ivwKöln
Institut für Versicherungswesen

Forschung am IVW Köln, 3/2015

Institut für Versicherungswesen

Calibration of Heston's stochastic volatility model to an empirical density using a genetic algorithm

Urij Dolgov

Fachhochschule Köln
Cologne University of Applied Sciences

Forschung am IVW Köln, 3/2015

Dolgov
Forschungsstelle FaRis

Calibration of Heston's stochastic volatility model to an empirical density using a genetic algorithm

Zusammenfassung

In diesem Artikel schlagen wir die Verwendung eines genetischen Algorithmus (GA) zur Kalibrierung eines Stochastischen Prozesses an eine empirische Dichte von Aktienrenditen vor. Anhand des Heston Models zeigen wir wie eine solche Kalibrierung durchgeführt werden kann. Neben des Pseudocodes für einen einfachen aber leistungsfähigen GA präsentieren wir zudem auch Kalibrierungsergebnisse für den DAX und den S&P 500.

Abstract

In this paper we propose the use of genetic algorithms when fitting a stochastic process to the empirical density of stock returns. Using the Heston Model as an example, we show how such a calibration can be carried out. We also present an easy to implement genetic algorithm and provide calibration results for the daily stock returns of the DAX and the S&P 500.

Schlagwörter:

Aktienrenditen, Dichtefunktion, empirische Dichte, Heston Model, Modellkalibrierung, Stochastische Prozesse

Keywords:

Empirical Density, Heston Model, Model Calibration, Probability Density, Stochastic Processes, Stock Returns

Calibration of Heston's stochastic volatility model to an empirical density using a genetic algorithm

Urij Dolgov

23 December 2014

Abstract

In this paper we propose the use of genetic algorithms when fitting a stochastic process to the empirical density of stock returns. Using the Heston Model as an example we show how such a calibration can be carried out. We also present an easy to implement genetic algorithm and provide calibration results for the daily stock returns of the DAX and the S&P 500.

1 The Heston Model and it's transition density

The Heston Model (HM) suggested by Heston (1993) is often seen as the first logical extension of the widely known Black and Scholes (BS) approach. It uses a stochastic volatility instead of the flat one suggested by it's less sophisticated counterpart.

Several empirical studies have shown already that the constant-volatility-assumption contradicts market realities (see e.g. Cont (2001), Guillaume et al. (1997)) The most salient drawback of the B&S-model is often considered to be it's inability to replicate the long tails which are observable in daily stock-returns. These however can be captured quite well by Heston's approach. (see Silva and Yakovenko (2003), Daniel (2003))

The model's dynamics are characterized by the following three equations

$$dS_t = \mu S_t dt + \sqrt{v_t} S_t dW_t^{(1)} \tag{1}$$

$$dv_t = -\gamma(v_t - \theta)dt + \kappa\sqrt{v_t} dW_t^{(2)} \tag{2}$$

$$dW_t^{(2)} = \rho dW_t^{(1)} + \sqrt{1-\rho^2} dZ_t \tag{3}$$

Here Z_t is a Wiener process independent of $W_t^{(1)}$.Defining r_t by $r_t = \ln(S_t/S_0)$, applying Ito's Formula and setting $x_t = r_t - \mu t$ one arrives at

$$dx_t = -0.5 v_t dt + \sqrt{v_t} dW_t^{(1)} \tag{4}$$

A transition density $P_t(x, v|v_i)$ for the joint realization of x_t and v at time t given an initial log-return $x = 0$ and variance v_i at $t = 0$ was constructed by Dragulescu and Yakovenko (2002). However, because the variance is not a directly observable market quantity $P_t(x, v|v_i)$ is not suitable for the calibration of the Heston Model to empirical data.

Fortunately Dragulescu and Yakovenko (2002) also introduce reduced densities by integrating out the variance.

$$P_t(x|v_0) = \int_0^{+\infty} P_t(x, v|v_0)dv \tag{5}$$

This is the density of having log-return x at time t given a variance v_0 at time $t = 0$. If we wanted to specify a starting variance the function above might be the correct choice. However deciding upon such a value can be quite arbitrary for variance can not be directly observed. To circumvent this uncertainty Dragulescu and Yakovenko (2002) use the stationary density of (2) which is given by

$$\Pi_*(v) = \frac{\alpha^\alpha}{\Gamma(\alpha)} \frac{v^{\alpha-1}}{\theta^\alpha} e^{-\alpha v/\theta}, \quad \alpha = \frac{2\gamma\theta}{\kappa^2} \tag{6}$$

Inserting (6) into (5) and integrating over v_0 they eventually arrive at the following result:

$$P_t(x) = \frac{1}{2\pi} \int_{-\infty}^{+\infty} e^{\mathrm{i}p_x + F_t(p_x)} dp_x \tag{7}$$

with

$$F(t, p_x) = \frac{\gamma\theta}{\kappa^2}\Gamma t - \frac{2\gamma\theta}{\kappa^2} \ln\left[\cosh\frac{\Omega t}{2} + \frac{\Omega^2 - \Gamma^2 + 2\gamma\Gamma}{2\gamma\Omega} \sinh\frac{\Omega t}{2}\right] \tag{8}$$

$$\Gamma = \gamma + \mathrm{i}\rho\kappa p_x \tag{9}$$

$$\Omega = \sqrt{\Gamma^2 + \kappa^2(p_x^2 - \mathrm{i}p_x)} \tag{10}$$

2 Empirical density function and fitting

Let a series of log-normal stock returns be given by $\boldsymbol{x}(\Delta t) = \{x_1, \ldots, x_n\}$. Here Δt denotes the time-step. Thus in the case of 21 trading days per month and $\Delta t = 1/252$, $\boldsymbol{x}(1/252)$ will be a series of daily returns. To make the data compatible with process (4) we also shift every return by $\mu\Delta t = \frac{\Delta t}{n}\sum_{i=1}^n x_i$ and thus reset

$$\boldsymbol{x}(\Delta t) = \{x_1 - \mu\Delta t, \ldots, x_n - \mu\Delta t\} \tag{11}$$

Furthermore we set $x_{min} = \min \boldsymbol{x}(\Delta t)$, $x_{max} = \max \boldsymbol{x}(\Delta t)$ and introduce the bin-size Δx. The number of bins is then given by $M = \mathrm{ceil}\left(\frac{x_{max} - x_{min}}{\Delta x}\right)$.

We then simply determine the relative probabilities p_i for each bin by weighting the number of returns in that bin by the total number of observations.

For a bin with boundaries $[a, b)$ we also introduce the bin representative $\bar{x}_i = a + \frac{1}{2}(b - a)$. Using the M representatives $\{\bar{x}_1 \ldots, \bar{x}_M\}$ we can now define the function

$$p_{emp}(x) = \sum_{i=1}^{M} p_i \delta_{\bar{x}_i}(x) \tag{12}$$

After the initial empirical function has been constructed one can get rid of outliers by determining the α and $1 - \alpha$ quantiles. Using these values to reset x_{min} and x_{max} one then proceeds to derive a new empirical density (without the outliers).

To underline that the shape of the analytical density depends on the choice of $(\gamma, \theta, \kappa, \rho)$, we set $P_t(x) = p(t, x, \gamma, \theta, \kappa, \rho)$ The distance function between the empirical distribution and the analytical one is then given by

$$f(\gamma, \theta, \kappa, \rho) = \sum_{1=1}^{M} \left[p_{emp}(\bar{x}) - p(\Delta t, \bar{x}, \gamma, \theta, \kappa, \rho))\right]^2 \tag{13}$$

To conclude the fitting procedure we have to minimize f with respect to $(\gamma, \theta, \kappa, \rho)$. Note that f as defined in (13) is just one choice of a distance function. Alternatives might be the the root-mean-squared error or the absolute distance.

3 The genetic algorithm approach

3.1 Motivation

In this section we assume our cost function to be $f : \mathbb{R}^n \to \mathbb{R}$ and our goal will be to minimize f. The function's input is thus a vector of the form $\vec{x} = (x_1, \ldots, x_n) \in \mathbb{R}^n$ and an input-solution-pair is a tuple $(\vec{x}, f(\vec{x})) \in \mathbb{R}^{n+1}$. Furthermore we restrict the search-space of our algorithm by limiting it to the hypercube

$$H(\vec{a}, \vec{b}) = \{\vec{x} \in \mathbb{R}^n, a_i \leq x_i \leq b_i, \forall \in \{1, \ldots n\}\} \tag{14}$$

However, if f is set to equal (13) it's calculation must be carried out via numerical integration and the integral might not be well-behaved for every choice of $(\gamma, \theta, \kappa, \rho)$. This can lead to the crash of most conventional optimization routines.

In the case of the Heston Model f is known explicitly. However, for many stochastic processes the transition density itself is often not known and has to be estimated numerically. Pedersen (1995) and Brandt and Santa-Clara (2001) for example suggest the use of Monte Carlo coupled with a Maximum-Likelihood estimator in oder to construct the transition density. In such a case approaches like Levenberg-Marquard or steepest descent (see Kelley (1999)) would be computationally expensive, for f' will have to be determined numerically. It would be best to use an optimization routine which:

- does not depend on the form of f,
- doesn't necessitate the computation of f',
- works reliably for larger n,
- is able to disregard local optima
- and is inherently parallel which will allow to exploit modern multi-core processors.

All of the criteria above are met by genetic algorithms (GA). For an introduction to GA see e.g. Sivanandam and Deepa (2007) or Gen (1997). Despite being considered heuristics they have been repeatedly shown to perform well with complex problems (see e.g. De Jong (1975) , Marco-Blaszka and Desideri (1999)). In the next section we are going to introduce a simple GA containing all the necessary buildings blocks which are inherent to this type of optimization procedure.

3.2 The Algorithm

The general structure of most genetic algorithms is quite simple. One first creates an initial population by randomly sampling from the search space. Afterwards the population members are selected and exchange information in order to produce new, fitter solutions. The population is then sorted according to the fitness level and the worst solutions are killed off. The members of this new population are again selected for "mating" and the entire procedure is repeated.

To embed our optimization problem into the GA-framework we must introduce some definitions first. From here on $\vec{x}$ will be called a chromosome and the tuple $(\vec{x}, f(\vec{x})) \in \mathbb{R}^{n+1}$ a member of the population $\tilde{P}$ which is simply a set of such tuples. To make the notation more compact we will refer to a population member via $\tilde{m}$, it's chromosome via $\tilde{m}(1)$ and it's cost via $\tilde{m}(2)$. In this context $\tilde{P}_i$ will be the i-th population member. Here we also access the chromosome via $\tilde{P}_i(1) \in \mathbb{R}^n$ and the cost via $\tilde{P}_i(2) \in \mathbb{R}$.

To initialize a GA one first has to generate an initial population by randomly sampling N_{ini} points from the space $H(\vec{a}, \vec{b})$ (see Algorithm 1)

input : The dimension of the objective function's input n , the size of the initial population N_{ini}, an empty container $\tilde{P}_{ini}$ and the vectors $\vec{a}, \vec{b} \in \mathbb{R}^n$
output: The container $\tilde{P}_{ini}$ filled with N_{ini} elements
local : $\vec{x} \in \mathbb{R}^n$

for $i = 1$ **to** N_{ini} **do**
  for $i = 1$ **to** n **do**
    $x_i =$ `drawUniformRandom` $[a_i, b_i]$;
  end
  Add $(\vec{x}, f(\vec{x}))$ to $\tilde{P}_{ini}$;
end

Algorithm 1: Creating an initial population

The next step consists of deciding which members of the population should mate. Here the best solution should also have the best chances of passing on it's information. To achieve that we use a cost-weighted-selection approach. Thus the lower the cost of a given population member the higher it's probability to be selected for mating. Assuming that the current population $\tilde{P}$ is already sorted so that $\tilde{P}_1(2) \leq \tilde{P}_i(2) \quad \forall \tilde{P}_i \in \tilde{P}$ the selection probabilities are given by

$$p_i = \left| \frac{P_i(2) - P_N(2)}{\sum_{j=1}^{N} (P_j(2) - P_N(2))} \right| \quad \forall i \in \{1, \ldots, n\} \tag{15}$$

Using this probabilities one than constructs a discrete probability distribution with $P(X = i) = p_i$ and uses it to draw members from $\tilde{P}$. The number of mating-pair selections is regulated by the variable $\eta \in [0, 1]$ specifying the fraction of the population that will mate (see Algorithm 2) Note that algorithm 2 allows self-pairing. Thus a member can be paired with itself.

input : The current population $\tilde{P}$ and it's size N and $\eta \in \mathbb{R}$
output: A list of mating-pairs $\tilde{M}$
local : $\vec{p} \in \mathbb{R}^N$, $\tilde{m}_1 \in \mathbb{R}^{n+1}$, $\tilde{m}_2 \in \mathbb{R}^{n+1}$

$\tilde{P}$=`sortPopulation` $(\tilde{P})$; `// so that` $\tilde{P}_1(2) \leq \tilde{P}_i(2) \quad \forall \tilde{P}_i \in \tilde{P}$

for $i = 1$ **to** N **do**
  $p_i = \left| \frac{P_i(2) - P_N(2)}{\sum_{j=1}^{N}(P_j(2) - P_N(2))} \right|$
end

for $i = 1$ **to** $\lfloor 0.5N\eta \rfloor$ **do**
  $\tilde{m}_1$ =`drawMemberFromPopulation` $(\vec{p})$;
  $\tilde{m}_2$ =`drawMemberFromPopulation` $(\vec{p})$;
  Add $(\tilde{m}_1, \tilde{m}_2)$ to $\tilde{M}$;
end

Algorithm 2: Selecting members for mating

Now that we have selected the mating pairs, we proceed to introduce another crucial genetic operator - the crossover. The crossover constitutes the step where the exchange of information between the different solutions present in the population takes place. The key idea is to combine data of two members (to "mate" them) in order to produce two fitter members. How this is accomplished varies depending on the concrete problem. The approach used here is described in Algorithm 3. Using the mating pairs obtained via Algorithm 2 one goes on to combine the genetic data of the two members to produce two new ones. The new members inherit most of their parent's data but one randomly selected gene. We collect this results in a container $\tilde{P}_{ch}$ and set $\tilde{P} = \tilde{P} \cup \tilde{P}_{ch}$. To avoid an exponential population growth we introduce the variable N_{max}, sort $\tilde{P}$ so that $\tilde{P}_1(2) \leq \tilde{P}_i(2) \quad \forall \tilde{P}_i \in \tilde{P}$ and remove all $\tilde{P}_i$ with $i > N_{max}$. This way the population size is kept constant to reduce computational overhead.

Like many optimization routines the GA-approach might also suffer from premature convergence. Thus it finds one local minimum and converges to it neglecting the better optimal solutions within the search-space. In the context of the GA this is often caused by the **genetic-drift**. This happens when the population $\tilde{P}$ contains one member $\tilde{m}_{best}$ which is significantly fitter than the others. $\tilde{m}_{best}$ will thus be frequently selected for mating and will end up dominating the "gene-pool". After several iterations the entire population might end up converging to $\tilde{m}_{best}$. The usual approach to remedy this situation is to introduce a steady flow of new and unbiased genetic information to the population. This can be achieved by randomly sampling points from the search space and adding them to $\tilde{P}$. Thus one runs Algorithm 1 to produce a specified number of new members N_{mut} and adds them to the current population. This

is also often referred to as introducing **mutation**.

Here it is important to note that this new members will be added to $\tilde{P}$ after it has been truncated to N_{max}. This way they will have a chance to mate in the next run.

input : A set of mating pairs $\tilde{M}$, their number N_M, factor $\beta \in [0,1]$, the current population $\tilde{P}$ and N_{max}
output: An updated population $\tilde{P}$
local : $\tilde{m}_1 \in \mathbb{R}^{n+1}$, $\tilde{m}_2 \in \mathbb{R}^{n+1}$, $a \in \mathbb{N}$, $\vec{x}, \vec{y} \in \mathbb{R}^n$, $z_1, z_2 \in \mathbb{R}$, $\vec{z}_1, \vec{z}_2 \in \mathbb{R}^n$, N_{new}

for $i = 1$ **to** N_M **do**
  $(\tilde{m}_1, \tilde{m}_2) = \tilde{M}_i$;
  a=`drawInteger` $(\{1, \ldots, n\})$;
  `// Uniform random draw of an integer from` $\{1, \ldots, n\}$
  $\vec{x} = \tilde{m}_1(1) = \{x_1, \ldots, x_a, \ldots x_n\}$;
  $\vec{y} = \tilde{m}_2(1) = \{y_1, \ldots, y_a, \ldots y_n\}$;
  $z_1 = x_a - \beta(x_a - y_a)$;
  $z_2 = y_a + \beta(x_a - y_a)$;
  $\vec{z}_1 = \{x_1, \ldots, x_{a-1}, z_1, x_{a+1} \ldots x_n\}$;
  $\vec{z}_2 = \{y_1, \ldots, y_{a-1}, z_1, y_{a+1} \ldots y_n\}$;
  $\tilde{m}_1 = (\vec{z}_1, f(\vec{z}_1))$ $\tilde{m}_2 = (\vec{z}_2, f(\vec{z}_2))$
  Add $\tilde{m}_1$ and $\tilde{m}_2$ to $\tilde{P}_{ch}$;
end
$\tilde{P} = \tilde{P} \cup \tilde{P}_{ch}$;
$\tilde{P}$=`sortPopulation` $(\tilde{P})$; `// so that` $\tilde{P}_1(2) \leq \tilde{P}_i(2) \quad \forall \tilde{P}_i \in \tilde{P}$
N_{new}=`getPopSize` $(\tilde{P})$;
$\tilde{P} = \tilde{P} \setminus \{\tilde{P}_{N_{max}+1}, \ldots, \tilde{P}_{N_{new}}\}$;

Algorithm 3: Mating/crossover and regulating population size

Finally we have to define a **stopping-criteria** for the algorithm. The most straight-forward approach and the one we use here, is to stop when the fittest population member $\tilde{P}_1$ hasn't changed K-times in a row.

input : The cost function $f : \mathbb{R}^n \to \mathbb{R}$, $n \in \mathbb{N}$, ;
$\vec{a}, \vec{b} \in \mathbb{R}^n$;
$N_{ini}, N_{max}, N_{mut}, K \in \mathbb{N}$;
$\eta, \beta \in [0, 1]$,
output: $\tilde{m}_{best} \in \mathbb{R}^{n+1}$
local : $\tilde{P}, \tilde{M}, \tilde{P}_{mut}$, $k \in \mathbb{N}$, $\tilde{m}^{old}_{best}, \tilde{m}^{new}_{best}$

$\tilde{P}$=`createPopulation` (N_{ini});
while $k < K$ **do**
  $\tilde{m}^{old}_{best}$=$\tilde{P}_1$;
  $\tilde{M}$=`selectForMating` (η);
  $\tilde{P}$=`mateAndTruncate` $(\tilde{M}, N_{max})$;
  $\tilde{P}_{mut}$=`createPopulation` (N_{mut});
  $\tilde{P}$=$\tilde{P} \cup \tilde{P}_{mut}$;
  $\tilde{P}$=`sortPopulation` $(\tilde{P})$; `// so that` $\tilde{P}_1(2) \leq \tilde{P}_i(2) \quad \forall \tilde{P}_i \in \tilde{P}$
  $\tilde{m}^{new}_{best}$=$\tilde{P}_1$;
  `// Check if the best member has changed`
  if $\tilde{m}^{new}_{best}(2) > \tilde{m}^{old}_{best}(2)$ **then**
    $k = 0$
  else
    k=k+1
  end
end

Algorithm 4: The Genetic Algorithm

4 Numerical Results

To arrive at the numerical results presented in this section the following inputs have been used:

Empirical data: Daily returns of the DAX for the time period 26.11.1990-15.08.2014

Genetic Algorithm parameters: $n = 4$, $N_{ini} = 200$, $N_{max} = 100$, $N_{mut} = 50$, $\rho = 0.5$, $\beta = 0.5$, $K = 10$

First we run a calibration on an empirical density created for $\Delta x = 0.001$. The resulting analytical density exhibits a very good fit to the empirical data and captures both the curtosis and the tails (see figure 1)

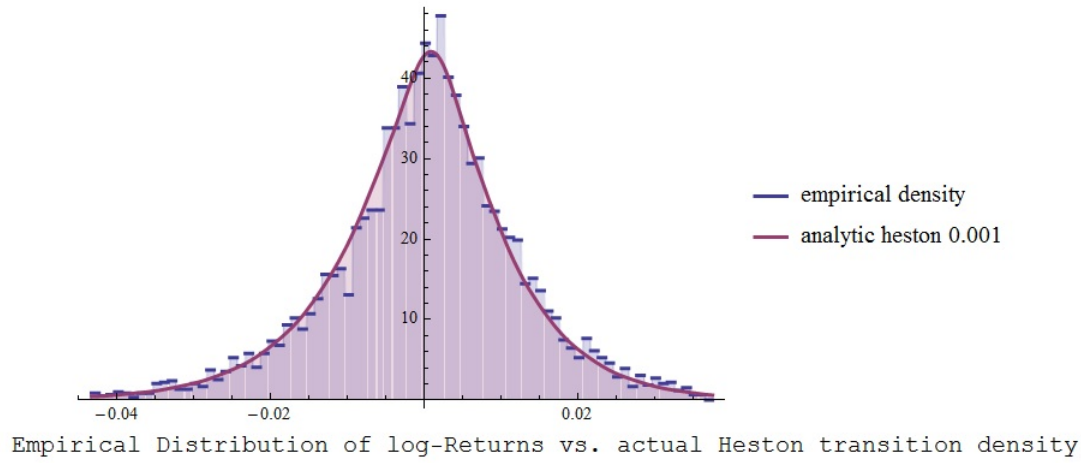

Figure 1: Heston analytical density plot for $\Delta t = 1/252, \gamma = 4.75, \theta = 0.04, \kappa = 0.5, \rho = -0.78$. Empirical density of DAX' daily returns with bin size 0.001

It is important to note that the evaluation speed of f as defined in (13) strongly depends on the number of bins M, for it equals the number of times we have to calculate the complex analytical Heston-Density $P_t(x)$. Thus while calibrating the model we ask ourselves whether the bin-size Δx has any significant effect on the parameters output by the algorithm.

For f as in equation (13) and for bin-sizes $\Delta x = 0.001, \Delta x = 0.0025, \Delta x = 0.005$ the algorithm outputs the following set of parameters:

Δx	γ	θ	κ	ρ
0,001	4.75	0.04	0.5	-0.78
0,0025	4.41	0.043	0.52	-1.0
0,005	5.64	0.041	0.55	-1.0

Table 1: DAX calibration results for different bin-sizes

Table 1 shows that θ, κ are relatively stable, whereas γ, ρ seem to have a non-negligible dependence on the bin-size Δx. However, if we plot the transition density (7) for this three sets of parameters (see figure 2), we see that the overall shape of the distribution is almost identical. The calibration quality thus does not seem to depend on the bin-size Δx.

This is a very helpful result, for it allows us to decrease the computation time of f making the entire algorithm significantly more efficient as shown by table 2. In our implementation we did not use parallelization, which would decrease the run-time even more.

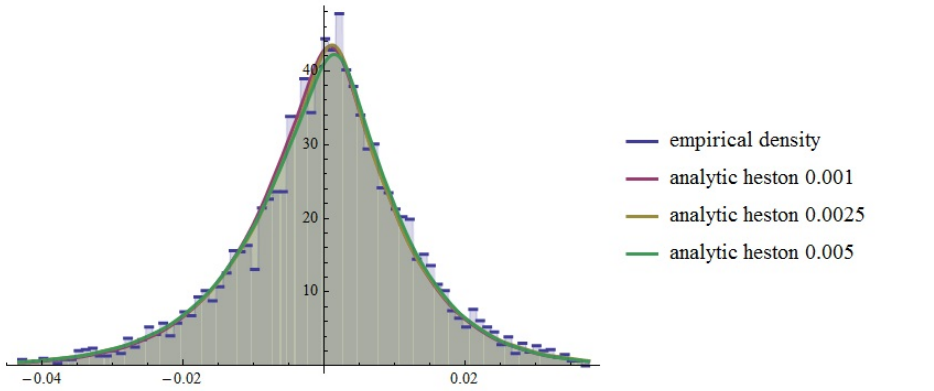

Figure 2: Heston analytical density plots with parameter sets as shown in table 1 for the DAX. Empirical density for $\Delta x = 0,001$

Δx	0.001	0.0025	0.005
Run-time	970 sec	560 sec	175 sec

Table 2: Run-time of the algorithm for different bin sizes

Besides the shape of $P_t(x)$ the density $\Pi_*(v)$ (see equation (6)) is also of interest. When running a Monte Carlo simulation one would sample from $\Pi_*(v)$ to get an initial variance for each path. Thus it is interesting to see whether the parameters shown in table 1 produce different shapes of $\Pi_*(v)$.

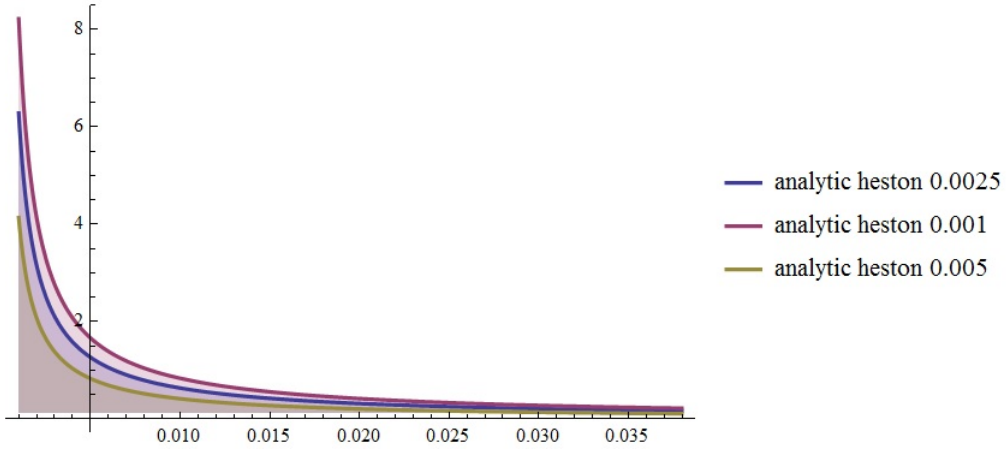

Figure 3: $\Pi(v)$ for the different parameter sets as shown in table 1 for the DAX.

Figure 3 shows that the three sets produce similar but still different densities $\Pi_*(v)$. Whether this effect can be ignored depends on the problem at hand. In case of derivatives pricing this might not be the case but could be tolerated in a risk-management context. However, the computational speed advantage of using a sparser empirical density derived using $\Delta x = 0.005$ bins is significant and should be factored in as well.

The performance of our optimization routine could be improved even further by reducing the number of parameters. In their empirical tests Dragulescu and Yakovenko (2002) disregard the correlation ρ by fixing it at zero. We tested whether the quality of the fit is affected by this approach. The calibration results for a set-up with $\rho = 0$ are presented in table 3. The newly calibrated parameters differ from the previous ones. However, figure 4 shows that the fitting quality is only marginally affected.

Δx	γ	θ	κ	ρ
0,001	2.70	0.038	0.35	0.00
0,0025	2.64	0.044	0.405	0.00
0,005	4.19	0.041	0.46	0.00

Table 3: DAX calibration results for different bin-sizes with $\rho = 0$

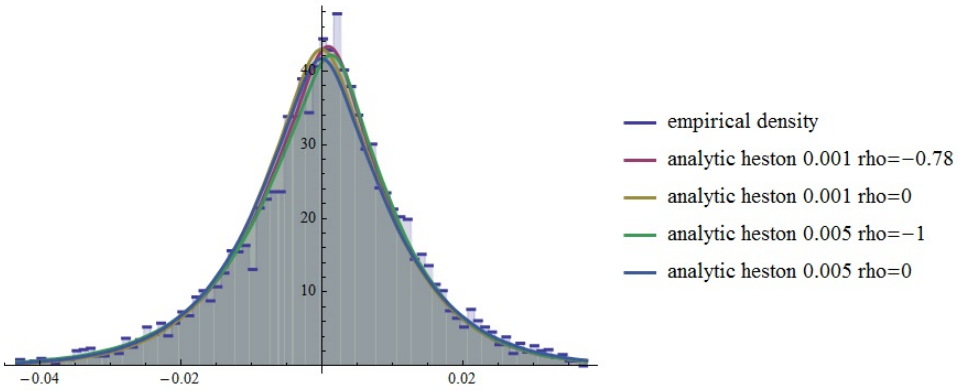

Figure 4: Heston analytical density plots for the DAX. Empirical density for $\Delta x = 0,001$. Comparison of calibration results "with and without ρ"

To see whether the results would also hold for a different time series we also run the algorithm (with $\rho = 0$) on the daily returns of the S&P 500 index (10.10.1990 - 10.10.2014). The results are illustrated in Figure 5 and table 4. Once again the bin size only has a marginal effect on the fitting result. Like we have already observed for the DAX, θ is again stable for different choices of Δx.

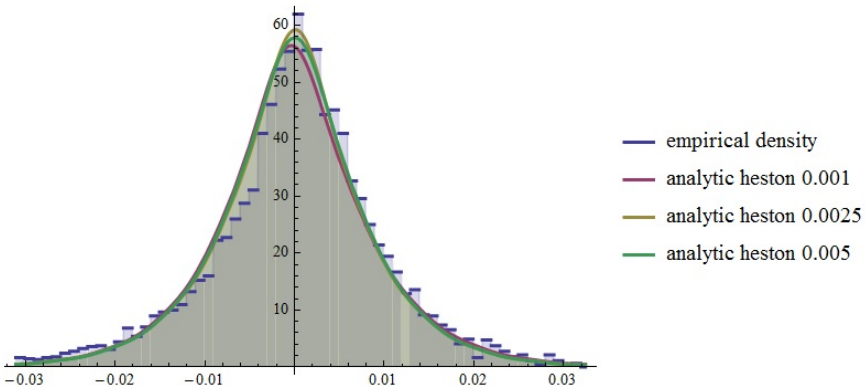

Figure 5: Heston analytical density plots with parameter sets as shown in table 4 for S&P 500. Empirical density for $\Delta x = 0,001$

Δx	γ	θ	κ	ρ
0,001	3.04	0.022	0.28	0.00
0,0025	2.85	0.020	0.26	0.00
0,005	5.43	0.021	0.30	0.00

Table 4: Calibration results for different bin-sizes for the S& P 500

5 Conclusion

The goal of this paper was to calibrate Heston's stochastic volatility model to an empirical density. After introducing the process and the density derived by Dragulescu and Yakovenko (2002) in section 1 we proceeded to shown how to construct an empirical density function from a given set of daily returns.

Seeing how the evaluation of $P_t(x)$ necessitates numerical integration, we looked for a stable optimization routine which can be used on numerical cost functions. Our solution was to apply a simple genetic algorithm which we described in detail in section 3. Using this algorithm we fitted the reduced transition density (7) to an empirical-density constructed using the daily-return data of the DAX. As shown in figure 1 we were able to achieve a very good fit in terms of both kurtosis and tails. The fitting quality was also only marginally affected if ρ was fixed at 0 which noticeably improved the computation time and stability of the optimization routine. We then applied our approach to a different time series (S&P 500) and were able to replicate the good fitting results obtained for the DAX (cf. figure 5)

To improve the computational efficiency an analysis of the impact of the empirical density's granularity on the calibration result has been carried out in section 4. Interestingly the bin-size Δx does not seem to affect the shape of the resulting density (cf. figure 2). However, the different parameterizations do affect the shape of $\Pi(v)$. Still, whether this deviations have any significant repercussions will depend on the problem at hand and must be analyzed in the respective context. The computational advantage of using a larger Δx is bolstered by table 2.

Despite our main focus being the calibration of Heston's volatility model, we have also shown that genetic algorithms can be applied in this context. Further research will have to show whether this result also holds for more complex numerical and empirical densities.

References

Michael W. Brandt and Pedro Santa-Clara. Simulated likelihood estimation of diffusions with an application to exchange rate dynamics in incomplete markets. NBER Technical Working Paper 0274, National Bureau of Economic Research, Inc, 2001. URL `https://ideas.repec.org/p/nbr/nberte/0274.html`.

Rama Cont. Empirical properties of asset returns: stylized facts and statistical issues. *Quantitative Finance*, 1:223–236, 2001.

Gilles Daniel. Goodness-of-fit of the heston model. *arXiv:cs/0305055*, May 2003. URL `http://arxiv.org/abs/cs/0305055`.

Kenneth Alan De Jong. *An Analysis of the Behavior of a Class of Genetic Adaptive Systems.* PhD thesis, University of Michigan, Ann Arbor, MI, USA, 1975. AAI7609381.

Adrian A. Dragulescu and Victor M. Yakovenko. Probability distribution of returns in the heston model with stochastic volatility. *arXiv:cond-mat/0203046*, March 2002. URL `http://arxiv.org/abs/cond-mat/0203046`. Quantitative Finance 2, 443 (2002).

Mitsuo Gen. *Genetic Algorithms and Engineering Design.* John Wiley & Sons, January 1997. ISBN 9780471127413.

Dominique M. Guillaume, Michel M. Dacorogna, Rakhal R. Dav, Ulrich A. Mller, Richard B. Olsen, and Olivier V. Pictet. From the bird's eye to the microscope: A survey of new stylized facts of the intra-daily foreign exchange markets. *Finance and Stochastics*, 1(2):95–129, April 1997. ISSN 0949-2984. doi: 10.1007/s007800050018. URL `http://link.springer.com/article/10.1007/s007800050018`.

S. L. Heston. A closed-form solution for options with stochastic volatility with applications to bond and currency options. *Review of Financial Studies*, 6(2): 327–343, January 1993. ISSN 0893-9454, 1465-7368. doi: 10.1093/rfs/6.2.327. URL `http://rfs.oxfordjournals.org/content/6/2/327`.

C. T. Kelley. *Iterative Methods for Optimization.* SIAM, January 1999. ISBN 9780898714333.

Nathalie Marco-Blaszka and Jean-Antoine Desideri. Numerical solution of optimization test-cases by genetic algorithms. February 1999. URL `http://hal.inria.fr/inria-00073055`.

Morten Bjerregaard Pedersen. A new approach to maximum likelihood estimation for stochastic differential equations based on discrete observations. *Scandinavian Journal of Statistics*, 22:55–71, 1995.

A. Christian Silva and Victor M. Yakovenko. Comparison between the probability distribution of returns in the heston model and empirical data for stock indexes. *Physica A: Statistical Mechanics and its Applications*, 324(1): 303–310, 2003. ISSN 0378-4371.

S. N. Sivanandam and S. N. Deepa. *Introduction to Genetic Algorithms*. Springer Science & Business Media, October 2007. ISBN 9783540731900.

Impressum

Diese Veröffentlichung erscheint im Rahmen der Online-Publikationsreihe „Forschung am IVW Köln“. Alle Veröffentlichungen dieser Reihe können unter www.ivw-koeln.de oder hier abgerufen werden.

Forschung am IVW Köln, 3/2015
Dolgov: Calibration of Heston's stochastic volatility model to an empirical density using a genetic algorithm
Köln, Februar 2015

ISSN (online) 2192-8479

Kontakt Autor / Contact author:

Urij Dolgov
Institut für Versicherungswesen /
Institute for Insurance Studies

Fakultät für Wirtschafts- und Rechtswissenschaften /
Faculty of Business, Economics and Law

Fachhochschule Köln / Cologne University of Applied Sciences
Gustav Heinemann-Ufer 54
50968 Köln

Tel. +49 221 8275-3271
Fax +49 221 8275-3277

Mail Urij.Dolgov@outlook.de

Zuletzt erschienen im Rahmen von „Forschung am IVW Köln“

2015

- Heep-Altiner, Berg: Mikroökonomisches Produktionsmodell für Versicherungen, Nr. 2/2015
- Institut für Versicherungswesen: Forschungsbericht für das Jahr 2014, Nr. 1/2015

2014

- Müller-Peters, Völler (beide Hrsg.): Innovation in der Versicherungswirtschaft, Nr. 10/2014
- Knobloch: Zahlungsströme mit zinsunabhängigem Barwert, Nr. 9/2014
- Heep-Altiner, Münchow, Scuzzarello: Ausgleichsrechnungen mit Gauß Markow Modellen am Beispiel eines fiktiven Stornobestandes, Nr. 8/2014
- Grundhöfer, Röttger, Scherer: Wozu noch Papier? Einstellungen von Studierenden zu E-Books, Nr. 7/2014
- Heep-Altiner, Berg (beide Hrsg.): Katastrophenmodellierung - Naturkatastrophen, Man Made Risiken, Epidemien und mehr. Proceedings zum 6. FaRis & DAV Symposium am 13.06.2014 in Köln, Nr. 6/2014
- Goecke (Hrsg.): Modell und Wirklichkeit. Proceedings zum 5. FaRis & DAV Symposium am 6. Dezember 2013 in Köln, Nr. 5/2014
- Heep-Altiner, Hoos, Krahforst: Fair Value Bewertung von zedierten Reserven, Nr. 4/2014
- Heep-Altiner, Hoos: Vereinfachter Nat Cat Modellierungsansatz zur Rückversicherungsoptimierung, Nr. 3/2014
- Zimmermann: Frauen im Versicherungsvertrieb. Was sagen die Privatkunden dazu?, Nr. 2/2014
- Institut für Versicherungswesen: Forschungsbericht für das Jahr 2013, Nr. 1/2014

2013

- Heep-Altiner: Verlustabsorbierung durch latente Steuern nach Solvency II in der Schadenversicherung, Nr. 11/2013
- Müller-Peters: Kundenverhalten im Umbruch? Neue Informations- und Abschlusswege in der Kfz-Versicherung, Nr. 10/2013
- Knobloch: Risikomanagement in der betrieblichen Altersversorgung. Proceedings zum 4. FaRis & DAV-Symposium am 14. Juni 2013, Nr. 9/2013
- Strobel (Hrsg.): Rechnungsgrundlagen und Prämien in der Personen- und Schadenversicherung - Aktuelle Ansätze, Möglichkeiten und Grenzen. Proceedings zum 3. FaRis & DAV Symposium am 7. Dezember 2012, Nr. 8/2013
- Goecke: Sparprozesse mit kollektivem Risikoausgleich - Backtesting, Nr. 7/2013
- Knobloch: Konstruktion einer unterjährlichen Markov-Kette aus einer jährlichen Markov-Kette, Nr. 6/2013
- Heep-Altiner et al. (Hrsg.): Value-Based-Management in Non-Life Insurance, Nr. 5/2013
- Heep-Altiner: Vereinfachtes Formelwerk für den MCEV ohne Renewals in der Schadenversicherung, Nr. 4/2013
- Müller-Peters: Der vernetzte Autofahrer – Akzeptanz und Akzeptanzgrenzen von eCall, Werkstattvernetzung und Mehrwertdiensten im Automobilbereich, Nr. 3/2013
- Maier, Schimikowski: Proceedings zum 6. Diskussionsforum Versicherungsrecht am 25. September 2012 an der FH Köln, Nr. 2/2013
- Institut für Versicherungswesen: Forschungsbericht für das Jahr 2012, Nr. 1/2013