n = 0 \text{ for } \gamma_h^n < S < \bar{S},$$
$$P^n = P_\circ \text{ for } 0 \le S \le \gamma_h^n,$$

  and $P^n(\bar{S}) = 0$,
- if P^n satisfies (25), stop else shift the point γ_h^n to the next node on the mesh left/right according to which constraint is violated by P^n.

With the notations introduced above, the algorithm for computing P_h^n is as follows:

Algorithm
choose k such that $\gamma_h^{n-1} = S_k$; set found=false;
while(**not** found)
.. solve

$$\begin{aligned} (\mathbf{M}(U^n - U^{n-1}) + \Delta t_n \mathbf{A}^n U^n)_i &= 0, \quad \text{for } i \ge k, \\ U_i^n &= U_i^0 \quad \text{for } i < k. \end{aligned} \tag{28}$$

.. **if** $((U^n - U^0)_{k+1} < 0\ )$
.. found=false; $k = k+1$;
.. **else**
.. compute $a = (\mathbf{M}(U^n - U^{n-1}) + \Delta t_n \mathbf{A}^n U^n)_{k-1}$;
.. **if** $(a < 0)$
.. found=false; $k = k-1$;
.. **else** found=true

In our tests, we have computed the average (over the time steps) number of iterations to obtain the position of the free boundary: it was found that (with a rather fine time-mesh), this number is smaller than 2.

3.4 A regularized active set strategy

The algorithm above is not easy to generalize in higher dimensions. For an algorithm based on active sets and generalizable in any dimension, we have to regularize first the problem. Following [13], we first go back to the semi-discrete problem: find $P^n \in \mathcal{K}$ such that

$$\forall v \in \mathcal{K}, \quad \left(P^n - P^{n-1}, v - P^n\right) + \Delta t_n a_{t_n}(P^n, v - P^n) \ge 0.$$

For any positive constant c, this is equivalent to finding $P^n \in V$ and a Lagrange multiplier $\mu \in V'$ such that

$$\begin{aligned} &\forall v \in V, \quad \left(\frac{P^n - P^{n-1}}{\Delta t_n}, v\right) + a_{t_n}(P^n, v) - \langle \mu, v \rangle = 0, \\ &\mu = \max(0, \mu - c(P^n - P^0)). \end{aligned} \tag{29}$$

When using an iterative method for solving (29), i.e. when constructing a sequence $(P^{n,m}, \mu^m)$ for approximating (P^n, μ), the Lagrange multiplier μ^m may not be a function if the derivative of the $P^{n,m}$ jumps, whereas μ is generally a function. Therefore, a dual method (i.e. an iterative method for computing μ) may be difficult to use. As a remedy, K.Ito and K.Kunisch [13] consider a one parameter family of regularized problems based on smoothing the equation for μ by

$$\mu = \alpha \max(0, \mu - c(P^n - P^0)), \tag{30}$$

for $0 < \alpha < 1$, which is equivalent to

$$\mu = \max(0, -\chi(P^n - P^0)), \tag{31}$$

for $\chi = c\alpha/(1-\alpha) \in (0, +\infty)$. We may consider a generalized version of (31):

$$\mu = \max(0, \bar{\mu} - \chi(P^n - P^0)), \tag{32}$$

where $\bar{\mu}$ is a fixed function. This turns to be useful when the complementarity condition is not strict.
It is now possible to study the fully regularized problem

$$\begin{aligned} &\forall v \in V, \quad \left(\frac{P^n - P^{n-1}}{\Delta t_n}, v\right) + a_{t_n}(P^n, v) - \langle \mu, v \rangle = 0, \\ &\mu = \max(0, \bar{\mu} - \chi(P^n - P^0)), \end{aligned} \tag{33}$$

and prove that it has a unique solution, with μ a square integrable function. A primal-dual active set algorithm for solving (33) is

Primal-dual active set algorithm

1. Choose $P^{n,0}$, set $k = 0$
2. Loop
   (a) Set
   $$\mathcal{A}^{-,k+1} = \{S : \bar{\mu}^k(S) - \chi(P^{n,k}(S) - P^0(S)) > 0\}$$
   and $\mathcal{A}^{+,k+1} = (0, \bar{S}) \backslash \mathcal{A}^{-,k+1}$.
   (b) Solve for $P^{n,k+1} \in V$: $\forall v \in V$,
   $$\left(\frac{P^{n,k+1} - P^{n-1}}{\Delta t_n}, v\right) + a_{t_n}(P^{n,k+1}, v) - (\bar{\mu} - \chi(P^{n,k+1} - P^0), 1_{\mathcal{A}^{-,k+1}} v) = 0. \tag{34}$$
   (c) Set
   $$\mu^{k+1} = \begin{cases} 0 & \text{on} \quad \mathcal{A}^{+,k+1}, \\ \bar{\mu} - \chi(P^{n,k+1} - P^0) & \text{on} \quad \mathcal{A}^{-,k+1} \end{cases} \tag{35}$$

(d) Set $k = k + 1$.

Calling A_n the operator from V to V': $\langle A_n v, w\rangle = \left(\frac{v}{\Delta t_n}, w\right) + a_{t_n}(v, w)$ and $F : V \times L^2(\mathbb{R}_+) \to V' \times L^2(\mathbb{R}_+)$

$$F(v,\mu) = \begin{pmatrix} A_n v + \mu - \frac{P^{n-1}}{\Delta t_n} \\ \mu - \max(0, \bar{\mu} - \chi(v - P^0)) \end{pmatrix},$$

it is proved in [13] that $G(v,\mu) : V \times L^2(\mathbb{R}_+) \to V' \times L^2(\mathbb{R}_+)$ defined by

$$G(v,\mu)h = \begin{pmatrix} A_n h_1 + h_2 \\ h_2 - \chi 1_{\{\bar{\mu} - \chi(v - P^0) > 0\}} h_1 \end{pmatrix}$$

is a generalized derivative of F in the sense that

$$\lim_{\|h\| \to 0} \frac{\|F(v + h_1, \mu + h_2) - F(v,\mu) - G(v + h_1, \mu + h_2)h\|}{\|h\|} = 0;$$

Note that

$$G(P^{n,k}, \mu^k)h = \begin{pmatrix} A_n h_1 + h_2 \\ h_2 - \chi 1_{\mathcal{A}^{-,k+1}} h_1 \end{pmatrix}.$$

Thus the primal-dual active set algorithm above can be seen as a semi-smooth Newton method applied to F, i.e.

$$(P^{n,k+1}, \mu^{k+1}) = (P^{n,k}, \mu^k) + G^{-1}(P^{n,k}, \mu^k) F(P^{n,k}, \mu^k). \tag{36}$$

Indeed, calling $(\delta P^n, \delta\mu) = (P^{n,k+1} - P^{n,k}, \mu^{k+1} - \mu^k)$, it is straightforward to see that in the primal-dual active set algorithm, we have

$$\begin{aligned} A_n \delta P^n + \delta\mu &= -A_n P^{n,k} - \mu^k + \frac{P^{n-1}}{\Delta t_n}, \\ \delta\mu &= -\mu^k \text{ on } \mathcal{A}^{+,k+1}, \\ \delta\mu - \chi\delta P^n &= -\mu^k + \bar{\mu} - \chi(P^{n,k} - P^0) \text{ on } \mathcal{A}^{-,k+1}, \end{aligned}$$

which is precisely (36).
In [13], Ito and Kunish, by using the results proved in [12], establish that the primal-dual active set algorithm converges from any initial guess, and that if the initial guess is sufficiently close to the solution of (33), then the convergence is superlinear.
To compute numerically the solution of (29), it is possible to compute successively the solutions $(P^n(\chi_\ell), \mu(\chi_\ell))$ of (33) for a sequence of parameters (χ_ℓ) converging to $+\infty$: to compute $(P^n(\chi_{\ell+1}), \mu(\chi_{\ell+1}))$, one uses the primal-dual active set algorithm with initial guess $(P^n(\chi_\ell), \mu(\chi_\ell))$.

Notice that it is possible to use the same algorithm for the fully discrete problem. Convergence results hold in the discrete case if there is a discrete maximum principle. The algorithm amounts to solving a sequence of systems of linear equations, and the matrix of the system varies at each iteration.

3.5 Mesh adaption

One of the key features of the finite element method is that it permits to compute reliable and often very efficient error indicators for the error between the exact and

discrete solutions. We do not wish to develop much on this topic here, because this needs long and technical arguments. We rather refer to [4]. The strategy relies on the fact that the error due to the time discretization can be estimated separately: there are indicators for the error due to time discretization, and at each time step, indicators for the error due to the discretization with respect to the price variable. The error indicators are local. Therefore, they tell us where the time grid, and the mesh in the S variable should be refined.
To illustrate this, we consider an American put, with strike $K = 100$. The interest rate is 0.04 as above, but the volatility is local and we choose:

$$\sigma(S,t) = 0.1 + 0.1 * 1_{100(t-0.5)^2 + \frac{(S-90)^2}{100} < 2}(S,t),$$

so the volatility is piecewise constant and takes the value 0.2 in an ellipse and 0.1 outside. With such a choice, the exercise boundary is expected to change slope as it enters and comes out the region where $\sigma = 0.2$. On Figure 1, we plot the volatility surface as a function of S and t. The exercise boundary is displayed on Figure 3: we see that the free boundary does change slope when the volatility jumps. We see also that refinement is crucial in order to catch properly the exercise boundary. Note that the function γ is not monotone. On Figure 2, two meshes at displayed: we see that the refinement follows the free boundary. On Figure 4, the error indicators with respect to S are plotted: here again, we see that the error indicators are large near the free boundary, where the function P is singular.

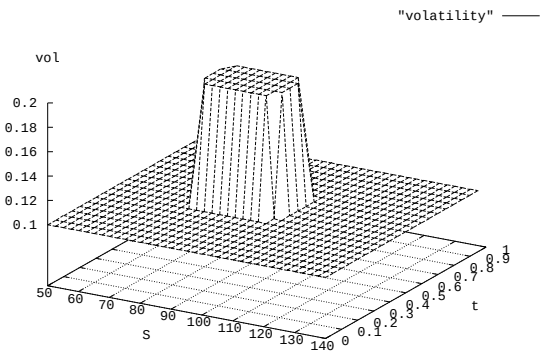
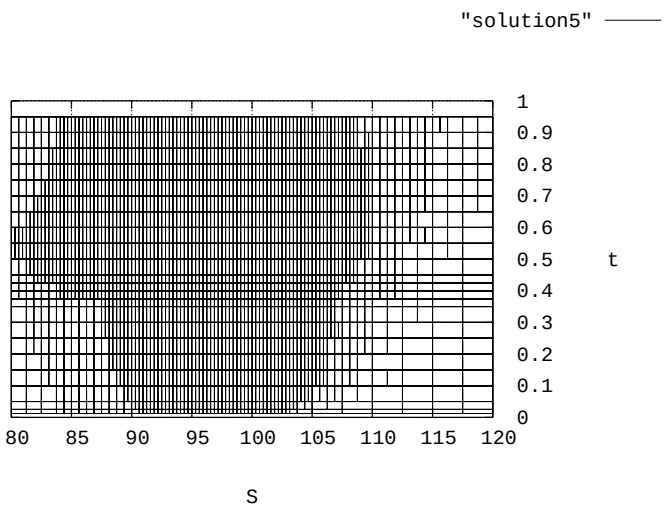


Figure 1: The local volatility surface

4 Calibration with American options

The calibration problem consists in finding η from the observations of

- the spot price $S_\circ$ today,
- the prices $(\bar{P}_i)_{i\in I}$ of a family of American puts with different maturities and different strikes $(T_i, K_i)_{i\in I}$.



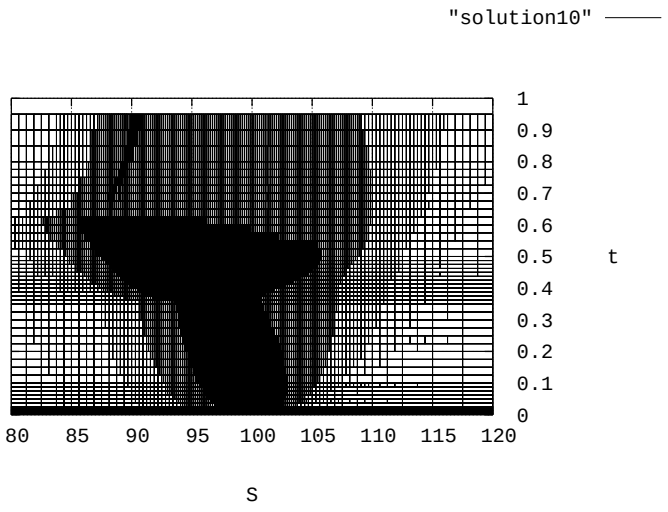


Figure 2: Two successive mesh refinements: the mesh is refined along the exercise boundary, see Figure 3

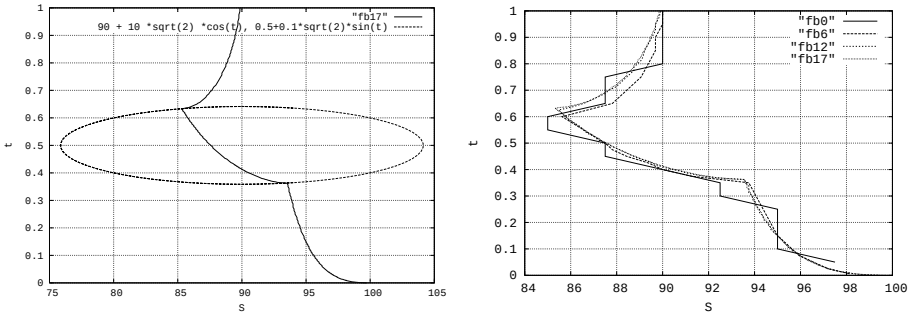


Figure 3: Left: the exercise boundary for the final mesh and the ellipse where the volatility jumps: there are two singularities corresponding to the jumps of volatility. Right: the exercise boundaries for different mesh refinements

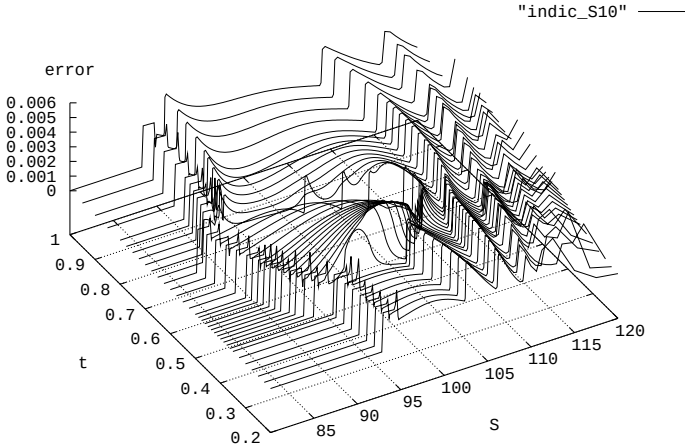


Figure 4: Indicators for the error due to the discretization with respect to S

We call $T = \max_{i\in I} T_i$. We assume that for any $i \in I$, the maturity T_i coincides with some node of the time grid, i.e. there exists $N_i \leq N$ such that $t_{N_i} = T_i$. We also assume that for any $i \in I$, the strike K_i is a node of the S-grid, i.e. there exists $\kappa_i < N_h$ such that $K_i = S_{\kappa_i}$.
We consider the least square problem:

$$\text{find } \eta \in \mathcal{H} \text{ minimizing} \quad J(\eta) + J_R(\eta), \qquad J(\eta) = \sum_{i\in I} |P_i^{N_i}(S_\circ) - \bar{P}_i|^2, \tag{37}$$

where $\mathcal{H}$ is a suitable closed subset of a possibly infinite dimensional function space, J_R is a suitable Tychonoff regularization functional, and

$$\text{find } (P_i^n)_{0\leq n\leq N_i}, P_i^n \in \mathcal{K}_{h,i} \text{ satisfying}$$
$$P_i^0 = P_{\circ,i}, \tag{38}$$
$$\text{and for all } n,\ 1 \leq n \leq N_i,$$
$$\forall v \in \mathcal{K}_{h,i}, \quad \left(P_i^n - P_i^{n-1}, v - P_i^n\right) + \Delta t_n a_{T_i - t_n}(P_i^n, v - P_i^n) \geq 0, \tag{39}$$

where $= P_{\circ,i} = (K_i - S)_+$. We call $\mu_{i,j}^n$ the real number

$$\mu_{i,j}^n = \left(P_i^n - P_i^{n-1}, w^j - P_i^n\right) + \Delta t_n a_{T_i - t_n}(P_i^n, w^j - P_i^n). \tag{40}$$

4.1 Optimality conditions

In [1], the inverse problem corresponding to the continuous counterpart of (39) is studied and optimality conditions are given for suitable choices of $\mathcal{H}$ and J_R. Here, we aim at finding optimality conditions for the fully discrete problem (37).
In order to find optimality conditions for the present least-square problem, we replace the state equations (38) (39) by the above mentioned penalized problem, whose penalty parameter, called ϵ, will tend to 0. Doing so, we obtain a new least square problem, for which necessary optimality conditions are easily found. Then, we pass to the limit as ϵ goes to zero: we obtain the following result:

Theorem 3 *Let η^* be a minimizer of (37) which can be found as a limit of a sequence η_ϵ^* of minimizers for the penalized problems, and let $(P_i^{*,n})_{i\in I}$ be the solutions to (38) (39) with $\eta = \eta^*$. There exist $y_i^{*,n} \in \tilde{V}_h$, and $\alpha_{i,j}^n \in \mathbb{R}$, $1 \leq n \leq N_i$, $\rho \leq j \leq N_h$, $i \in I$, such that $\forall v \in \tilde{V}_h$,*

$$\begin{aligned}
&\left(y_i^{*,N_i}, v\right) + \Delta t_{N_i}\left(a_0^*(v, y_i^{*,N_i}) + \sum_{j=\rho}^{N_h} \alpha_{i,j}^{N_i} v(S_j)\right) = 2(P_i^{*,N_i}(S_\circ) - \bar{P}_i)v(S_\circ),\\
&\left(y_i^{*,n} - y_i^{*,n+1}, v\right) + \Delta t_n\left(a_{T_i - t_n}^*(v, y_i^{*,n}) + \sum_{j=\rho}^{N_h} \alpha_{i,j}^n v(S_j)\right) = 0, \quad 1 \leq n < N,
\end{aligned} \tag{41}$$

with, for all j, n, $\rho \leq j \leq N_h$, $\quad 1 \leq n \leq N_i$,

$$\alpha_{i,j}^n(P_i^{*,n}(S_j) - P_\circ(S_j)) = 0, \quad \mu_{i,j}^{*,n} y_i^{*,n}(S_j) = 0, \quad \alpha_i^n y_i^{*,n}(S_j) \geq 0,$$

such that for any $\eta \in \mathcal{H}$, noting by $\delta\eta = \eta - \eta^*$,

$$\begin{aligned}
0 \leq &< DJ_R(\eta^*), \delta\eta >\\
&- \frac{1}{2}\sum_{i\in I}\sum_{n=1}^{N_i} \Delta t_n \sum_{j=\rho}^{N_h} S_j^2 \delta\eta(S_j, T_i - t_n) y_i^{*,n}(S_j) \left(\begin{array}{c} \dfrac{P_i^{*,n}(S_j) - P_i^{*,n}(S_{j-1})}{h_j} \\ + \dfrac{P_i^{*,n}(S_j) - P_i^{*,n}(S_{j+1})}{h_{j+1}} \end{array} \right).
\end{aligned}$$

4.2 Differentiability

Proposition 1 *Take $\Delta t \leq \frac{1}{2\lambda}$ with λ as in (16). Assume that for all $\eta \in \mathcal{H}$ verifying (14) (15), the the parameters h and $\frac{h^2}{\min_n \Delta t_n}$ are small enough so that the matrices $\mathbf{A}_l^n$ and $\mathbf{M}_l + \Delta t_n \mathbf{A}_l^n$ are tridiagonal irreducible M-matrices for all n, $1 \leq n \leq N$ and l, $\alpha \leq l < N_h$. Let $\eta \in \mathcal{H}$ be such that the strict complementarity conditions*

$$P_i^n(S_j) > P_{\circ,i}(S_j) \Leftrightarrow \mu_{i,j}^n = 0, \tag{42}$$

for all $i \in I$ and for all $j, \rho \leq j \leq N_h$, where P_i^n is the solution to (11) (12), and $\mu_{i,j}^n = \left(P_i^n - P_i^{n-1}, w^j\right) + \Delta t_n a_{T-t_n}(P_i^n, w^j)$. The functional J is differentiable at η, and for any admissible variation χ of η,

$$\begin{aligned} &< DJ(\eta), \chi >= \\ &-\frac{1}{2}\sum_{i\in I}\sum_{n=1}^{N}\Delta t_n \sum_{j=\rho}^{N_h} S_j^2 \chi(S_j, T-t_n) y_i^n(S_j) \left(\begin{array}{c} \dfrac{P_i^n(S_j) - P_i^n(S_{j-1})}{h_j} \\ +\dfrac{P_i^n(S_j) - P_i^n(S_{j+1})}{h_{j+1}} \end{array} \right). \end{aligned} \tag{43}$$

where $y_i^n = y^n(\eta)_i \in \tilde{V}_h$, $\alpha_{i,j}^n \in \mathbb{R}$, $\rho \leq j \leq N_h$, are the solution to: $\forall v \in \tilde{V}_h$,

$$\left(y_i^{N_i}, v\right) + \Delta t_{N_i}\left(a_0(v, y_i^{N_i}) + \sum_{j=\rho}^{N_h} \alpha_{i,j}^{N_i} v(S_j)\right) = 2(P_i^{N_i}(S_\circ) - \bar{P}_i)v(S_\circ),$$

$$\left(y_i^n - y_i^{n+1}, v\right) + \Delta t_n\left(a_{T_i - t_n}(v, y_i^n) + \sum_{j=\rho}^{N_h} \alpha_{i,j}^n v(S_j)\right) = 0, \quad 1 \leq n < N,$$

with

$$\alpha_{i,j}^n(P_i^n(S_j) - P_\circ(S_j)) = 0, \quad \mu_{i,j}^n y_i^n(S_j) = 0, \quad \alpha_i^n y_i^n(S_j) \geq 0.$$

4.3 Algorithm

We describe the simplest possible projected descent method in the space Y, where the descent direction is computed thanks to the considerations above. The degrees of freedom of a function $\chi \in Y$ are the values of χ at some nodes of a grid and we call them $(\Lambda_\ell^*(\chi))_{1\leq\ell\leq L}$, ($\Lambda_\ell^*$ is the linear form on Y which maps χ to its value at a given node). We endow Y with the basis $(\Lambda_\ell(\chi))_{1\leq\ell\leq L}$ defined by $\Lambda_\ell^*(\Lambda_k) = \delta_{\ell k}$, and we define the inner product $(\sum_{\ell=1}^L a_\ell\Lambda_\ell, \sum_{\ell=1}^L b_\ell\Lambda_\ell)_Y = \sum_{l=1}^L a_\ell b_\ell$.

Algorithm

- Choose $\eta \in \mathcal{H}$, $\epsilon > 0$ and $\rho > 0$, set $e = +\infty$.
- **While** $e > \epsilon$ **do**
  1. Compute $(P_i)_{i\in I}$ by (38) (39), by using for example one of the algorithms proposed in §3.3 and $J(\eta) + J_R(\eta)$, $J(\eta) = \sum_{i\in I} |P_i^{N_i}(S_\circ) - \bar{P}_i|^2$,
  2. For all $i \in I$, compute $(y_i^n)_{1\leq n\leq N_i}$, $y_i^n \in \tilde{V}_h$ satisfying (41).

3. compute $\zeta \in Y$ such that for all $\chi \in Y$,

$$(\zeta, \chi)_Y = -\frac{1}{2} \sum_{i \in I} \sum_{n=1}^{N_i} \Delta t_n \sum_{j=\rho}^{N_h} S_j^2 \chi(S_j, T_i - t_n) y_i^n(S_j) \left(\begin{array}{l} \frac{u_i^n(S_j) - u_i^n(S_{j-1})}{h_j} \\ + \frac{u_i^n(S_j) - u_i^n(S_{j+1})}{h_{j+1}} \end{array} \right). \tag{44}$$

4. set $\tilde{\eta} = \pi_{\mathcal{H}}(\eta - \rho(\text{grad}J_R(\eta) + \zeta))$, $e = \|\tilde{\eta} - \eta\|$, $\eta = \tilde{\eta}$, where $\pi_{\mathcal{H}}$ is the projection on $\mathcal{H}$.

- **end_do**

The complete justification of the algorithm above is still an open question because it is not proved that $-\text{grad}J_R(\eta) - \zeta$ is always a descent direction. However, from Proposition 1, we know that most often, ζ is exactly $\text{grad}J(\eta)$: in this case, the algorithm coincides with a projected gradient method.
In the numerical tests below, we have used variants of this algorithm (an interior point algorithm due to J. Herskovits[10]-it is a quasi-Newton algorithm which can handle general constraints), which have proved very robust. In particular, we never experienced breakups caused by the fact that the direction ζ is not a descent direction.

Parallelism The algorithm above can be parallelized in a very natural way on a distributed memory machine with N_p processors, because the computations of the pairs (P_i, y_i), $i \in I$ are independent from each other. We split I in $I = \cup_{k=1}^{N_p} I_k$ in order to balance the amount of work among the processors, the processor labelled k being responsible for the sums over $i \in I_k$ in $J(\eta)$ and (44). Note that the complexity of the computation of P_i, y_i depends on i, so load balancing is not straightforward. The data for η and ζ are replicated on the N_p processors. The processor labelled k computes its own contribution to $J(\eta)$ and to (44), i.e. the sums over $i \in I_k$, in an independent manner, then communications are needed for assembling the sums over $i \in I$ in $J(\eta)$ and in (44).
For programming, we have used $C++$ with the message passing library *mpi*.

4.4 Results with American Puts on the FTSE Index

In this paragraph, we consider American puts on the footsie index. The data correspond to June 6, 2001. We thank José Da Fonseca for providing us with the data.
The price of the underlying asset is $x_\circ = 5890$. The American puts correspond to four different maturities: $0.122, 0.199, 0.295, 0.55$ years. We set $T = 0.55$. The interest rate r varies with time, so r is replaced by $r(t)$ in (39), and this function is known. For these maturities, the prices of the observed options vs strike are plotted on Figure 5. The aim is to find the volatility surface from these prices. The volatility is discretized by functions that are the sum of

- a piecewise affine function in the S-variable which is constant in the regions $S < 1000$ and $S > 9000$ and affine in the region $1000 < S < 9000$
- a bicubic spline in the region $1000 < S < 9000$, $|t - T/2| < T/2 + 0.1$, whose value and derivatives vanish on the boundary of this rectangle. The control points of the spline are plotted on Figure 6, where the time variable is $T - t$. We see that the control points are not uniformly distributed: the mesh is refined for small times t and at the money.

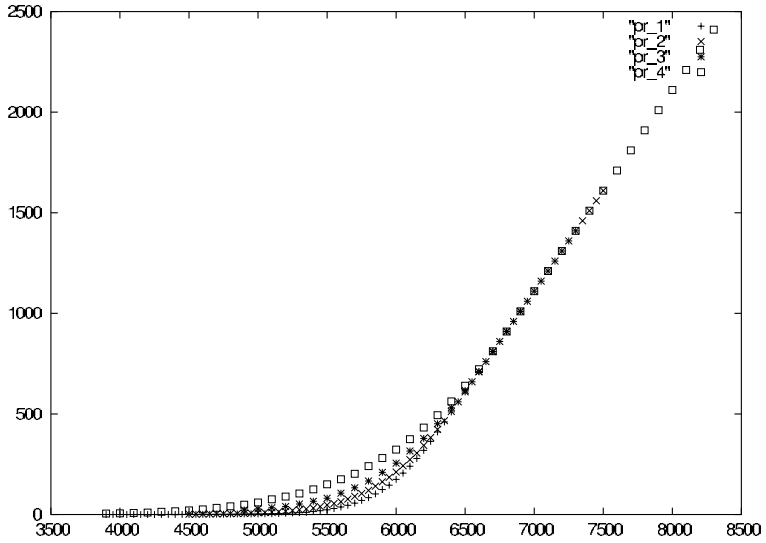


Figure 5: The data for the inverse problem: the prices of a family of American puts on the footsie index

The grid for u is non uniform with 745 nodes in the S-direction and 210 nodes in the t direction. For simplicity, the grid is chosen in such a way that the points $(T_i, K_i)_{i\in I}$ coincide with some grid nodes.
The (squared) volatility obtained at convergence is displayed on Figure 7: the surface has a smile shape. The relative errors between the observed prices and those computed at convergence are plotted on Figure 8, top. They are rather large for small values of K. However, we have to realize that the available observed prices are themselves given with a round-off error, which is exactly 0.5. On Figure 8, bottom, we have plotted the relative round-off error on the observed prices. Doing so, we see that the relative errors on the prices at convergence are of the same order as the round-off error on the observed prices. Therefore, it is very natural that the optimization program cannot improve on this level of error.

5 Automatic Differentiation by Operator Overloading

Derivatives of functions defined by their computer implementations can be calculated automatically and exactly. Several techniques are available; here the *forward mode* [9] is used. The basic idea is that each line of a computer program can be differentiated automatically, except perhaps branching statements but since there are only a finite number of them in a computer program differentiability will be obtained almost everywhere at worst.

Derivatives of a function can be computed from its differential form; this observation is easy to understand on the following example:

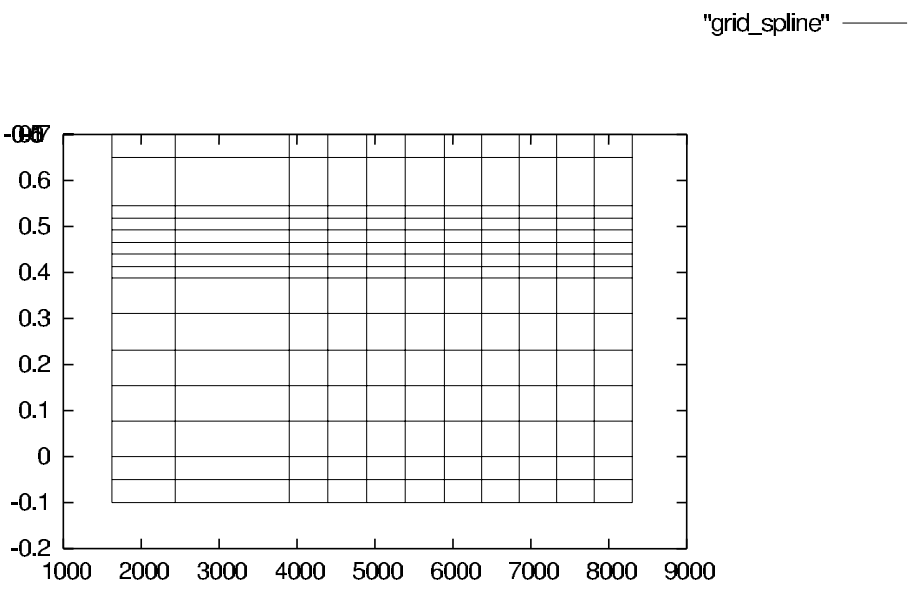


Figure 6: The control points of the bicubic splines

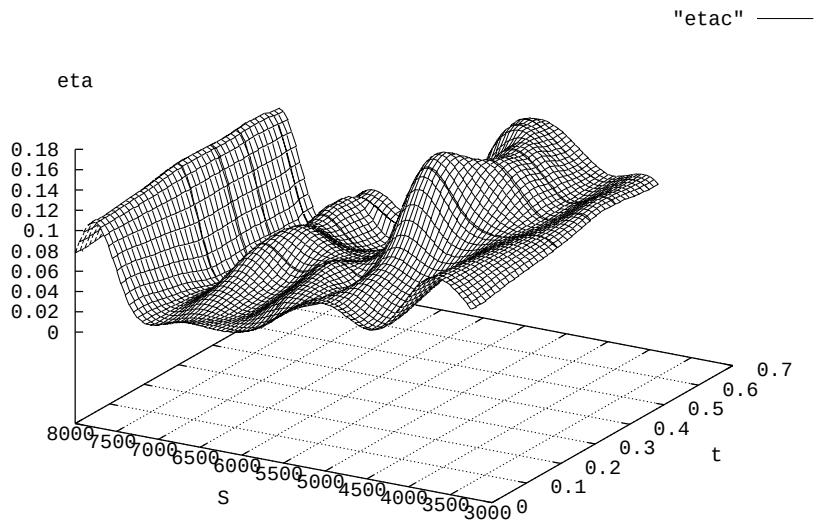


Figure 7: The squared volatility surface obtained by running the calibration program

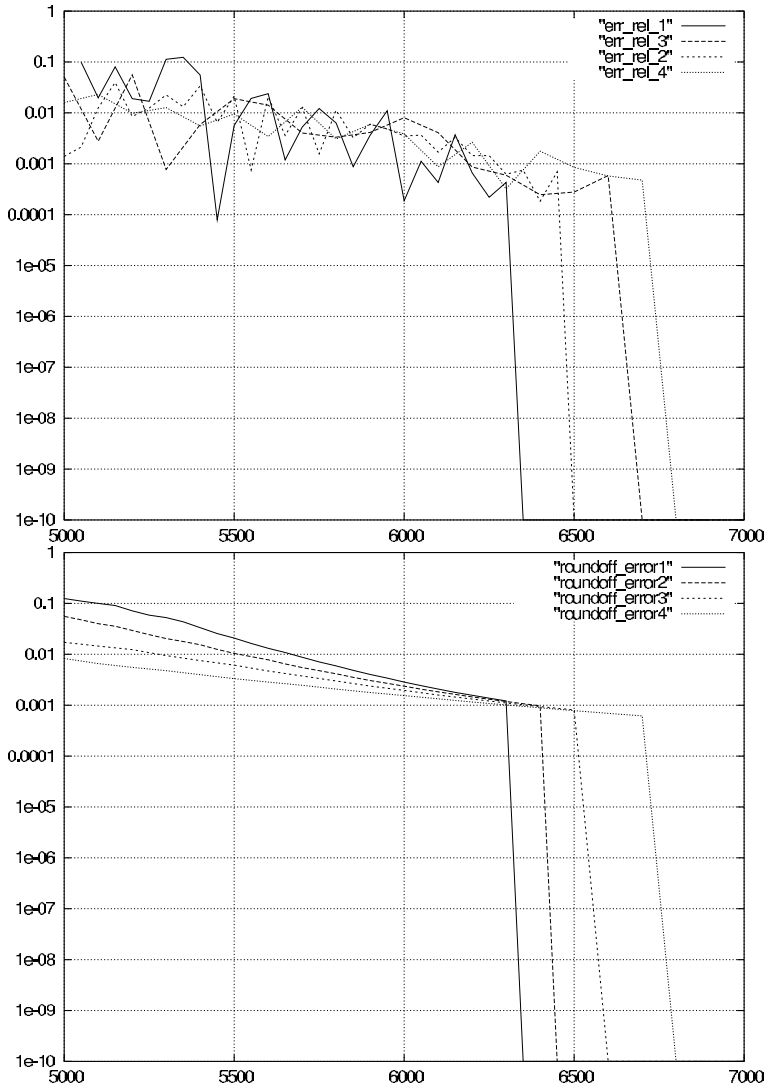


Figure 8: Top : relative errors between the observed prices and those obtained with η found after running the calibration program. A curve corresponds to a given maturity. Bottom: relative round-off error on observed prices. The two errors are of the same order.

Let $J(u) = |u - u_d|^2$, then its differential is

$$\delta J = 2(u - u_d)(\delta u - \delta u_d) \tag{45}$$

and obviously the derivative of J with respect to u is obtained by putting $\delta u = 1, \quad \delta u_d = 0$ in (45):

$$\frac{\partial J}{\partial u} = 2(u - u_d)(1.0 - 0.0)$$

Now suppose that J is programmed in C/C++ by

```
    double J(double u, double u_d){
        double z = u-u_d;
        z = z*(u-u_d);
        return z;
     }

    int main(){
        double u=2,u_d = 0.1;
        cout << J(u,u_d) << endl;
     }
```

A program which computes J and its differential can be obtained by writing above each differentiable line its differential form:

```
double JandDJ(double u, double u_d, double du,
                              double du_d, double *pdz)
{   double dz = du - du_d;
    double z = u-u_d;
    double dJ = dz*(u-u_d) + z*(du - du_d);
    z = z*(u-u_d);
    *pdz = dz;
    return z;
} int main()
     {      double u=2,u_d = 0.1;
            double dJ;
            cout << J(u,u_d,1,0,&dJ) << endl;
     }
```

Except for the embarrassing problem of returning both `z,dz` instead of z, the procedure is fairly automatic. It can be automatized more systematically by introducing a structured type of differentiable variable to hold the value of the variable and the value of its derivative:

```
    struct {double val[2];} ddouble;
```

and rewrite the above as

```
ddouble J(ddouble u, ddouble u_d) {
    ddouble z;
    z.val[1] = u.val[1]-u_d.val[1];
    z.val[0] = u.val[0]-u_d.val[0];
...
```

```
    return z;
} int main() {
    ddouble u;
    u.val[0]=2; u_d.val[0] = 0.1; u.val[1]=1; u_d.val[1] = 0.;
    cout  << J(u,u_d).val[1]<< endl;
}
```

In C++ the program can be simplified further by redefining the operators =, - and *. Then a class has to be used instead of a struct:

```
class ddouble{ public: double val[2];
  ddouble(double a, double b=0){ v[0] = a; v[1]=b;}
  ddouble operator=(const ddouble& a)
  {
    val[1] = a.val[1]; val[0]=a.val[0];
    return *this;
  }
friend dfloat operator - (const dfloat& a, const dfloat& b)
       {    dfloat c;
            c.v[1] = a.v[1] - b.v[1];   // (a-b)'=a'-b'
            c.v[0] = a.v[0] - b.v[0];
            return c;
       }
friend dfloat operator * (const dfloat& a, const dfloat& b)
       {    dfloat c;
            c.v[1] = a.v[1]*b.v[0] + a.v[0]* b.v[1];
            c.v[0] = a.v[0] * b.v[0];
            return c;
       }
};
```

As before a differentiable variable has two data fields, its value and the value of its derivative. Then we need a constructor to initialize such a variable and also the operator = to assign them to another one, so that `u=v` triggers `u.val[1]=v.val[1]` and `u.val[0]=v.val[0]`. The operator minus does the usual minus operation on the value of the variables and also on the value of their differentials. For the product the rule for the differentiation of products is used. Finally the function and its calling program are

```
ddouble J(ddouble u, ddouble u_d) {
    ddouble z= u-u_d
    z = z*(u-u_d); return z;
}
int main() {  ddouble u(2,1), u_d=0.1;
    cout  << J(u,u_d)<< endl;
}
```

The operator `<<` needs to be redefined also inside the class `ddouble`.

The conclusion is that a C program can be differentiated simply by replacing the keyword `double` *by* `ddouble`*.*

Of course C programs are not only assignments and it remains to check that branching statements, loops and function calls etc have the same property, for more details see [9].

5.1 Numerical Results

Market data will be made artificially available for each T and each K of Table 5.1

$100\frac{K}{S} =$	85	90	95	100	105	110	115	120	130	140
T=	0.175	0.425	0.695	0.94	1	1.5	2			

This is done by computing American options with these 70 values T_i, K_j and the volatility surface shown on figure 9, i.e. given by

```
double sigvalue(int n, int m)
 { int nn = n*60, mm = m*30;
   if(nn <= 1320)
      if((nn-600)*(nn-600)+(mm-750)*(mm-750) < 400*400)
        return 0.5;
   else
      if((nn-1800)*(nn-1800)+(mm-750)*(mm-750) < 400*400)
        return 0.3;
   return 0.4;
 }
```

Figure 9 shows also the spline interpolation of the test volatility surface with 10×7 parameters on the grid of known values.

In the computations there are 100 time steps and 150 spatial points for the finite difference discretization of S. The splines are defined with 5 points in S and 4 points in t, so the number of parameters for the volatility surface is 20 or 21; the additional parameter is the constant value of η outside the interval in which the spline is defined; indeed it is useless to try to vary η when S is small or large because the option is very insensitive to such variations, on the other hand the constant value of η in these small and large price regions is an important parameter. Only for American option with strategy 4, we did we not used this extra parameter and fixed it to the exact value 0.4.

There are 70 variational inequalities to solved each time the cost function or a component of its gradient needs to be evaluated. The convergence behavior is shown on the table below, each iteration requires between 3 and 5 trial and error in Armijo's rule.

Iteration number	Cost function	Gradient
0	52.917	58075.5
1	45.4533	110341
2	39.7139	112675
3	33.7126	33916.8
4	29.2176	6098.08
5	25.8939	18806.8
6	16.6763	7949.74
7	16.0644	10250.2
8	15.6205	17035

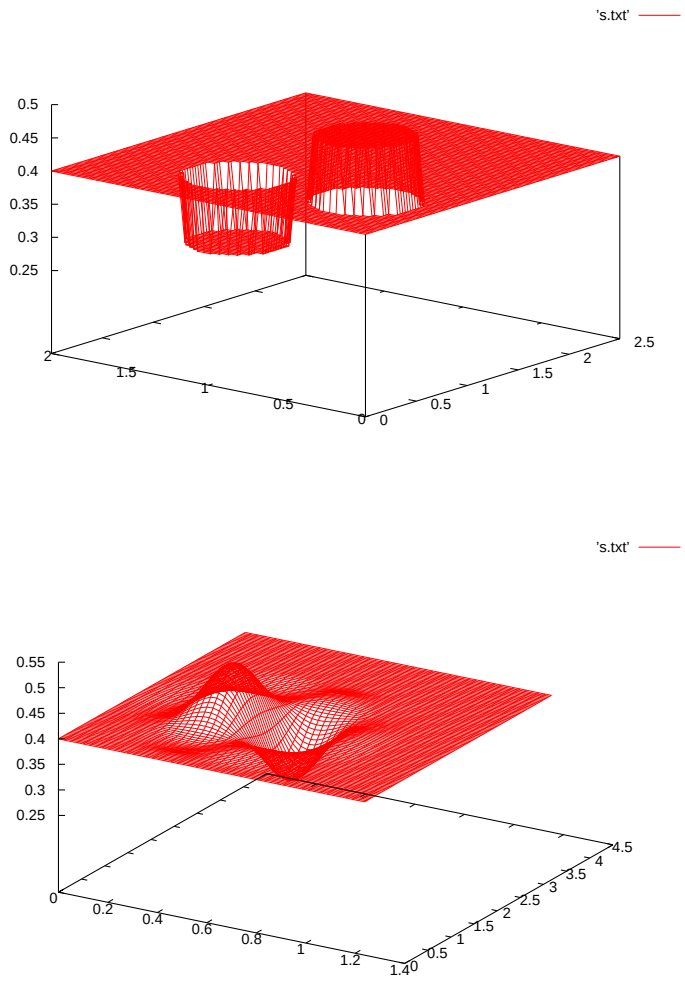


Figure 9: Test volatility surface for calibration of American options. On the right the same surface interpolated on a 10×7 spline is shown.

[**voldat**]

The elapsed time is 998.54 sec. on a pentium centrino 1 GHz.
The optimization algorithm is a conjugate gradient with Armijo's rule for the step size. The spline is applied to $S, t \to \psi(S,t)$ and then we set $\eta = \psi^2/(1+\psi^2)$; this insures $\eta \in (0,1)$ and is much more stable than a penalization. The results cannot be exactly the input smile because the splines are not the same. The two humps are found, more or less at the right place but one is too large; on the other line the error between the observations and the synthetic prices have been reduced by three orders of magnitude compared to a constant volatility and are very small.

References

[1] Yves Achdou. An inverse problem for a parabolic variational inequality arising in the calibration of American options. to appear in SIAM J. Control Optim.

[2] Yves Achdou and Olivier Pironneau. Volatility smile by multilevel least squares. *International Journal of Theoretical and Applied Finance*, 5(6):619–643, 2002.

[3] Yves Achdou and Olivier Pironneau. A numerical procedure for the calibration of American options. to appear in Applied Mathematical Finance, 2004.

[4] Yves Achdou and Olivier Pironneau. Computational methods for option pricing. to appear in SIAM Frontiers in Applied mathematics, 2005.

[5] M. Avellaneda, M. Friedman, C. Holmes, and D. Samperi. Calibrating volatility surfaces via relative entropy minimization. *Applied Mathematical Finances*, 4:37–64, 1997.

[6] R. Cont and P. Tankov. *Financial modelling with jump processes*. Chapman and Hall, 2003.

[7] B. Dupire. Pricing and hedging with smiles. In *Mathematics of derivative securities (Cambridge, 1995)*, pages 103–111. Cambridge Univ. Press, Cambridge, 1997.

[8] Jean-Pierre Fouque, George Papanicolaou, and K. Ronnie Sircar. *Derivatives in financial markets with stochastic volatility*. Cambridge University Press, Cambridge, 2000.

[9] Andreas Griewank. *Evaluating derivatives*, volume 19 of *Frontiers in Applied Mathematics*. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 2000. Principles and techniques of algorithmic differentiation.

[10] J. Herskovits. Feasible direction interior-point technique for nonlinear optimization. *J. Optim. Theory Appl.*, 99(1):121–146, 1998.

[11] S. Heston. A closed form solution for options with stochastic volatility with application to bond and currency options. *Review with Financial Studies*, pages 327–343, 1993.

[12] M. Hintermüller, K. Ito, and K. Kunisch. The primal-dual active set strategy as a semismooth Newton method. *SIAM J. Optim.*, 13(3):865–888 (electronic) (2003), 2002.

[13] Kazufumi Ito and Karl Kunisch. Semi-smooth Newton methods for variational inequalities of the first kind. *M2AN Math. Model. Numer. Anal.*, 37(1):41–62, 2003.

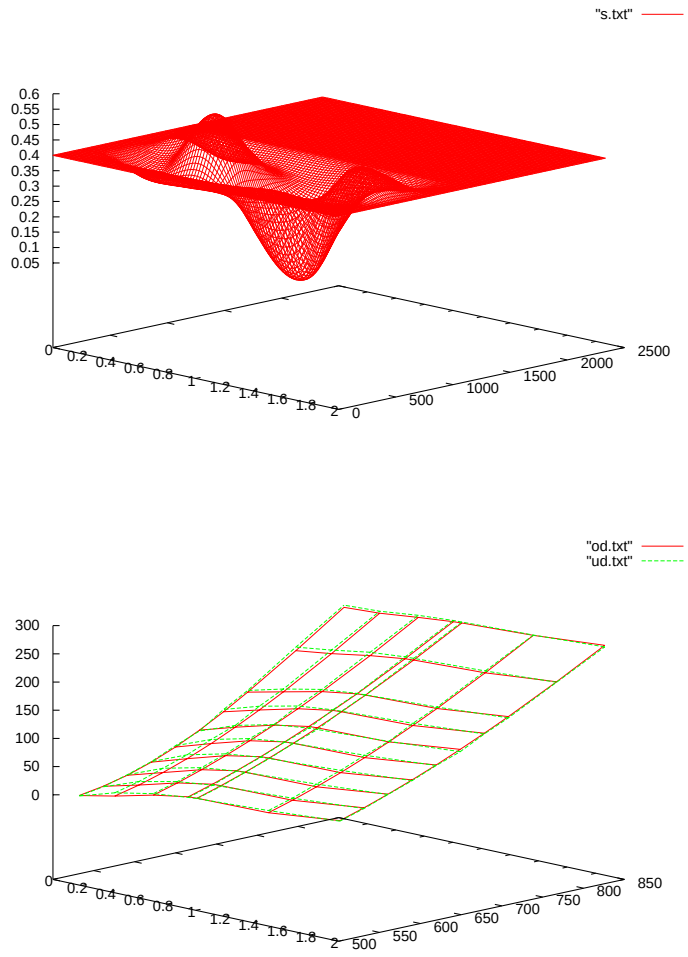


Figure 10: Test volatility surface obtained by calibration on American options using all 70 variational inequalities. On the right the observed and measured S, t data prices surfaces are shown. On the left the surface is to be compared with figure 9.

[AHH]

[14] Patrick Jaillet, Damien Lamberton, and Bernard Lapeyre. Variational inequalities and the pricing of American options. *Acta Appl. Math.*, 21(3):263–289, 1990.

[15] R. Lagnado and S. Osher. Reconciling differences. *RISK*, 10:79–83, 1997.

[16] R. Lagnado and S. Osher. A technique for calibrating derivative security pricing models: numerical solutions of an inverse problem. *The journal of computational finance*, 1(1):13–25, 1997.

[17] D. Lamberton and B. Lapeyre. *Introduction au calcul stochastique appliqué à la finance*. Ellipses, 1997.