Control and Cybernetics

vol. **39** (2010) No. 3

Reduced order models in PIDE constrained optimization*

by

Ekkehard W. Sachs and Matthias Schu

Universität Trier, 54286 Trier, Germany
sachs@uni-trier.de, schu@uni-trier.de

Abstract: Mathematical models for option pricing often result in partial differential equations originally starting with the Black-Scholes model. In this context, recent enhancements are models driven by Levy processes, which lead to a partial differential equation with an additional integral term. If one solves the problems mentioned last numerically, this yields large linear systems of equations with dense matrices. However, by using the special structure and an iterative solver the problem can be handled efficiently. To further reduce the computational cost in the calibration phase we implement a reduced order model, like proper orthogonal decomposition (POD), which proves to be very efficient. In this paper we use a special multi-level trust region POD algorithm to calibrate the option pricing model and give numerical results supporting the gain in efficiency.

Keywords: jump diffusion models, PIDE, proper orthogonal decomposition, trust region method, TRPOD.

1. Introduction

The pricing of European options is a well-known problem in financial mathematics. The most cited papers in this context were published by Black, Scholes (1973) and Merton (1973) over 30 years ago. Generally, the pricing is based on assumptions concerning behavior of the underlying price. In the above mentioned papers the authors assume a normally distributed return with constant volatility. This approach can be improved by models using a local or stochastic volatility or, more recently, including additional jumps driven by compound Poisson processes, so called jump-diffusion models (see Cont and Tankov, 2004, or Schoutens, 2003). While the first basic models result in partial differential equations, the latter, in addition, include an integral term.

In this paper we deal with two issues arising in the numerical aspects of this modeling approach. First we consider the partial integro-differential equations

*Submitted: July 2009; Accepted: June 2010.

(**PIDE**) and implement an efficient way of solving them. The second issue is the use of a reduced order model in the calibration problem, which results in an optimization problem with PIDE constraints.

During the numerical solution of the PIDE the integral term leads to some difficulties, since it causes the systems of equations, which result from a space discretization, to contain dense matrices. In principle, two approaches have emerged in the numerical solution of these dense systems. Either one uses a splitting approach, where the dense part is treated explicitly and the rest implicitly, or one employs a fully implicit scheme where the special structure of the dense matrix is exploited.

The first approach, the implicit-explicit scheme, has been followed by Cont and Voltchkova (2003) or Briani, Natalini and Russo (2004) who split the matrix into two parts by treating the dense part resulting from the integral term explicitly.

For stability purposes it is desirable to work with an implicit scheme instead. However, the cost of the dense system solves are prohibitive unless one uses the structure of the dense matrix. In Sachs and Strauss (2008) the case of symmetric jump size distribution functions was considered, which leads to the dense matrix being symmetric and Toeplitz. In this case a conjugate gradient algorithm with special preconditioners resulted in an overall efficiency of $\mathcal{O}(n \log n)$ for each time step, which is of similar complexity as in the case without the dense matrix term, where n denotes the discretization steps in space direction.

In this paper, we consider a general distribution function, which is not necessarily symmetric. For the discretized version, this yields a nonsymmetric Toeplitz matrix in the system to be solved at each time step, if one uses a fully implicit scheme like Crank-Nicolson. Also, if one uses the Dupire version of the problem, the symmetry of the matrix even in case of a symmetric distribution functions is destroyed. The resulting systems of equations are solved by an iterative scheme like a preconditioned GMRES algorithm. For the dense, but Toeplitz, matrix the matrix-vector-multiplication can be implemented efficiently via fast Fourier transformation. Also in this case we obtain an overall complexity of $\mathcal{O}(n \log n)$. We present numerical results which confirm this claim and also compare our approach with an implicit-explicit method.

Other research along the line of fully implicit methods was, for instance, done by Almendral and Oosterlee (2005) or Ikonen and Toivanen (2006), who both used a splitting technique to solve the dense linear systems of equations, or by Matache, von Petersdorff and Schwab (2004), who used a special wavelet basis for the spatial discretization.

During the calibration process of a PIDE model, many system solves are required, which could result in a fairly long computing time. Hence, the second goal of the paper is the acceleration of the calibration of a jump-diffusion model. We take here advantage of the multiple solution of similar problems by building a reduced order model for the PIDE, based on proper orthogonal decomposition (**POD**). Since the reduced order model is only a local model

based on a particular set of parameters, it has to be updated from time to time. A well-defined algorithm, called TRPOD, was developed by Arian, Fahl and Sachs (2000) by embedding the reduced model in a trust-region framework. The well-known quadratic model function of an ordinary trust-region algorithm is replaced by a non-quadratic function based on the POD model. To further reduce the computational effort a multi-level strategy is implemented. A hierarchy of discretization levels is set up and the levels are organized like a nested iteration in a multi-grid solver.

The paper is organized as follows. Section 2 gives a brief overview of option pricing models, especially jump-diffusion models, leading to partial integro-differential equations. The numerical solution of these PIDEs via finite element discretization in space direction and finite difference discretization in time direction is discussed in Section 3 including numerical results. Section 4 describes the proper orthogonal decomposition and how this can be used in order to create a reduced order model. Numerical results of the TRPOD and the multi-level TRPOD algorithms are presented and discussed in Section 5 followed by some final conclusions.

2. Option pricing models

First we give a brief overview about option pricing models. For the definition of an option and the financial background we refer the reader to, e.g. Hull (2006). The approach of Black, Scholes (1973) and Merton (1973) was to model the market price of the underlying with a Brownian motion, i.e. the return of the underlying is normally distributed and therefore the price itself is log-normally distributed. Upon applying some stochastic arguments the price of a call option $C(t,S)$, depending on the current time t and the current underlying price S, is given by the following partial differential equation, known as Black-Scholes equation

$$\begin{aligned} C_t(t,S) + \tfrac{1}{2}\sigma^2 S^2 C_{SS}(t,S) + rSC_S(t,S) - rC(t,S) = 0, \\ (t,S) \in (0,T) \times (0,\infty), \\ C(0,t) = 0, \quad t \in [0,T], \\ C(T,S) = \max\{S-B,0\}, \quad S \in (0,\infty), \end{aligned} \tag{1}$$

where T is the maturity, B the strike price of the option, and S_T the underlying price at maturity. Parameters, which have to be defined, e.g. through a calibration process, are the volatility σ and the risk-free interest rate r.

Although this model admits a closed-form solution, it contains also some weaknesses, like the fact that parameters r and σ are constant. It has been extended to local volatility models with varying parameters $r(t,T)$ or $\sigma(t,T,S,B)$, e.g. Dupire (1994). Another weakness is the fact that the underlying price sometimes makes large jumps, which cannot be modeled with a log-normal distribution. One approach to solve this problem are Lévy models, where the Brownian

motion of the Black-Scholes model is replaced by a more general Lévy process. Here a composite Poisson process is added to the Brownian motion, resulting in jump-diffusion models (see Cont and Tankov, 2004, or Schoutens, 2003). In this case the call price is given by the solution of the following parabolic integro-differential equation (PIDE):

$$\begin{gathered} C_t(t,S) + \tfrac{1}{2}\sigma^2(t,S)S^2C_{SS}(t,S) + r(t)SC_S(t,S) - r(t)C(t,S) \\ +\lambda \int\limits_{-\infty}^{+\infty} \Big(C(t,Se^y) - C(t,S) - S(e^y-1)C_S(t,S)\Big) f(y)dy = 0, \\ (t,S) \in (0,T)\times(0,\infty), \\ C(0,t) = 0, \quad t \in [0,T], \\ C(T,S) = \max\{S-B,0\}, \quad S \in (0,\infty) \end{gathered} \tag{2}$$

with the additional parameter $\lambda \geq 0$ for the frequency of the jumps and a density function $f(y)$ for the distribution of the jump sizes.

EXAMPLE 1 *Merton model:* $f(y) = \frac{1}{\sqrt{2\pi}\sigma_M} exp\{-\frac{(y-\mu_M)^2}{2\sigma_M^2}\}$ *(see Merton, 1976)*

EXAMPLE 2 *Kou model:* $f(y) = p\cdot\eta_1\cdot e^{-\eta_1 y}\cdot 1_{\{y\geq 0\}} + (1-p)\cdot\eta_2\cdot e^{\eta_2 y}\cdot 1_{\{y<0\}}$ *with* $\eta_1 > 1$ *and* $\eta_2 > 0$ *(see Kou, 2002)*

Note that the solution of the PIDE (2) depends on the strike price B and the maturity T and has to be solved for each option with different T or B. This is the case for a calibration problem when σ or f are to be determined from market prices of options with the same underlying but with various maturities T and strike prices B. In a least squares formulation, a function evaluation itself would require a tremendous amount in computing time to solve a whole family of problems of the type (2).

A similar problem occurs when we consider the extension of the volatility in the original Black-Scholes equation from a constant to a function, the so-called local volatility model. Dupire (1994) solved the problem described in the previous paragraph by formulating a different PDE where T and B occur as variables and the current time t and stock price S show up in the initial conditions of the PDE. Therefore, only one PDE has to be solved for a function evaluation in a least squares formulation.

For the PIDE case, there is a similar variant of Dupire's equation. Andersen and Andreasen (2000) also include a maturity- and strike price-dependent volatility in the PIDE. According to this approach and the additional variable transformation $x = \ln\left(\frac{B}{S}\right)$ we obtain the prices also by solving the following PIDE, where the current time t and the current stock price S appear in the initial condition (we consider here only $t = 0$), but the prices can be obtained

from one solution $D(T,x)$ for all maturities T and strike prices B (through x).

$$\begin{aligned} D_T(T,x) - \tfrac{1}{2}\sigma^2(T,x)D_{xx}(T,x) + \big(r(T) + \tfrac{1}{2}\sigma^2(T,x) - \lambda\zeta\big)\, D_x(T,x) \\ +\lambda(1+\zeta)D(T,x) - \lambda \int_{-\infty}^{+\infty} D(T,x-y)e^y f(y)dy = 0, \\ (T,x) \in (0,T_{max}) \times (-\infty,\infty), \\ D(0,x) = \max\{S_0 - S_0 e^x, 0\} =: D_0(x), \quad x \in (-\infty,\infty), \end{aligned} \tag{3}$$

where we use the abbreviation $\zeta = \int_{I\!R} e^y f(y)\ dy - 1$.

The next section addresses the numerical solution of the PIDE above.

3. Numerical solution of the PIDE

For the numerical solution of (3) we use a finite element approach for the spatial variable and an implicit finite difference scheme, like Crank-Nicolson, for the time discretization. Hence, we first restrict the infinite spatial interval $(-\infty,+\infty)$ to a finite one $[\underline{x},\overline{x}]$ and introduce boundary conditions

$$D(\underline{x},T) = S_0 - S_0 e^x e^{-rT} \quad \forall x \le \underline{x} \ \text{ and } \ D(\overline{x},T) = 0 \quad \forall x \ge \overline{x} \tag{4}$$

which can easily be derived by economic considerations. Note that even after truncation of the real line we need boundary conditions also on $(-\infty,\underline{x}]$ and $[\overline{x},+\infty)$, due to the translation in the integral term in the PIDE.

For the spatial discretization we obtain the following variational formulation:

$$\begin{aligned} \int_{\underline{x}}^{\overline{x}} D_T(x,T)\Phi(x)dx &= -\int_{\underline{x}}^{\overline{x}} \frac{1}{2}\sigma^2(x,T)D_x(x,T)\Phi'(x)dx \\ &- \int_{\underline{x}}^{\overline{x}} \left(r(T) + \frac{1}{2}\sigma^2(x,T) - \lambda\zeta + \sigma(x,T)\sigma_x(x,T)\right) D_x(x,T)\Phi(x)dx \\ &- \int_{\underline{x}}^{\overline{x}} \lambda(1+\zeta)D(x,T)\Phi(x)dx + \lambda \int_{\underline{x}}^{\overline{x}} \int_{I\!R} D(x-y,T)\Phi(x)e^y f(y)dy\, dx. \end{aligned}$$

The right-hand side of this equation defines a time-dependent bilinear form $a(T;v,w) : [0,T_{max}] \times (H^1 \times H^1) \to I\!R$. If $D^b(x,T) \in W([0,T_{max}],H^1)$, where $W([a,b],V) := \{u : u \in L_2((a,b),V), u' \in L_2((a,b),V')\}$ (see Dautray and Lions, 1992), is an arbitrary function that fulfills the boundary conditions, we obtain the following weak problem formulation

PROBLEM 1 *Find* $\tilde{D} \in W([0,T_{max}],H_0^1(\underline{x},\overline{x}))$ *satisfying*

$$\big\langle \tilde{D}_T(\cdot,T), w(\cdot)\big\rangle_{L_2} + a\big(T;\tilde{D}(\cdot,T),w(\cdot)\big) = F\big(T;w(\cdot)\big) \quad \forall\ w \in H_0^1(\underline{x},\overline{x})$$

with initial condition $\big\langle \tilde{D}(\cdot,0), w(\cdot)\big\rangle_{L_2} = \big\langle \tilde{D}_0(\cdot), w(\cdot)\big\rangle_{L_2} \quad \forall\ w \in H_0^1(\underline{x},\overline{x})$. *Here*

$$\begin{aligned} F\big(T;w(\cdot)\big) &= -\big\langle D_T^b(\cdot,T), w(\cdot)\big\rangle_{L_2} - a\big(T;D^b(\cdot,T),w(\cdot)\big), \\ \tilde{D}_0(x) &= D_0(x) - D^b(x,0). \end{aligned}$$

The next step is the discretization of $H_0^1(\underline{x}, \overline{x})$ by an appropriate finite-dimensional space $\mathcal{H}_n$ with orthonormal basis function $\{\Phi_i\}_{i=1}^n$, e.g. linear splines. Let M denote the mass matrix $M_{ji} = \langle \Phi_i, \Phi_j \rangle_{L_2}$, $A(T)$ the time-dependent stiffness matrix $A_{ji}(T) = a(T; \Phi_i, \Phi_j)$, $F(T)_j = F(T; \Phi_j)$ and $B_j = \langle \tilde{D}_0, \Phi_j \rangle_{L_2}$. This leads to the system of ordinary differential equations (ODE)

$$M \cdot \dot{\alpha}(T) + A(T) \cdot \alpha(T) = F(T) \tag{5}$$

with initial condition $M \cdot \alpha(0) = B$.

Due to the integral term of the bilinear form, the resulting matrix $A(T)$ is dense. Hence an implicit method for the ODE leads to systems of equations with dense matrices.

Sachs and Strauss (2008) did not consider the Dupire version of the PIDE but the original PIDE, which leads to a slightly different type of the system, where the stiffness matrix is symmetric and Toeplitz. This allows the use of a conjugate gradient method with special preconditioners for the solution of the implicit time step, whereas here this method cannot be used due to the lack of symmetry.

The splitting approach divides the matrix $A(T)$ into two parts, A_I containing the terms resulting from the double integral and the rest called $A_{NI}(T)$. The entries along the diagonals of A_I are constant for equidistant grids and while the matrix is no longer symmetric it is still Toeplitz. This allows the use of fast Fourier transformation to efficiently calculate matrix-vector products with complexity of $\mathcal{O}(n \log n)$. The remaining part $A_{NI}(T)$ is non-Toeplitz due to the space dependent volatility function, but sparse since we use basis function with local support:

$$\underbrace{A(T)}_{dense\ matrix} = \underbrace{A_{NI}(T)}_{tridiagonal\ matrix} + \underbrace{A_I}_{Toeplitz\ matrix} . \tag{6}$$

Cont and Voltchkova (2003) or Briani, Natalini and Russo (2004) propose a splitting technique to solve the system of ODEs. Since a fully explicit scheme would be restricted by a strong CFL condition, they use for the dense part a higher order Runge-Kutta method method to avoid too small time steps ΔT versus Δx. For example, if m denotes the number of discretization steps in time direction and the sparse part of the matrix is treated by a Crank-Nicolson scheme and only the dense integral part explicitly by an Euler method, we obtain

$$\begin{aligned} &\Big(M + \frac{\Delta T}{2} A_{NI}(T_{k+1})\Big)\alpha(T_{k+1}) = && k = 1, \ldots, m \\ &= \Big(M - \frac{\Delta T}{2} A_{NI}(T_k) - \Delta T A_I\Big)\alpha(T_k) + \frac{\Delta T}{2}\Big(F(T_{k+1}) + F(T_k)\Big). \end{aligned} \tag{7}$$

In contrast, a full implicit method to solve the problem by a Crank-Nicolson

scheme results in

$$\Big(M + \frac{\Delta T}{2} A(T_{k+1})\Big)\alpha(T_{k+1}) = \qquad k = 1, \dots, m$$
$$= \Big(M - \frac{\Delta T}{2} A(T_k)\Big)\alpha(T_k) + \frac{\Delta T}{2}\Big(F(T_{k+1}) + F(T_k)\Big). \tag{8}$$

The system matrix from the equation above is split as follows and the resulting flops for the matrix-vector-multiplication are indicated below:

$$\Big(M + \frac{\Delta T}{2} A(T_{k+1})\Big)x = \underbrace{M \cdot x}_{\mathcal{O}(n)} + \frac{\Delta T}{2}\Big(\underbrace{A_{NI}(T_{k+1}) \cdot x}_{\mathcal{O}(n)} + \underbrace{A_I \cdot x}_{\mathcal{O}(n \log n)}\Big).$$

Although the whole stiffness matrix is dense, the matrix vector product can be realized with only $\mathcal{O}(n \log n)$ operations due to the Toeplitz structure of the dense part.

As an iterative solver we use a preconditioned GMRES algorithm with the inverse of the tridiagonal matrix $A_{NI}(T)$ as a preconditioner. In our numerical tests the GMRES algorithm terminated on the average after three iterations per time step using an accuracy of 10^{-8} on the residual.

Table 1 shows some numerical results. We consider the jump-diffusion model by Merton (see EXAMPLE 1). We compared the error to the closed-form solution given by an infinite series. Furthermore, a comparison to the implicit-explicit splitting technique according to (7) is given. The constants used were $\underline{x} = -6$, $\overline{x} = 6$, $T_{max} = 5y$, $S_0 = 1$, $r = 3\%$, $\sigma \equiv 30\%$, $\lambda = 50\%$, $\mu_J = 0\%$, $\sigma_J = 50\%$ and the l^∞-error was measured on the interval $[\underline{B}, \overline{B}] := [0.2, 10]$.

Table 1. l^∞-error between discretized problem and closed-form solution (Merton model) at different time slices

Discretization		Effort			l^∞-error at time slice		
x	T	time(sec.)	mult^{toepl}	iter	$T = 1$	$T = 3$	$T = 5$
Implicit-explicit scheme							
1000	800	1.11	800	—	1.3e-004	2.9e-004	4.0e-004
1000	1600	2.13	1600	—	6.7e-005	1.5e-004	2.0e-004
1000	3200	4.70	3200	—	3.5e-005	7.6e-005	1.0e-004
2000	800	3.20	800	—	1.3e-004	2.9e-004	4.0e-004
2000	1600	7.72	1600	—	6.6e-005	1.5e-004	2.0e-004
2000	3200	13.80	3200	—	3.3e-005	7.3e-005	1.0e-004
Crank-Nicolson scheme							
1000	100	1.25	649	349	3.2e-004	5.6e-005	1.7e-005
1000	200	3.95	1204	604	3.6e-005	5.6e-006	7.5e-006
1000	400	7.14	2400	1200	9.4e-006	5.8e-006	7.5e-006
2000	100	3.56	666	366	5.4e-004	2.0e-004	1.2e-004
2000	200	6.42	1208	608	1.5e-004	2.6e-005	7.3e-006
2000	400	12.78	2398	1198	1.5e-005	1.4e-006	1.8e-006

Note that an increase in the time or spatial discretization shows the linear dependence on the computing time (third column). The implicit-explicit scheme

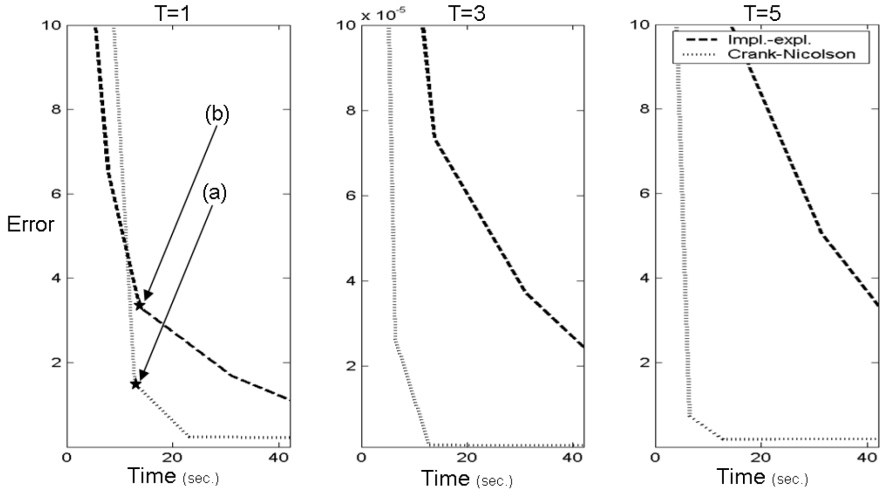


Figure 1. x-axis: time required for the solution at different T-discretizations (x-steps fixed to 2000), y-axes: l^∞-error at different time slices for Crank-Nicolson and implicit-explicit method

needs one matrix-vector-multiplication per time step (fourth column), whereas the GMRES algorithm in the Crank-Nicolson method requires approximately six per time step (fourth column).

In order to make the results comparable, the T-discretization of the implicit-explicit scheme was refined so that the computing times for the implicit-explicit scheme and for Crank-Nicolson were similar. The last three columns show the l^∞-errors for three different time slices, $T = 1y$, $T = 3y$ and $T = 5y$, where T_{max} was also set to $5y$. Even for a highly refined time stepping in the implicit-explicit scheme, the results for the Crank-Nicolson scheme provided higher accuracy.

Fig. 1 is a graphical illustration of this effect. The space discretization is fixed to 2000 steps. For instance, the asterisk in the first picture, tagged with an (a), represents the error of the Crank-Nicolson method at time slice $T = 1$ for a discretization of 2000 x-steps and 400 T-steps. The solution requires 12.78 sec (see Table 1), the value on the x-axis, with an the error of $1.5 \cdot 10^{-5}$, the value on the y-axis. Varying the time discretization results in different computing times and different errors and leads to light dotted curve. The dark dotted curve in the first picture represents the same for the implicit-explicit scheme. The point tagged by (b) represents an error of $3.3 \cdot 10^{-5}$ at a computing time of 13.80 sec with a discretization of 2000×3200. The three graphs show that if the discretization is chosen such that both approaches require the same computing time, then the Crank-Nicolson method usually leads to more accurate results.

4. Proper orthogonal decomposition

We introduce briefly the proper orthogonal decomposition (POD), a technique to obtain a problem of much smaller order than the original discretized version.

Let u_i, $i = 1, \ldots, n$ be elements of a real separable Hilbert space H which, for example, approximate the solution $u(t_i)$ of a differential equation at various time instances t_i. Those elements u_i are sometimes called "snapshots". The space spanned by the snapshots has dimension $r \geq 1$, i.e. $dim(span(u_1, \ldots, u_n)) = r$. Proper orthogonal decomposition consists of first finding elements $\Psi_j \in H$, $j = 1, \ldots, r$, that build an orthonormal basis of $span(u_1, \ldots, u_n)$ and have the following additional property: Considering the partial basis $\Psi_1, \ldots, \Psi_p$ for an arbitrary $p \in \{1, \ldots, r\}$, there are no other orthonormal basis functions $\Phi_1, \ldots, \Phi_p$, which approximate the average element of $span(u_1, \ldots, u_n)$ in a better way. The projection of a $v \in span(u_1, \ldots, u_n)$ on the space spanned by arbitrary orthonormal functions $\{\Phi_j\}_{j=1}^p$ can be computed from its Fourier expansion:

$$\tilde{v} = \sum_{j=1}^{p} \langle v, \Phi_j \rangle \Phi_j.$$

The mathematical definition for the POD basis functions is formulated as follows:

DEFINITION 1 *Given vectors $u_1, ..., u_n \in H$, find orthonormal vectors $\Psi_1, \ldots, \Psi_r \in span(u_1, \ldots, u_n)$ by solving the minimization problem:*

$$\min_{\Psi_1, \ldots, \Psi_p} \sum_{i=1}^{n} \gamma_i \left|\left| u_i - \sum_{j=1}^{p} \langle u_i, \Psi_j \rangle_H \Psi_j \right|\right|_H^2$$
$$s.t. \quad \langle \Psi_k, \Psi_l \rangle_H = \delta_{kl} \quad \forall k, l = 1, \ldots, p$$

for all $p \in \{1, \ldots, r\}$ with weights $\gamma_i > 0$, $i = 1, \ldots, n$. The first p vectors $\psi_1, ..., \psi_p$ are called a POD basis of rank p.

We shortly review how to calculate these POD basis functions. For this purpose we introduce the matrix $\mathcal{K} \in I\!R^{n \times n}$ with

$$\mathcal{K}_{ij} := \gamma_i \langle u_j, u_i \rangle_H \quad \forall\, i, j = 1, \ldots, n.$$

Solving the eigenvalue problem

$$\mathcal{K} v^k = \lambda_k v^k \quad k = 1, \ldots, r$$

(see Volkwein, 2001) where $v^k \in I\!R^n$, the POD basis functions Ψ_k for a basis of rank p ($\leq r$) are given by

$$\Psi_k = \frac{1}{\sqrt{\lambda_k}} \sum_{i=1}^{n} \gamma_i v_i^k u_i$$

where u_i, $i = 1, \ldots, n$ are the snapshots from above.

Considering only the first $p < r$ POD basis functions for a representation of the u_i, we have to deal with an approximation, the projection of u_i on the space spanned by $\{\Psi_k\}_{k=1}^{p}$. The error resulting from dropping the information contained in $\Psi_{p+1}, \ldots, \Psi_r$ is, according to Volkwein (2001), given by:

$$\sum_{i=1}^{n} \gamma_i \left\| u_i - \sum_{j=1}^{p} \langle u_i, \Psi_j \rangle_H \Psi_j \right\|_H^2 = \sum_{k=p+1}^{r} \lambda_k. \tag{9}$$

If we apply this technique to our problem, we specify the snapshots as the approximation to the solution of the problem at fixed time steps $t_1, \ldots, t_n$, i.e. $D(\cdot, t_1), \ldots, D(\cdot, t_n)$ by the finite element approach. From POD we obtain some orthonormal basis functions containing specific information about the solution. Solving the PIDE problem via a POD approach means that we replace the finite element basis functions by the POD basis function calculated from a given solution of the problem. Since we now only need a few functions - numerical tests show that 10 is already a sufficient quantity - compared to, e.g., 1000 finite element basis functions, the systems of equations that have to be solved are much smaller. Although the mass matrix is dense for the resulting system, this does not matter due to the small dimension of the system. The solution of such a reduced system, based on the POD basis functions, will be denoted by D^{POD} in the following section.

5. Calibration results

In this section we show how to use the reduced order model in the calibration process. Let market prices D_i^m $(i = 1, \ldots, k)$ of call options for different maturities T_i and different strike prices B_i $(= S_0 e^{x_i})$ be given. Denote by u the vector of the calibration parameters, which will be determined later, we can formulate the calibration problem as a least squares problem

$$\min f(u) := \sum_{i=1}^{k} ||\tilde{D}(u;\ T_i, x_i) - D_i^m||^2. \tag{10}$$

One function evaluation requires one solution of the PIDE in Problem 1. Of course, we cannot totally discard the PIDE for the POD model since the latter is built on a solution of the PIDE. But in a simplistic approach one would calculate the solution of the PIDE once, then build a POD model based on this solution and in the calibration phase only use this POD model. The error between the POD approximation and the true PIDE solution has been estimated in Sachs and Schu (2009), extending the results of Kunisch and Volkwein (2001) to the time dependent case.

Note, however, that the POD model is only a local model which approximates the PIDE solution. Therefore, if we veer away from the starting parameters during the calibration process, the error estimates that hold true for

unchanged parameters cannot be applied any longer. This was also confirmed with numerical results by Sachs and Schu (2007). For this reason Arian, Fahl and Sachs (2000) proposed a trust region POD algorithm for an optimal flow control problem in order to adaptively adjust the POD model. The idea here is to use a POD model function

$$m_k^{POD}(s) = \sum_{i=1}^{k} ||D_k^{POD}(s;\ T_i, x_i) - D_i^m||^2$$

instead of the well-known quadratic model function $m^{quad}(u_k + s) = f(u_k) + \nabla f(u_k)^T s + \frac{1}{2} s^T H_k s$ in the trust region framework.

ALGORITHM 1 *(Basic TRPOD Algorithm)*

1. *Solve the PIDE for the parameter set* u_k.
2. *Derive the corresponding POD basis and the POD control model.*
3. *Compute an approximate minimizer,* s_k, *of* $m_k^{POD}(u_k + s)$, *where* $||s|| \leq \delta_k$.
4. *Solve the PIDE for* $u_k + s_k$ *and compute*

$$\rho_k = \frac{f(u_k) - f(u_k + s_k)}{m_k^{POD}(u_k) - m_k^{POD}(u_k + s_k)}.$$

5. *Update the trust region radius* δ_{k+1}.
   Keep or drop the new solution $u_k + s_k$ *according to the value of* ρ_k.
   $k \leftarrow k + 1$. *GOTO 2.*

Regarding the computational effort, the most expensive part in the algorithm is the solution of the full PIDE in *4* to get $f(u_k + s_k)$.

To further reduce computational cost, Kragel (2005) introduced a multi-level approach. The discretization of the PIDE is not kept fixed, but one can switch between different discretization levels. Goal is to avoid solutions on the finest grids. Gratton, Sartenaer and Toint (2006) used a similar approach for the case of a quadratic model function and via an adaptive control of the discretization levels they even could state convergence results.

Denoting by h_k the level of discretization in step k, a general algorithm for POD based non-quadratic model functions could be

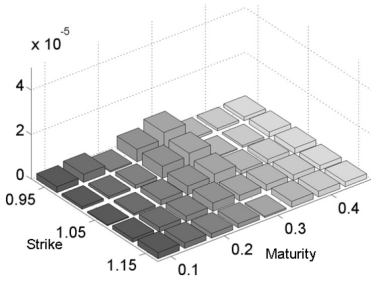
ALGORITHM 2 *(Multi-level TRPOD Algorithm)*

1. *Solve the PIDE for the parameter set* u_k *on level* h_k.
2. *Derive the corresponding POD basis and the POD control model.*
3. *Compute an approximate minimizer,* s_k, *of* $m_k^{POD}(u_k + s)$, *where* $||s|| \leq \delta_k$.
4. *Solve the PIDE on level* h_k *for* $u_k + s_k$ *and compute*

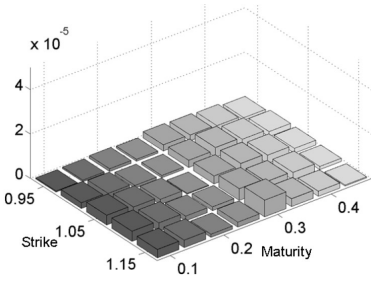
$$\rho_k = \frac{f(u_k) - f(u_k + s_k)}{m_k^{POD}(u_k) - m_k^{POD}(u_k + s_k)}.$$

5. *Update the trust region radius* δ_{k+1}.
   Keep or drop the new solution $u_k + s_k$ *according to the value of* ρ_k.
   Refine, coarse or keep the discretization level h_{k+1}.
   $k \leftarrow k+1$. *GOTO 2.*

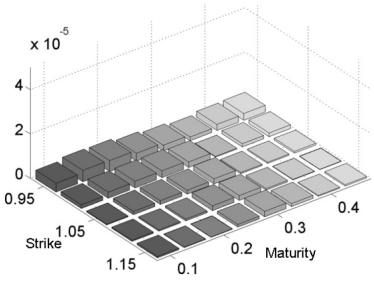
We give some numerical results for the calibration of a Lévy model with normally distributed jump sizes according to Merton (see EXAMPLE 1). We aim at calibrating the four parameters σ, λ, μ_J and σ_J, so that the model call prices fit to 40 artificially made market prices for calls at different maturities and strike prices. Note that the stiffness matrix depends on these parameters, so it has to be recalculated whenever a parameter is changed. This holds true for a normal finite element solution of the PIDE as well as for the POD solution.



(a) Line search algorithm



(b) TRPOD algorithm



(c) Multi-level TRPOD algorithm

Figure 2. Calibration error for the 40 options that are calibrated

In Fig. 2 and Table 2 we compare the two introduced algorithms implemented in Matlab with the Matlab routine *fmincon* which is based on a line search algorithm. For the finite element solution of our PIDE model we choose a discretization of 2000 space steps and 100 time steps on the finest grid. The multi-level algorithm also uses coarser grids 500×50 and 1000×80. The POD model only needs 10 POD basis functions, which leads to a linear system of equations of size 10×10 in each time step. Further settings used here are: $\underline{x} = -4$, $\overline{x} = 4$, $T_{max} = 0.5y$, $S_0 = 1$, $r = 3\%$, $\sigma^{opt} \equiv 30\%$, $\lambda^{opt} = 50\%$,

$\mu_J^{opt} = 0\%$, $\sigma_J^{opt} = 50\%$, $\sigma^{start} \equiv 40\%$, $\lambda^{start} = 70\%$, $\mu_J^{start} = 20\%$, $\sigma_J^{start} = 70\%$. Both trust-region algorithms used the Matlab function *fmincon* to solve the trust-region subproblems. Stopping criterion was a sufficient small function value. The error results for the 40 different options that are fitted can be found in Fig. 2 and are comparable to each other.

Table 2. Calibration results of three different algorithms. Comparison of runtimes and evaluations of the PIDE on different discretization levels, respective evaluations of the POD model

Algorithm	Time (sec.)			Evaluations			
	Total	FEM	POD	L^{2000}	L^{1000}	L^{500}	POD
Line search	963	963	—	324	—	—	—
TRPOD	59	35	24	9	—	—	328
Multi-level TRPOD	31	17	14	4	2	2	244

The interesting point here is the total computing time. As we compare the line search algorithm and the TRPOD we see a time reduction by factor of 16 and for the multi-level algorithm even a reduction factor of 32. The last four columns of the table show the reason for this improvement. In every algorithm the total number of function evaluations is in the same order of magnitude between 252 and 327, but the deciding number are the number of evaluations on the finest grid L^{2000}, where we have to solve a 100 systems of equations of size 2000×2000. Here the TRPOD approach only needs a fraction of the amount needed by the line search algorithm. And using coarser levels in the multi-level approach even leads to an additional improvement.

Gradient information was calculated via finite differences, what leads to a high number of function evaluations in all algorithms. Calibrating models with time- and space-dependent volatility function $\sigma(x, T)$ will require the use of adjoint equations to provide gradient information, which will be a topic of future research.

6. Conclusions

We showed an alternative way for solving a general class of PIDEs resulting from jump-diffusion option pricing models with time- and space-dependent parameters. After space and time discretization, the dense linear systems of equations were solved by an implicit time stepping scheme using a preconditioned GMRES algorithm for each time step. Here we made use of the fact that the dense part of the matrix of the linear system of equations is a Toeplitz matrix and via fast Fourier transformation the computing costs reduces to $\mathcal{O}(m \cdot n \cdot \log n)$.

Since the PIDE is of parabolic type, POD is well suited to create a reduced order model. Through the embedding in a trust-region framework a well-defined algorithm is constructed leading to a significant speed-up in the calibration

process. The multi-level approach further reduces the costs by avoiding the repeated solution of the PIDE on the finest grid.

References

Almendral, A. and Oosterlee, C.W. (2005) Numerical valuation of options with jumps in the underlying. *Appl. Numer. Math.* **53**, 1–18.

Andersen, L. and Andreasen, J. (2000) Jump-Diffusion Processes: Volatility Smile Fitting and Numerical Methods for Option Pricing. *Review of Derivatives Research* **4**, 231–262.

Arian, E., Fahl, M. and Sachs, E.W. (2000) Trust-Region Proper Orthogonal Decomposition for Optimal Flow Control. *NASA/CR-2000-210124, ICASE Report.*

Black, F. and Scholes, M. (1973) The Pricing of Options and Corporate Liabilities. *Journal of Political Economy* **81** (3), 637–654.

Briani, M., Natalini, R. and Russo, G. (2005) Implicit-Explicit Numerical Schemes for Jump-Diffusion Processes. *Calcolo.*

Cont, R. and Tankov, P. (2004) *Financial Modelling with Jump Processes.* Chapman and Hall.

Cont, R. and Voltchkova, E. (2005) A finite difference scheme for option pricing in jump diffusion and exponential Lévy models. *SIAM J. Numer. Anal.* **43** (4), 1596–1626

Dautray, R. and Lions, J.-L. (1992) *Mathematical Analysis and Numerical Methods for Science and Technology, Vol. 5: Evolution Problems I.* Springer, Berlin.

Dupire, B. (1994) Pricing with a smile. *Risk* **7** (1), 18–20.

Gratton, S., Sartenaer, A. and Toint, P.L. (2008) Recursive Trust-Region Methods for Multiscale Nonlinear Optimization. *SIAM J. on Optimization* **19** (1), 414–444.

Hull, J.C. (2006) *Options, Futures, and Other Derivatives.* Sixth ed., Prentice Hall, Upper Saddle River, N.J.

Ikonen, S. and Toivanen, J. (2006) Numerical valuation of European and American options with Kou's jump-diffusion model. *Amamef Conference on Numerical Methods in Finance*, INRIA-Rocquencourt, France.

Kou, S.G. (2002) A Jump-Diffusion Model for Option Pricing. *Mgmt. Sci.* **48** (8), 1086–1101.

Kragel, B. (2005) Streamline Diffusion POD Models in Optimization. *PhD* thesis, Universität Trier.

Kunisch, K. and Volkwein, S. (2001) Galerkin proper orthogonal decomposition methods for parabolic problems. *Numerische Mathematik* **90**, 117–148.

Matache, A.-M., von Petersdorff, T. and Schwab, C. (2004) Fast deterministic pricing of options on Lévy driven assets. *Math. Modelling Numer. Anal.*, **38** (1), 37–71.

Merton, R.C. (1973) Theory of rational option pricing. *Bell Journal of Economics and Management Science* **4**, 141–183.

Merton, R.C. (1976) Option pricing when underlying stock returns are discontinuous. *Journal of Financial Economics* **3** (1-2), 125–144.

Sachs, E.W. and Schu, M. (2009) A priori error estimates for reduced order models in finance. In preparation.

Sachs, E.W. and Schu, M. (2007) Reduced order models (POD) for calibration problems in finance. *ENUMATH Proceedings 2007*, Springer, Heidelberg, 735–742.

Sachs, E.W. and Strauss, A. (2008) Efficient solution of a partial integro-differential equation in finance. *Appl. Numer. Math.* **58** (11), 1687–1703.

Schoutens, W. (2003) *Levy-Processes in Finance: Pricing Financial Derivatives.* Wiley.

Volkwein, S. (2001) Optimal control of a phase-field model using proper orthogonal decomposition. *Zeitschrift für angewandte Mathematik und Mechanik* **81**, 83–97.