Calibration of GARCH Models using Concurrent Accelerated Random Search

Juliane Müller

Cornell University, School of Civil and Environmental Engineering
209 Hollister Hall, Ithaca, NY 14853, USA, juliane.mueller2901@gmail.com

Juho Kanniainen

Department of Industrial Management, Tampere University of Technology
P.O. Box 541, 33101 Tampere, Finland

Robert Piché

Tampere University of Technology, Department of Automation Science and Engineering,
Room SD206, 33720 Tampere, Finland

Abstract

This paper investigates a global optimization algorithm for the calibration of stochastic volatility models. Two GARCH models are considered, namely the Leverage and the Heston-Nandi model. Empirical information on option prices is used to minimize a loss function that reflects the option pricing error. It is shown that commonly used gradient based optimization procedures may not lead to a good solution and often converge to a local optimum. A concurrent approach where several optimizers ("particles") execute an accelerated random search (ARS) procedure has been introduced to thoroughly explore the whole parameter domain. The number of particles influences the solution quality and computation time, leading to a trade-off between these two factors. In order to speed up the computation, distributed computing and variance reduction techniques are employed. Tests show that the concurrent ARS approach clearly outperforms the standard gradient based method.

Key words: Finance, GARCH, option pricing, accelerated random search, distributed computing

1. Introduction

The literature on quantitative and statistical finance and economics has greatly improved the performance of option pricing models by enriching the return process and return volatility process. In this respect the family of generalized autoregressive conditional heteroskedasticity (GARCH) models captures the stylized facts of stock markets well, and these models are important tools in derivative pricing and risk management (see for example [1, 2, 3, 4, 5], and [6] and the references therein). In addition to option pricing, Bollerslev *et al.* [7] describe several empirical applications based on financial data.

Whereas many current papers aim to improve the volatility *models* for better option pricing performance, the goal of this paper is very different. The aim is to examine *optimization algorithms* for the calibration of GARCH models, i.e. given observations of option prices the goal is to find the best parameters of the GARCH models to fit the data. In particular, a global optimization method to calibrate two GARCH models using information on option prices is introduced. When using optimization techniques to find model parameters that minimize the error between the model prices and market prices of options, the success of the calibration can substantially depend on the optimization algorithm, and the calibration performed with an unsuitable algorithm can lead to parameter instability. In fact, as will be demonstrated, the optimization method used to calibrate a model can become as crucial as the model itself.

Using two widely recognized GARCH specifications, namely the Leverage and the Heston-Nandi model, and a large set of option data, the performance of a concurrent method based on accelerated random search (ARS) is examined and compared to a standard gradient based search method. Whereas for the Heston-Nandi model a closed-form option pricing formula exists, the Leverage specification requires computationally expensive Monte Carlo simulations, in which case standard deterministic gradient based methods often do not yield appropriate results as the derivatives of the pricing error can be difficult to calculate due to Monte Carlo errors.

Quite extensive literature exists where stochastic volatility models are calibrated based on empirical information on option prices using the non-linear least squares (NLS) approach (see for example [1, 8, 9, 10] and the references therein). Some stochastic optimization algorithms have been examined with GARCH models using maximum likelihood estimation (MLE) (see [11]), but only little heed has been paid to the efficiency of the optimization algorithms in the non-linear least squares (NLS) calibration procedure using empirical option data, even though it may be preferable to use the NLS approach rather than MLE for the purpose of option valuation, i.e. to estimate the parameters directly using information on option prices (see for example [1]). In fact, in most papers only the minimization problem to be solved is posed, but the applied optimization routines are not considered or even mentioned. Whereas with MLE local optima can be overcome by running the calibration multiple times with different parameter initial values, this is not necessarily possible with NLS as it is computationally more expensive than MLE, especially if no closed-form solutions are available for option prices. Therefore, fast global optimization algorithms are crucial when calibrating volatility models using option data.

Depending on the underlying model it is possible that several local and/or global optima exist, and thus the application of commonly used optimization routines might not lead to the global optimum, and instead get trapped in a local optimum. As the main contribution, local optimality is addressed by using a concurrent approach in which each optimizer ("particle") executes an accelerated random search [12]. This approach requires the adjustment of several parameters such as the number of particles and the number of iterations used during the optimization procedure, and the number of option price simulation paths (in case of the Leverage model). Those parameters influence the computation time and also the solution quality. In order to reduce the computation time distributed computing has been applied together with the empirical martingale methods described by Duan and Simonato [13] and antithetic variates.

The remainder of this paper is organized as follows. The considered GARCH models, the data set, the estimation methodology, and the optimization procedure are described in Section 2. In Section 3 implementation issues are depicted and simulation results presented. Conclusions and future research topics are outlined in Section 4.

2. Problem Formulation and Solution Algorithm

GARCH models were pioneered by Bollerslev [14] and Engle [15] and are used for modeling financial time series that exhibit time-varying volatility clustering, i.e. periods of swings followed by periods of relative calm. GARCH models are useful in situations where the volatility of returns is a central issue. The family of GARCH models captures empirical properties well, and these models are therefore important tools in asset pricing and risk management. The success of GARCH models motivated their extension to option valuation (see for example [1, 2, 3, 4, 5, 6] and [16] and the references therein).

This section describes the two specific GARCH models considered in the numerical experiments, namely the Leverage and the Heston-Nandi model. A loss function is introduced that describes the difference between empirical and model option prices. In order to minimize this loss function the accelerated random search (ARS) algorithm [12] has been used and will be briefly outlined. At the end of this section the parameter restrictions of both pricing models will be considered and implementation issues detailed.

2.1. Leverage Model

Under the physical probability measure, the logarithmic stock returns and volatility follow

$$\begin{aligned} R_t \equiv \ln\left(\frac{S_t}{S_{t-1}}\right) &= r + \lambda\sqrt{h_t} - \frac{1}{2}h_t + \sqrt{h_t}z_t, \ z_t \sim N(0,1), \\ h_t &= \beta_0 + \beta_1 h_{t-1} + \beta_2 h_{t-1}(z_{t-1} - \theta)^2, \end{aligned}$$

where S_t denotes the spot price at time t, $\lambda > 0$ is the price of the risk, and $r > 0$ denotes the risk-free rate. Moreover, h_t is the squared volatility,

$\beta_0, \beta_1, \beta_2 > 0$ are structural parameters, and $\theta > 0$ represents the coefficient of leverage. In a GARCH context, Duan [3] showed that a locally risk-neutral valuation relationship is satisfied by a risk-neutral measure $\mathbb{Q}$ if

$$\mathbb{E}_t^Q \left[\exp(R_{t+1})\right] = \exp(r)$$

and

$$\mathrm{Var}_t^Q[R_{t+1}] = \mathrm{Var}_t^P[R_{t+1}] = h_t,$$

respectively, where $\mathbb{E}_t^Q[\cdot]$ and $\mathrm{Var}_t^Q[\cdot]$ are the conditional mean and variance under the measure $\mathbb{Q}$, respectively, and $\mathrm{Var}_t^P[\cdot]$ denotes the conditional variance under the original (physical) measure $\mathbb{P}$. This implies that under the risk-neutral measure,

$$R_t = r - \frac{1}{2}h_t + \sqrt{h_t}z_t^*, \tag{1}$$

where $z_t^* \sim N(0,1)$ under $\mathbb{Q}$ (see also [1]). The above is satisfied if

$$z_t = z_t^* - \lambda.$$

Thus, when employing the Leverage model under the risk-neutral probability measure, the volatility dynamic is given by

$$\begin{aligned} h_t &= \beta_0 + \beta_1 h_{t-1} + \beta_2 h_{t-1}(z_{t-1}^* - \lambda - \theta)^2 \\ &= \beta_0 + \beta_1 h_{t-1} + \beta_2 h_{t-1}(z_{t-1}^* - \beta_3)^2, \end{aligned} \tag{2}$$

where $\beta_3 = \lambda + \theta$. The model prices of European call options can then be determined by

$$C(t, S_t, h_t; K, T, r, \boldsymbol{\beta}) = \exp\{-r(T-t)\}\mathbb{E}_t^Q\left[\max\{S_T - K, 0\}\right], \tag{3}$$

where T denotes the time to maturity, S_T the spot price at maturity, K the strike price, and $\boldsymbol{\beta}$ the vector of the structural parameters.

A disadvantage of this model is the lack of a closed-form pricing formula, and therefore Monte Carlo methods have to be applied to obtain the option prices. For a given spot price S_0 and spot volatility h_0 the conditional expectation is calculated by simulating (1) together with (2) iteratively. To speed up the computation, distributed computing [17] has been used together with the empirical martingale methods described by Duan and Simonato [13] and antithetic variates. For estimating the model parameters more than one day of option prices has been used, and hence the volatility of different dates has been linked using historical time series of stock returns. In particular, to update h_{t+1} from h_t using the observed daily returns of the underlying stock, $\tilde{R}$,

$$z^*_{t-1} = \left(\tilde{R}_{t-1} - r + \frac{1}{2}h_{t-1}\right) / \sqrt{h_{t-1}}$$

has been substituted into (2) (see also [1, on-line Appendix]).

2.2. Heston-Nandi Model

Under the physical probability measure, the return and volatility dynamics of the Heston-Nandi model [18] are given by

$$\begin{aligned} R_t \equiv \ln\left(\frac{S_t}{S_{t-1}}\right) &= r + \gamma h_t + \sqrt{h_t} z_t, \;\; z_t \sim N(0,1), \\ h_t &= \beta_0 + \beta_1 h_{t-1} + \beta_2 (z_{t-1} - \theta\sqrt{h_{t-1}})^2, \end{aligned}$$

where γ represents the coefficient of relative risk aversion. Note that the expected rate of return is proportional to the squared volatility rather than volatility. Using the locally risk-neutral valuation relationship, it follows that the volatility process under the risk-neutral probability measure is given by

$$h_t = \beta_0 + \beta_1 h_{t-1} + \beta_2 \left(z^*_{t-1} - \beta_3 \sqrt{h_{t-1}}\right)^2, \tag{4}$$

where $z^*_{t-1} = z_{t-1} + \left(\gamma + \frac{1}{2}\right)\sqrt{h_{t-1}}$ and $\beta_3 = \gamma + \theta + \frac{1}{2}$. This model has a closed-form solution for determining option prices (see [18]).

2.3. Data and Methodology

In this study the model parameters are estimated using empirical information of option prices. Among others, Bakshi *et al.* [16] employ this estimation methodology using loss functions and minimize the pricing error of the daily cross-section of options. In this paper daily data of S&P 500 index call options traded at the Chicago Board Options Exchange, in particular mid-point bid-ask quotes, are used. The option prices are sampled every Wednesday from January 3, 1990 till December 31, 1993, a total of 12,754 observations[1]. The data set has been divided into two subsets. The period between January 3, 1990 and December 31, 1992 has been used exclusively for in-sample calibration. The period from January 1, 1993 to December 29, 1993 has been used as an out-of-sample data set for evaluation of the fit. The S&P 500 index values are not the closing values, but rather from the moment when the option bid-ask quote is recorded. To price options for all maturities and each trading day interest rates using data on daily treasury yield curve rates for 1, 3, 6 and 12 months have been approximated. Dividends are taken into account by calculating the present dividend values until the maturity of each option, and then subtracting them from the spot prices (see also [1, 16]).

There are different approaches to calibrate option pricing models. First, by following [16] models can be estimated day by day using a daily cross-

[1] The data were graciously provided by Peter Christoffersen.

section of option prices and treating the spot variance h_t as a model parameter estimated from option prices. This approach is commonly employed by practitioners who often estimate the coefficients on observed option prices by static daily calibration. A second approach is to use multiple cross-sections of options on a multi-day (year) sample by extracting the spot variance on different dates using observations on stock returns. One may try to minimize the overall squared option pricing error or use a likelihood-based approach. Multi-day data on option prices and returns is widely used in the academic literature to compare different volatility characterizations, see for example [1, 2, 19, 20, 21, 22] and the references therein.

Even though the practitioners often seem to rely on re-calibrations on daily data and the use of large data samples may worsen the performance of Newton methods, there are reasons for preferring a multi-day sample [23]. First, to capture the time variation in the underlying price and volatility we need to use multi-day data. Second, structural parameters are assumed constant, but the daily re-calibrations on single-day option data make the given parameter estimates vary over time. Moreover, according to [24], “given that the number of calibration constraints (option prices) is finite (and not very large), there may be many Lévy triplets which reproduce call prices with equal precision”, which suggest the use of multi-day data. Finally, the academic literature uses multi-day data to assess the long-term performance of the models.

Some exclusionary criteria are used to discard option data that could make the calibration problematic, as is done for example in [1, 3, 16, 18, 25]. First, options with fewer than six days to maturity have been discarded because these options may have liquidity related biases [16]. Second, options whose price is less than \$3/8 have been discarded for reasons of price discreteness. Third, the options whose prices conflict with the lower bound of the no-arbitrage rule

$$\hat{C}_{\text{Bid}} \geq \max(0, S_t - \text{PVDIV} - Ke^{-r(T-t)}),$$

have been discarded. Here, PVDIV denotes the present value of dividends during the life of an option and $T-t$ the time to maturity. Tables 1 (a) and (b) describe the properties of the in-sample and out-of-sample data sets in more detail.

Following [16], the literature has commonly used the actual dividend data in this context. Another approach would be to estimate the dividend yield as a structural parameter or to use the futures prices instead of the spot prices.

For both models four parameters have to be determined and there are, in general, two methods that can be applied. Maximum likelihood estimation aims at finding the parameter values matching the conditional density of the returns. The second approach finds parameters using the non-linear least squares approach that minimizes a loss function that characterizes the mismatch between model and market prices of vanilla options. In this study, the NLS approach is employed, and the goal is to determine the parameters of the models in Sections 2.1 and 2.2, respectively, based on option prices by minimizing the following loss function. Denote $\hat{C}_{i,t_i}$ the true price of option i at time t_i, and $C_{i,t_i} = C(t_i, S_{t_i}, h_{t_i}; K_i, T_i, r_{t_i}, \boldsymbol{\beta})$ the model price of option i at time t_i. Then the objective is to minimize the pricing error of all options with

Table 1: The number of contracts of the in-sample and out-of-sample data sets are reported by grouping the data into three groups based on maturity (trading days) and into six groups based on moneyness (S/K).

(a) In-Sample Data

Moneyness	Maturity (days to expiration)			
S/K	< 60	60 - 180	≥ 180	Total
< 0.94	189	682	552	1423
$0.94 - 0.97$	489	555	266	1310
$0.97 - 1.00$	636	547	241	1424
$1.00 - 1.03$	601	522	250	1373
$1.03 - 1.06$	565	460	151	1176
≥ 1.06	987	986	497	2470
Total	3467	3752	1957	9176

(b) Out-of-Sample Data

Moneyness	Maturity (days to expiration)			
S/K	< 60	60 - 180	≥ 180	Total
< 0.94	2	155	138	295
$0.94 - 0.97$	129	219	59	407
$0.97 - 1.00$	247	237	53	537
$1.00 - 1.03$	253	232	61	546
$1.03 - 1.06$	233	204	38	475
≥ 1.06	535	539	244	1318
Total	1399	1586	593	3578

respect to the structural parameters, i.e. find the values of $\boldsymbol{\beta}$, that minimize

$$\text{RMSE} = \sqrt{\frac{1}{N_T}\sum_{t,i}\left(\hat{C}_{i,t} - C_{i,t}\right)^2}, \tag{5}$$

where $N_T = \sum_{t=1}^{T} N_t$ and N_t is the number of option prices in the sample at time t, and the prices $C_{i,t}$ are calculated according to (3) and the solution in [18], respectively.

2.4. Accelerated Random Search (ARS)

The performance of commonly used (local) optimization methods usually depends on the starting values for the parameters and the solution often requires prohibitively many function evaluations. In order to overcome problems of local optimality and to thoroughly explore the variable domain, the accelerated random search (ARS) algorithm [12] has been employed for minimizing (5). The ARS algorithm is for convenience outlined in the following. Let $\tilde{D}$ denote a $d-$dimensional unit hypercube $[0,1]^d$, and let $\|\cdot\|$ denote the 2-norm on $\tilde{D}$. Denote by $B(x,\tau) = \{y \in \tilde{D} : \|x-y\| \leq \tau\}$ the closed ball centered at x with radius τ. With the contraction factor $c > 1$ and the precision threshold $\rho > 0$ given, the following steps are executed when minimizing the objective function.

1. Set $n \leftarrow 1$, $\tau_1 \leftarrow 1$ and generate a sample X_1 from a given distribution on $\tilde{D}$.

2. Given $X_n \in \tilde{D}$ and $\tau_n \in (0,1]$, generate a sample Y_n from a given distribution on $B(X_n, \tau_n)$.

3. If $f(Y_n) < f(X_n)$, set $X_{n+1} \leftarrow Y_n$ and $\tau_{n+1} \leftarrow 1$.
   Else, set $X_{n+1} \leftarrow X_n$ and $\tau_{n+1} \leftarrow \tau_n/c$.
   If $\tau_{n+1} < \rho$, then $\tau_{n+1} \leftarrow 1$.
   Increment $n \leftarrow n+1$, and go to Step 1.

Step 3 implies that in case the new candidate Y_n is not better than the current X_n, the radius of the ball shrinks, and thus the search space shrinks. On the other hand, if Y_n delivers a better function value than X_n, the search space is reinitialized to the whole domain $\tilde{D}$. Since the radius is decreased whenever the new candidate is not better than the current candidate, the search space will shrink quickly until sampling happens only in the neighborhood of a local optimum. After a certain number of shrinks, the search space is reinitialized to the whole space, and thus escape from local optima is possible.

This algorithm can be implemented in a concurrent framework so that in every iteration $\mathcal{M}$ optimizers ("particles") execute an ARS, each starting with a different sample. Initially $\mathcal{M}$ samples are drawn from the search domain, i.e. generate $\mathbf{x}_{1,1}, \ldots, \mathbf{x}_{1,\mathcal{M}}$, where $\mathbf{x}_{1,i}$ is a vector containing the required parameters, and $\boldsymbol{\tau}$ is an $\mathcal{M}$-vector of ones. In the nth step new samples $\mathbf{y}_{n,1}, \ldots, \mathbf{y}_{n,\mathcal{M}}$ are generated from the hyper-spheres $B(\mathbf{x}_{n,i}, r_{n,i})$, where $\mathbf{x}_{n,i}$ denotes the ith particle, and $r_{n,i}$ is the corresponding radius. The conditions (6) and (7) described in Section 2.5 must be checked for validity. All particles $j \in \mathcal{J} \subseteq \mathcal{M}$ that do not fulfill the conditions are discarded and replaced by new samples from the corresponding spheres $B(\mathbf{x}_{n,j}, r_{n,j})$, $j \in \mathcal{J}$.

In Step 3 the root mean squared errors for the current sample points and the new candidate points are calculated and compared. Here the root mean squared error corresponds to the objective function $f(\cdot)$ in Step 3 of the ARS algorithm. The particles with the better, i.e. lower, RMSE are kept, and Steps 2 and 3 are repeated for a given number of iterations I. The best RMSE obtained during the simulation and the corresponding parameter settings are finally used for calculating the out-of-sample errors.

The ARS algorithm applies, similar to simulated annealing (SA) [26, 27] and evolutionary algorithms (EA) [28, 29, 30], a neighborhood search strategy. However, in contrast to SA and EA, ARS never accepts solutions that are worse than the current solution. The solution improves or stays at least the same from one iteration to another. ARS tries to overcome local optimality by reinitializing the search radius to the whole variable domain while SA and EA accept worse solutions in order to escape from local optima. SA and EA restrict the search space to a set of neighboring solutions and the algorithms stop after several unsuccessful improvement trials. ARS on the other hand restarts the search similar to a multi-start optimization procedure and continues the search until the predefined number of iterations has been reached or some other stopping criterion has been met. Thus, ARS explores the parameter domain better and is for this reason applied in this paper.

2.5. Parameter Restrictions

The parameters β_i, $i = 0, 1, 2, 3$, of the Leverage model in (2) have the following stationary restrictions that must be taken into account during the optimization process:

$$\beta_i \geq 0, \forall i = 0, 1, 2, 3, \quad \beta_1 + \beta_2(1 + \beta_3^2) \leq 1. \tag{6}$$

Since there are four parameters to be adjusted the problem is of dimension $d = 4$. The conditions on the parameters do not define a unit cube, so in order to apply the ARS algorithm, each parameter must be scaled to the unit interval so that $\tilde{D} = [0, 1]^4$. Since all parameters are bounded to be non-negative, only the upper bounds need to be adjusted. Denote β_0^u, β_1^u, β_2^u and β_3^u the upper bounds for parameters β_0, β_1, β_2, and β_3, respectively.

In the first step of the ARS algorithm $\mathcal{M}$ particles of the form $\mathbf{x}_{1,i} = \left[\tilde{\beta}_0, \tilde{\beta}_1, \tilde{\beta}_2, \tilde{\beta}_3\right]_i'$ are uniformly drawn from the unit hypercube $\tilde{D}$, where the subscript i denotes the parameters of the ith particle. Thus, the parameter values that must be checked with respect to conditions (6) are $\beta_j = \tilde{\beta}_j \beta_j^u$, $j = 0, 1, 2, 3$, for every particle. The upper bounds for the parameter values are except for β_0 implied by conditions (6):

- For β_1^u the conditions imply that $\beta_1 \leq 1$, and thus $\beta_1^u = 1$.
- Similarly, the conditions require $\beta_2 \leq 1$, and thus $\beta_2^u = 1$.
- For β_1^u and β_2^u the conditions demand that $\beta_1 + \beta_2 \leq 1$, and thus β_1^u and β_2^u are not independent. Drawing $\tilde{\beta}_1$ before $\tilde{\beta}_2$ from the given distribution implies that $\beta_2^u = 1 - \beta_1$, and drawing first $\tilde{\beta}_2$ from the given distribution results in the upper bound $\beta_1^u = 1 - \beta_2$.
- The latter condition in (6) implies a dependency of the parameters β_1, β_2 and β_3. Thus, sampling at first $\tilde{\beta}_1$ and $\tilde{\beta}_2$ from $\tilde{D}$ implies that $\beta_3^u = \sqrt{\frac{1-\beta_1}{\beta_2} - 1}$.

An upper bound for β_0 is not implied by the conditions. Computational experiments showed that especially small values for β_0 lead to good solutions. Thus, $\beta_0^u = 10^{-5}$ was used throughout the simulations. Assuming the parameters for every particle are sampled in the sequence $\tilde{\beta}_0 \rightarrow \tilde{\beta}_1 \rightarrow \tilde{\beta}_2 \rightarrow \tilde{\beta}_3$, the unscaled parameter domain is a cuboid $D = [0, 10^{-5}] \times [0, 1] \times [0, 1 - \beta_1] \times \left[0, \sqrt{\frac{1-\beta_1}{\beta_2} - 1}\right]$.

The parameter restrictions for the Heston-Nandi model are given by

$$\beta_i \geq 0, \forall i = 0, 1, 2, 3, \quad \beta_1 + \beta_2 \beta_3^2 \leq 1, \tag{7}$$

which implies that $b_3^u = \sqrt{\frac{1-\beta_1}{\beta_2}}$, and that there is no dependency of β_1 and β_2 as is in the Leverage model. Thus, an upper bound for β_2 must be defined. Experiments showed that also for this variable small values give good results, and therefore simulations using $\beta_2^u = 10^{-4}$ have been done. Assuming again that the parameters for every particle are sampled in the sequence $\tilde{\beta}_0 \to \tilde{\beta}_1 \to \tilde{\beta}_2 \to \tilde{\beta}_3$, the cuboid is defined by $D = [0, 10^{-5}] \times [0, 1] \times [0, 10^{-4}] \times \left[0, \sqrt{\frac{1-\beta_1}{\beta_2}}\right]$.

3. Numerical Experiments

The algorithm has been implemented in Matlab and tested on the data set described in Section 2.3. The contraction factor c and the precision threshold ρ of the ARS algorithm have been set to 2 and 2^{-20}, respectively. The simulations have been distributed over the 800-core PC grid at Tampere University of Technology. For the Leverage model the calculations for each particle at each step of the ARS iteration have been distributed on two levels: the options are distributed, and the Monte Carlo price paths for each option are distributed (Figure 1). For the Heston-Nandi model the price path simulations are not needed and the computation is only distributed at the option level.

The CPU time as well as the solution quality are dependent on the number of particles used during the calibration, the number of iterations I of the ARS algorithm, the number of options N_t in the data sample, and in case of the Leverage model, on the number of price simulation paths P. The number of particles influences the exploration of the parameter domain, i.e. the more particles used, the higher the chance to find a good solution. As the number of particles increases also the computation times increase, and a trade-off

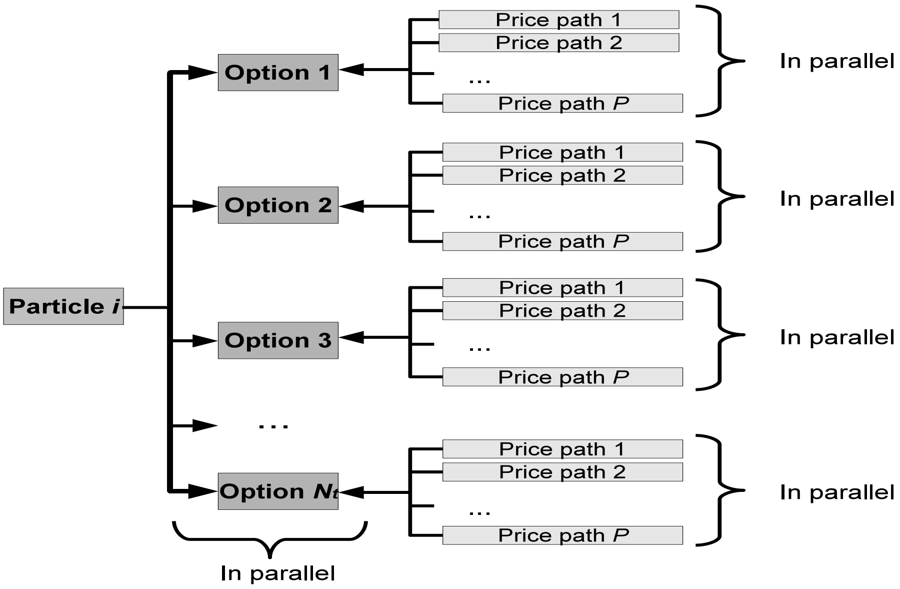

Figure 1: Leverage model: distributed computation for ith particle

between solution quality and computation time arises.

The number of iterations I determines the number of attempts of the ARS algorithm to improve the current RMSEs of all particles. However, with an increasing number of iterations it becomes harder to find further improvements of the loss function. Thus, also the adjustment of I must be considered. If improvements of the RMSE is not ”reasonable” compared to the number of additional iterations, the total number of iterations should not be further increased. For the Leverage model, the number of price simulation paths also influences the computation time.

The following paragraphs explore in more detail the trade-off of the number of particles vs. solution quality and number of price simulation paths vs. solution quality (Leverage model), respectively. In order to examine the influence of the number of particles on the RMSE, the number of iterations and price simulation paths was fixed to $I = 100$ and $P = 10000$ (for the Leverage model), respectively.

3.1. The Influence of Particle Number on Solution Quality

Since the Leverage and Heston-Nandi model include due to the sampling of the particles (and in case of the Leverage model also due to the simulation of the option price paths) random components, the optimization algorithm was run three times for each parameter setting.

3.1.1. Leverage Model

The wall-clock computation time for the model calibration greatly depends on the availability of computers for the distributed computing. Since the grid uses the idle computers in the PC labs of Tampere University of Technology, the capacity varied between and also within simulations. Thus, the wall-clock time for the simulations varied strongly. For example, for 40 particles the time varied between 26 and 94 hours. Due to the setup of the computing environment, the wall-clock time is not a good measure of performance because it heavily depends on the amount of resources that can be used and varies between different times of the day and also between different weekdays. A detailed wall-clock time examination should be done in a controlled environment where the number of working cores does not change.

The results of the simulations (Table 2) show that the RMSE (averaged over three runs) is lowest when 30 particles were used, and the difference to the other solutions is very small. The optimization runs with 60 particles returned the best results in terms of the average out-of-sample errors (OOSE).

Table 2: Leverage model: average RMSE after 100 iterations and average OOSE with different numbers of particles

# Particles	10	20	30	40	50	60
avg. RMSE	1.7805	1.7760	1.7635	1.7811	1.7696	1.7915
avg. OOSE	2.2346	2.4323	2.3453	2.1251	2.2064	1.9280

Table 3 shows for each particle number the minimal RMSE and the corresponding OOSE attained during all three runs as well as the corresponding parameter values with their standard errors in parentheses. The results show that the lowest RMSE has been reached when using 60 particles. On the other hand, the best out-of-sample error has been reached when using 40 particles. Noticeable is the magnitude of the parameter β_2 which represents the GARCH effect. The best value is about 0.02 which may indicate a weak model identification [31].

The average RMSEs of all particles in every iteration are illustrated in Figure 2. The figure shows that the graphs of the average RMSE have the shapes of "steps" with the step height decreasing as the number of iterations increases. The average RMSEs of all particle numbers are highest in the first iterations, but quickly decrease until iteration 15. A further noticeable decrease of the average RMSE can be seen between iterations 35 and 55. After iteration 60 the RMSE reduces only slightly and after iteration 90 the average RMSE is about equal for all particle numbers. The figure indicates that 100 iterations are sufficient since the rate of improvement becomes insignificant

Table 3: Leverage model: best parameter settings with different numbers of particles

# Particles	min. RMSE	min. OOSE	β_0	β_1	β_2	β_3
10	1.4903	2.0400	1.2783e-6	0.9435	0.0388	0.5141
			(8.2e-8)	(4.1e-3)	(3.7e-3)	(4.8e-2)
20	1.4143	2.0965	1.8939e-6	0.8081	0.0885	1.0528
			(6.4e-8)	(5.5e-3)	(5.8e-3)	(5.3e-2)
30	1.4230	2.2732	2.6424e-6	0.7986	0.0892	1.0542
			(1.1e-7)	(7.1e-3)	(7.4e-3)	(6.5e-2)
40	1.1459	1.6807	7.6875e-7	0.9058	0.0223	1.7748
			(4.2e-8)	(3.5e-3)	(1.3e-3)	(7.9e-2)
50	1.3646	2.2216	2.1238e-6	0.8501	0.0481	1.3645
			(1.0e-7)	(5.4e-3)	(4.2e-3)	(9.3e-2)
60	1.1423	1.6980	8.0677e-7	0.8869	0.0226	1.9812
			(4.1e-8)	(4.0e-3)	(1.2e-3)	(8.5e-2)

after 90 iterations.

Figure 3 illustrates the relative improvement of the average RMSE from one iteration to the next for all particle numbers. It can be seen that iterations with several improvements and iterations with almost no improvements are alternating. For example, until iteration 15 many improvements have been found. Between iterations 20 and 30 only very small improvements could be attained while after iteration 30 again higher improvements can be seen. This phenomenon can be related to the search radius of the ARS algorithm. This radius is reduced whenever no better solution has been obtained, and until a better solution has been found, the old solution stays. If that happens for several particles at the same time, the average RMSE changes only very slightly. Thus, between iterations 20 and 30, only very few improvements have been found, and after iteration 30, after the search radius has been reinitialized, the particles had more chances to find improved solutions. On the other hand, the magnitude of the relative improvements decreases as well. The improvements between iterations 1 and 15 are highest, while those between iterations 30 and 60 are already lower. Finally the improvements between iterations 75 and 100 are lowest (not taking into account the almost zero-improvements in the remaining iterations). From this consideration it seems reasonable to stop the algorithm after about 60 iterations.

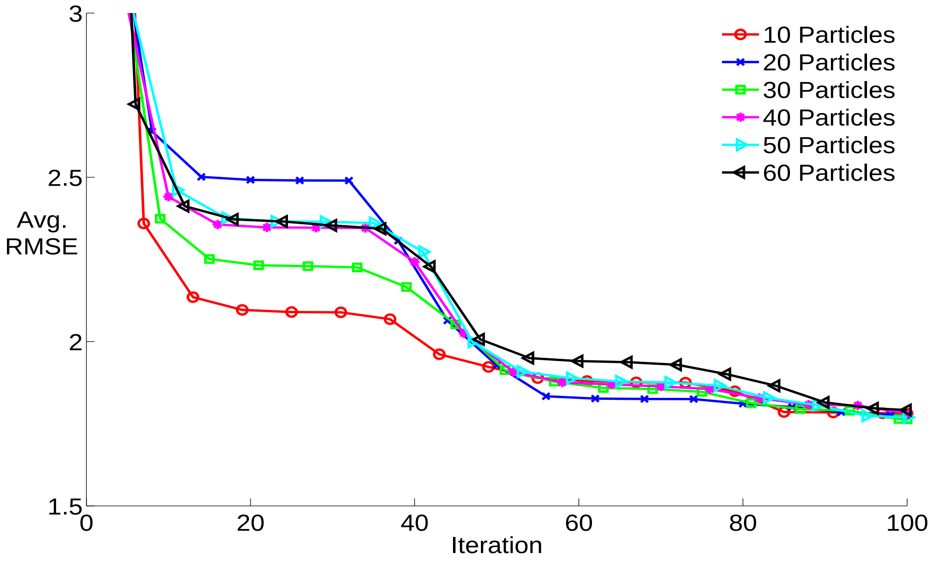

Figure 2: Leverage model: average RMSE vs. iteration number

3.1.2. Heston-Nandi Model

Since the Heston-Nandi model has a closed form option pricing formula, Monte Carlo simulations are not necessary, and therefore only the number of

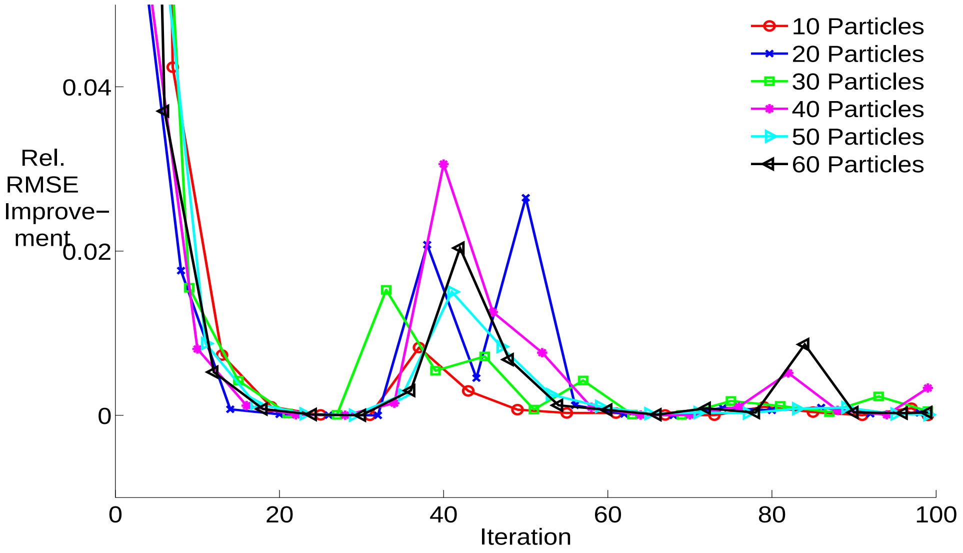

Figure 3: Leverage model: relative improvement of average RMSE between iterations

iterations must be fixed ($I = 100$ has been used) while changing the number of particles. The results obtained for the Heston-Nandi model are summarized in Table 4 and show that the RMSE values do not change significantly with the number of particles, but the OOSE decreases as the number of particles increases. The RMSE is in general higher for the Heston-Nandi model than for the Leverage model. Note also that not always the lowest RMSE implies the lowest OOSE. Table 5 shows the minimal RMSE and OOSE.

The average RMSEs over all particles in each iteration are illustrated in Figure 4. A comparison with Figure 2 shows that the development of the RMSE is for all particle numbers very similar. In the first 15 iterations significant RMSE reductions can be observed while between iterations 15 and 30 almost no changes are visible. A further small improvement can

Table 4: Heston-Nandi model: average RMSE after 100 iterations and average OOSE with different numbers of particles

# Particles	10	20	30	40	50	60
avg. RMSE	1.9505	1.9530	1.9562	1.9526	1.9866	1.9473
avg. OOSE	2.9122	2.8685	2.7257	2.8053	2.7674	2.5714

be observed between iterations 30 and 50, but after iteration 60 the RMSE stays approximately constant for all particle numbers. Figure 5 illustrates the relative improvement of the average RMSE from one iteration to the next. Also this illustration shows that after iteration 60 almost no improvements could be achieved.

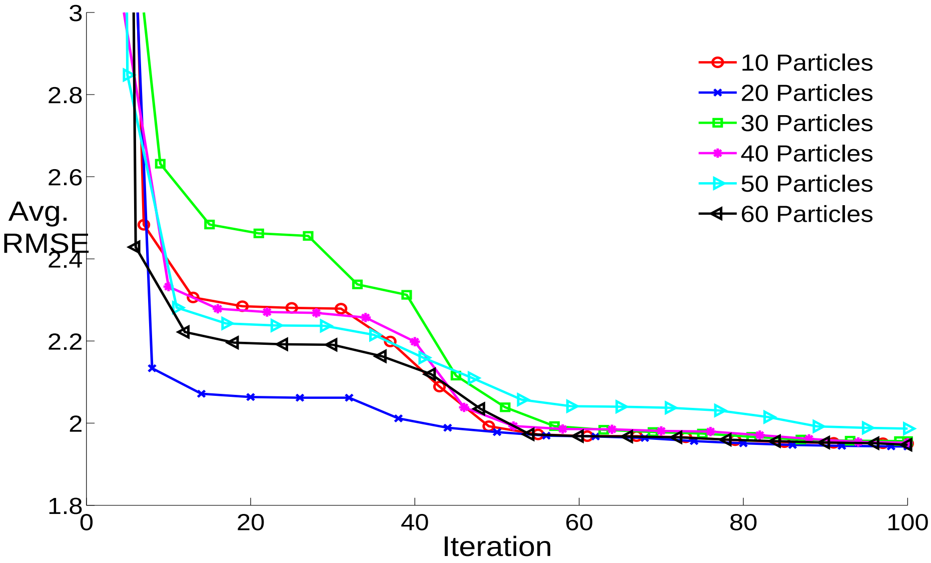

Figure 4: Heston-Nandi model: average RMSE vs. iteration number model

Table 5: Heston-Nandi model: best parameter settings with different numbers of particles

# Particles	min. RMSE	min. OOSE	β_0	β_1	β_2	β_3
10	1.7478	3.0401	4.4388e-6	0.3153	8.9365e-6	253.3633
			(1.1e-6)	(8.7e-2)	(1.5e-6)	(33.3)
20	1.7758	2.7094	1.4968e-6	0.2726	1.7707e-5	177.1633
			(1.2e-6)	(7.5e-2)	(1.3e-6)	(11.9)
30	1.7290	2.7363	5.0699e-6	0.1999	8.7936e-6	277.7067
			(1.2e-6)	(0.1)	(1.6e-6)	(40.4)
40	1.7458	2.7904	4.9489e-6	0.3420	9.9591e-6	229.8892
			(1.3e-6)	(9.2e-2)	(2.0e-6)	(34.6)
50	1.5672	2.6616	9.4076e-7	0.7282	3.9952e-6	240.8244
			(2.4e-7)	(2.0e-2)	(4.1e-7)	(18.8)
60	1.3970	2.2835	1.7733e-7	0.6924	2.7526e-6	322.0671
			(4.8e-7)	(0.1)	(5.0e-7)	(59.7)

3.2. Influence of Price Simulation Path Number on Solution Quality

For the Leverage model the option prices must be calculated by Monte Carlo simulations. The implications of the number of price simulation paths on the objective function values are examined below. The number of particles was set to $\mathcal{M} = 60$. The number of iterations has been fixed to $I = 100$ while the number of price simulation paths has been varied between $P = 50$ and $P = 10000$. The results are summarized in Tables 6 and 7, respectively.

The results in Table 6 indicate that the lowest average RMSE was obtained when using 1000 price simulation paths. The out-of-sample error is lowest when 50 simulation paths are used, but the differences between the out-of-sample errors for different price simulation path numbers are rather

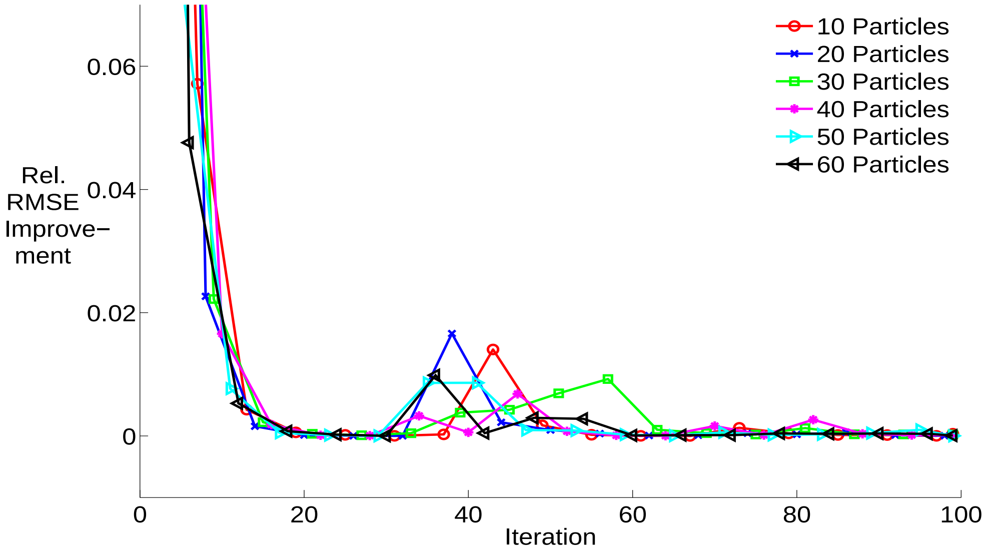

Figure 5: Heston-Nandi model: relative improvement of average RMSE between iterations model

small. Table 7 shows the minimal RMSE and OOSE achieved during all three simulation runs. The best results were achieved by the 5000 path case.

The average RMSEs for all price simulation path numbers are illustrated in Figure 6. It can be seen that the 50 price path case has the highest average RMSE. Figure 7 illustrates the relative RMSE improvement from one iteration to the next. The major improvements are found during the first 15 iterations, and between iterations 40 and 60.

3.3. Comparison to Standard Optimization Routines

For reasons of comparison, the GARCH models have also been calibrated using a gradient based method (fmincon in Matlab's optimization toolbox). For this purpose different particle numbers have been used and each particle

Table 6: Leverage model: average RMSE after 100 iterations and OOSE with different numbers of price simulation paths

# Paths	50	100	500	1000	5000	10000
avg. RMSE	1.8489	1.8493	1.7843	1.7698	1.7971	1.7915
avg. OOSE	1.9180	2.2994	2.1939	2.2315	2.0095	1.9280

executes the optimization using fmincon. Since the number of iterations has been limited to 100 for the ARS approach, the option *MaxFunEvals* of fmincon has been set to 100 so as to compare the final results of both optimization methods after an equal number of function evaluations. The fmincon computations have been distributed at the particle level.

The initial guess for the solution required by fmincon has been obtained in the same way as for the ARS approach for every particle. Constraints (6) and (7) have been supplied as non-linear inequality constraints, the lower parameter bounds have been set to $[\beta_0^l, \beta_1^l, \beta_2^l, \beta_3^l] = [0, 0, 0, 0]$, and the upper bounds were adjusted to $[\beta_0^u, \beta_1^u, \beta_2^u, \beta_3^u]_L = [10^{-5}, 1, 1, 10]$ for the Leverage model, and to $[\beta_0^u, \beta_1^u, \beta_2^u, \beta_3^u]_{\mathrm{HN}} = [10^{-5}, 1, 10^{-4}, 10]$ for the Heston-Nandi model.

The resulting RMSEs of all simulation runs are summarized in Tables 8(a) and 8(b). The results show that for both models fmincon delivers worse results than the ARS optimization approach. The results for the Leverage model (Table 8(a)) show that the average and minimal RMSE found by fmincon reduce as the number of particles increases. However, for the Heston-Nandi model the average RMSE does not continuously decrease with an increasing number of particles (see Table 8(b)). Figure 8 shows the average

Table 7: Leverage model: best parameter settings with different numbers of price simulation paths

# Paths	min. RMSE	min. OOSE	β_0	β_1	β_2	β_3
50	1.4700	1.6946	1.0667e-6	0.8835	0.0585	0.9630
			(4.1e-8)	(3.3e-3)	(2.8e-3)	(4.4e-2)
100	1.4641	2.4881	2.7314e-6	0.7557	0.0390	2.2115
			(1.5e-7)	(1.3e-2)	(3.5e-3)	(0.1)
500	1.3296	1.8864	1.2119e-6	0.8843	0.0352	1.4455
			(5.8e-8)	(4.3e-3)	(2.6e-3)	(9.0e-2)
1000	1.3466	2.1235	1.7916e-6	0.8573	0.0597	1.1023
			(7.7e-8)	(4.6e-3)	(4.6e-3)	(6.7e-2)
5000	1.1254	1.6754	5.4416e-7	0.9274	0.0132	2.1017
			(5.5e-8)	(3.5e-3)	(9.0e-4)	(0.1)
10000	1.1423	1.6980	8.0677e-7	0.8869	0.0226	1.9812
			(4.1e-8)	(4.0e-3)	(1.2e-3)	(8.5e-2)

RMSE values for both optimization algorithms and both models after an equal number of function evaluations (see Tables 2, 4, and 8 for the data). In general, when comparing the solution quality of the ARS approach and fmincon, as summarized in Table 9 and illustrated in Figure 8, the ARS approach should clearly be favored.

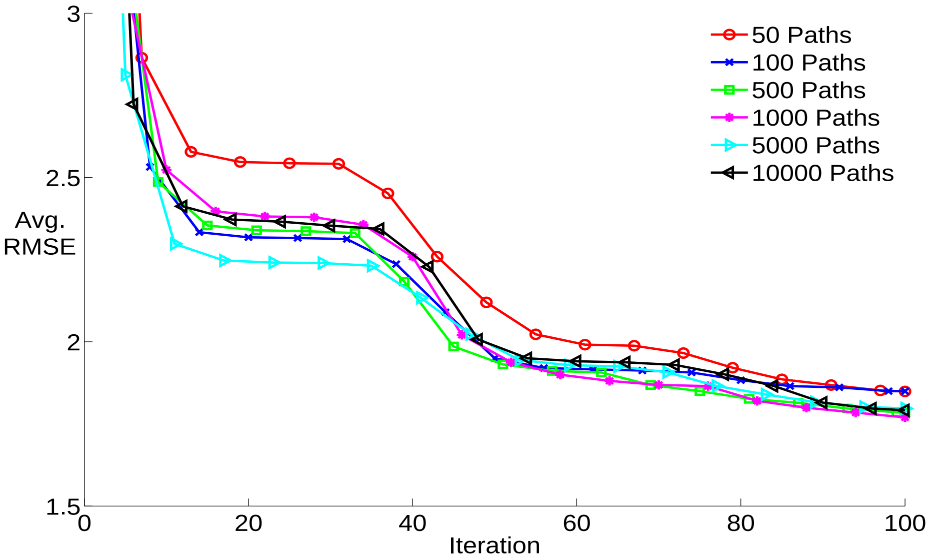

Figure 6: Leverage model: average RMSEs with different price simulation path numbers

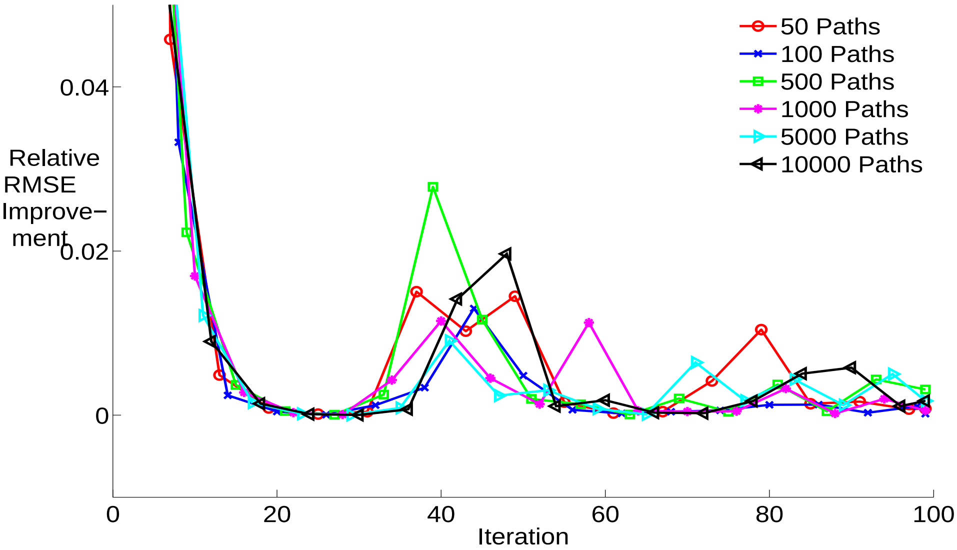

Figure 7: Leverage model: relative Improvement of average RMSE between iterations with different price simulation path numbers

Table 8: Average RMSEs for Leverage and Heston-Nandi model with fmincon

(a) Leverage Model

#Particles	10	20	30	40	50	60
avg. RMSE	7.4405	5.9206	4.3440	3.6893	3.6459	3.3123
min. RMSE	5.1619	3.9771	3.4096	3.1463	2.9109	2.7620

(b) Heston-Nandi Model

#Particles	10	20	30	40	50	60
avg. RMSE	13.1037	10.4078	8.6576	9.3890	6.5176	7.9945
min. RMSE	10.5699	7.6385	6.3539	5.5625	5.1313	4.7738

Table 9: Summary of minimum RMSE reached by ARS and fmincon for Leverage and Heston-Nandi model, respectively

	# Particles	10	20	30	40	50	60
Leverage	ARS	1.4903	1.4143	1.4230	1.1459	1.3646	1.1423
	fmincon	5.1619	3.9771	3.4096	3.1463	2.9109	2.7620
Heston-Nandi	ARS	1.7478	1.7758	1.7290	1.7458	1.5672	1.3970
	fmincon	10.5699	7.6385	6.3539	5.5625	5.1313	4.7738

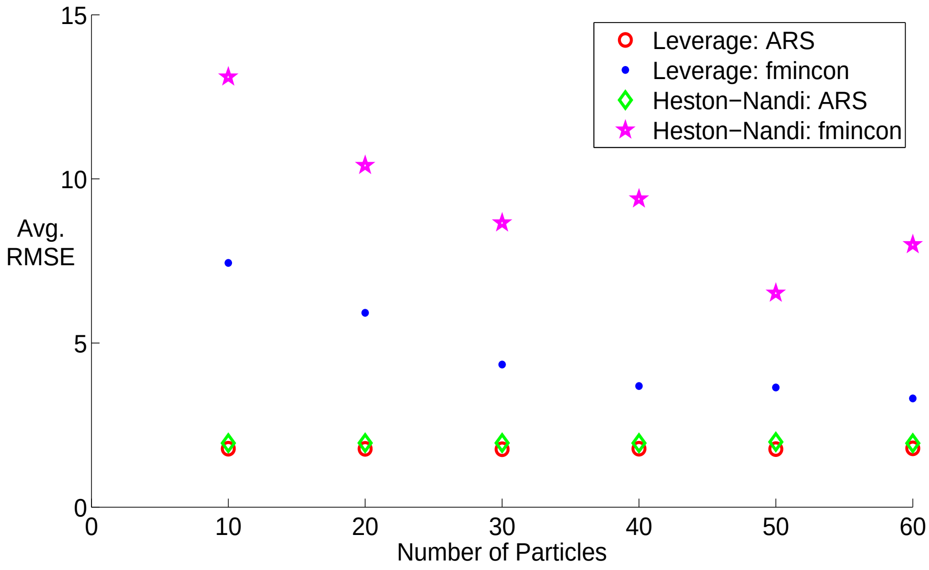

Figure 8: Average RMSE for Leverage and Heston-Nandi model with ARS and fmincon

4. Conclusions

In this paper a global optimization approach for the calibration of GARCH models has been examined. The Leverage and the Heston-Nandi model have been calibrated by minimizing a loss function using the accelerated random search approach. The number of concurrent optimizers ("particles") used and, for the Leverage model, the number of price simulation paths have been varied in order to investigate their effects on the solution quality and computation time. A gradient based method, Matlab's optimization routine fmincon, has as well been used for the model calibration for comparison. The computation has been sped up by applying distributed computing, empirical martingale simulation, and antithetic variables.

The results show that the ARS optimization is a valid approach for calibrating GARCH models using information on option prices, and that the solution quality is clearly better compared to the gradient based method (fmincon). In general it can be concluded that the ARS approach has better parameter domain exploration qualities than standard optimization routines even if a multi-start procedure is applied with the latter. Although implemented for the specific models considered in this paper, the parallel ARS optimization method can easily be generalized and applied to a wide variety of optimization problems.

The application of ARS to the maximum likelihood method for model calibration should be considered in future research work. Also, since for the Leverage model the computation time to obtain the RMSE for one parameter set tends to become prohibitively high as the number of price simulation paths increases, it would be reasonable to investigate the use of surrogate model based algorithms [32].

Acknowledgement

The authors thank the reviewers for their helpful comments and improvement suggestions.

References

[1] P. Christoffersen, K. Jacobs, Which GARCH model for option valuation?, Management Science 50 (2004) 1204–1221.

[2] P. Christoffersen, K. Jacobs, C. Ornthanalai, Y. Wang, Option Valuation wilth Long-run and Short-run Volatility Components, Technical Report, CREATES Research Paper 2008-11, 2008.

[3] J.-C. Duan, The GARCH option pricing model, Mathematical Finance 5 (1995) 13–32.

[4] J.-C. Duan, P. Ritchken, Z. Sun, Jump starting GARCH: Pricing and hedging options with jumps in returns and volatilities., 2006. Working Paper, Federal Reserve Bank of Cleveland, No. 0619.

[5] R. Garcia, E. Renault, A note on hedging in ARCH and stochastic volatility option pricing models, Mathematical Finance 8 (1998) 153–161.

[6] P. Ritchken, R. Trevor, Pricing options under generalized GARCH and stochastic volatility processes, Journal of Finance 54 (1999) 377–402.

[7] T. Bollerslev, R. Chou, K. Kroner, ARCH modeling in finance: A review of the theory and empirical evidence, Journal of Econometrics 52 (1992) 5–59.

[8] G. Barone-Adesi, R. F. Engle, L. Mancini, A GARCH option pricing model with filtered historical simulation, Review of Financial Studies 21 (2008) 1223–1258.

[9] M. Chernov, E. Ghysels, A study towards a unified approach to the joint estimation of objectove and risk neutral measures for the purpose of options valuation, Journal of Financial Economics 56 (2000) 407–458.

[10] M. L. Guerra, L. Sorini, Testing robustness in calibration of stochastic volatility models, European Journal of Operational Research 163 (2005) 145–153.

[11] M. E. Jerrel, W. A. Campione, Global optimization of econometric functions, Journal of Global Optimization 20 (2001).

[12] M. J. Appel, R. Labarre, D. Radulović, On accelerated random search, SIAM Journal on Optimization 14 (2003) 708–731.

[13] J.-C. Duan, J.-G. Simonato, Empirical martingale simulation for asset prices, Management Science 44 (1998) 1218–1233.

[14] T. Bollerslev, Generalized autoregressive conditional heteroscedasticity, Journal of Econometrics 31 (1986) 307–327.

[15] R. F. Engle, Autoregressive conditional heteroscedasticity with esti-

mates of the variance of United Kingdom inflation, Econometrica 50 (1982) 987–1007.

[16] G. Bakshi, C. Cao, Z. Chen, Empirical performance of alternative option pricing models, The Journal of Finance 52 (1997) 2003–2049.

[17] J. Kanniainen, R. Piché, T. Mikkonen, Use of distributed computing in derivative pricing, International Journal of Electronic Finance 3 (2009) 270–283.

[18] S. L. Heston, S. Nandi, A closed-form GARCH option valuation model, The Review of Financial Studies 13 (2000) 585–625.

[19] D. S. Bates, Post-'87 crash fears in the S&P500 futures option market, Journal of Econometrics 94 (2000) 181–238.

[20] P. Christoffersen, K. Jacobs, C. Ornthanalai, Dynamic jump intensities and risk premiums: Evidence from S&P500 returns and options, Journal of Financial Economics 106 (2012) 447–472.

[21] B. Eraker, Do stock prices and volatility jump? Reconciling evidence from spot and option prices, Journal of Finance 59 (2004) 1367–1403.

[22] J. Huang, L. Wu, Specification analysis of option pricing models based on time-changed Lévy processes, Journal of Finance 39 (2004) 1405–1440.

[23] J. Kanniainen, R. Piché, Stock price dynamics, dividends, and option prices with volatility feedback, Physica A 392 (2013) 722–740.

[24] R. Cont, P. Tankov, Non-parametric calibration of jump-diffusion option pricing models, Journal of Computational Finance 7 (2004) 1–50.

[25] P. Christoffersen, S. Heston, K. Jacobs, Option valuation with conditional skewness, Journal of Econometrics 131 (2006) 253–284.

[26] S. Kirkpatrick, C. Gelatt, M. Vecchi, Optimization by Simulated Annealing, Science 220 (1983) 671–680.

[27] V. Černý, A thermodynamical approach to the travelling salesman problem: an efficient simulation algorithm, Journal of Optimization Theory and Applications 45 (2006) 41–51.

[28] J. Holland, Adaption in natural and artificial systems, Ann Arbor: The University of Michigan Press, 1975.

[29] I. Rechenberg, Evolutionsstrategie - Optimierung technischer Systeme nach Prinzipien der biologischen Evolution, Stuttgart: Frommann-Holzboog, 1973.

[30] H.-P. Schwefel, Numerical optimization of computer models, Chichester: Wiley & Sons, 1981.

[31] J. Ma, C. Nelson, R. Startz, Spurious inference in the GARCH (1,1) model when it is weakly identified, Studies in Nonlinear Dynamics & Econometrics 11 (2007).

[32] J. Müller, R. Piché, Mixture surrogate models based on Dempster-Shafer theory for global optimization problems, Journal of Global Optimization 51 (2010) 79–104.