Proceedings of 2012 4th International Conference on Machine Learning and Computing
IPCSIT vol. 25 (2012) © (2012) IACSIT Press, Singapore

Real Coded Genetic Algorithm Particle Filter for Improved Performance

Muhammad Shakir Hussain [1]
[1] University College London, Dept. of Computer Science, London, United Kingdom

Abstract. Particle filters are an important class of online posterior density function estimation algorithms. In this paper we propose a real coded genetic algorithm particle filter (RGAPF) by adding a real coded genetic algorithm layer in the particle filter. The real coded genetic algorithm layer improves the prediction accuracy of the particle filter, and provides accurate results even when we decrease the number of particles. We test the performance of the RGAPF on the stochastic volatility estimation problem.

Keywords: Particle filters, Real coded genetic algorithms, Stochastic volatility estimation.

1. Introduction

In mathematical finance, the problem of stochastic volatility estimation arises while pricing derivative securities. The volatility of the underlying asset is hidden, but it can be estimated based on its price process. The volatility is given by an auto regressive model, with a mean reverting behavior. Our aim is to correctly estimate the volatility of the underlying asset, and estimate the parameters of the mathematical model relating the price process with the stochastic volatility.

Various particle filters have been proposed in literature for the stochastic volatility estimation problem, however in [9] Liu & West reasoned that along with volatility estimation, parameters of the model should also be estimated online. For this purposed they proposed a particle filter in [9], that used an annealing mutation operator. In [15], Raggi et al added a markov chain Monte Carlo (MCMC) step to this particle filter. Their algorithm, called Particle learning algorithm (PLA) will be used as a bench mark in our current work for performance comparison.

The similarities between particle filters and evolutionary algorithms have been known and exploited by many researchers [11, 12, 13]. Patrigo et al [13] recently suggested a framework for combining particle filters with population based metaheuristics for visual articulated motion tracking. Uosaki et al in [16] proposed an evolutionary strategies particle filter (ESP) for the fault detection in dynamic system. Patrigo used a metaheuristic to improve performance for the high dimensional visual articulated motion tracking problem, while Uosaki et al were more focused on the dual estimation problem.

The main contribution of this paper is the application of real coded genetic algorithm particle filters to the stochastic volatility estimation problem. For real coded genetic algorithms we experiment with 4 different recombination operators. The performance of the RGAPF is then compared with a sequential importance resampling particle filter (SIR), the parameter learning algorithm (PLA) given by Raggi et al [15] and SIR with a Covariance Matrix Adaptation (Evolution Strategies) layer.

2. Problem Formulation

To define the filtering problem, consider a dynamic system represented by the state sequence $\{x_k, k \in N\}$ where the temporal evolution is provided by the state equation:

$$x_k = f(x_{k-1}, v_{k-1}), \quad (1)$$

where *f* is a nonlinear function and **N** is the set of natural numbers, $\{v_k, k \in \boldsymbol{N}\}$ is the process noise sequence. The state process is hidden, but we are provided with online measurements of the observation process that is given by the observation equation:

$$z_k = h(x_k, n_k) \tag{2}$$

where *h* is a non-linear function of x_k. The objective is to recursively estimate x_k whenever we obtain a new measurement of z_k [3, 5, 7].

3. The Stochastic Volatility estimation problem

The Heston model to approximate the stochastic volatility is given by the following stochastic differential equation:

$$dX(t)/X(t) = \sqrt{V(t)}\,dW_x(t), \tag{3}$$

$$dV(t) = k\left(\theta - V(t)\right)dt + \varepsilon\sqrt{V(t)}\,dW_v(t), \tag{4}$$

Where k, , θ are strictly positive constants, and where W_x and W_v are scalar Brownian motions in some probability measure; we assume that $dW_x(t).dW_v(t) = \rho dt$. Where the correlation ρ is some constant in [-1, 1]. X(t) represents an asset price process.

The Euler discretization of the above stochastic differential equation is given by:

$$\ln\hat{X}(t+\Delta) = \ln\hat{X}(t) - \frac{1}{2}\hat{V}(t)^{+}\Delta + \sqrt{\hat{V}(t)^{+}}Z_x\sqrt{\Delta} \tag{5}$$

$$\hat{V}(t+\Delta) = \hat{V}(t) + k\left(\theta - \hat{V}(t)^{+}\right)\Delta + \varepsilon\sqrt{\hat{V}(t)^{+}}Z_v\sqrt{\Delta} \tag{6}$$

where $\hat{X}(t)$, is the observed price process, and $\hat{V}(t)$, is the volatility process that has to be estimated.

For a dual estimation process, the parameters k, , θ will also be estimated on line along with the volatility estimation [2].

4. Particle Filters

Particle filtering algorithms are an important class of online posterior density function estimation algorithms for non-linear, non-Gaussian state space models. We will compare the performance of our algorithm with particle filtering algorithms found in literature, the SIR particle filter, the particle learning algorithm (PLA) given in [13] and SIR with ES-CMA layer. For the algorithm of ES-CMA layer, see [14].

4.1. SIR Algorithm

Many different types of particle filters have been proposed in literature. We will focus on the Sequential Importance Resampling (SIR) particle filter. An SIR algorithm as presented in [6] is:

Choose a proposal distribution $q(x_{k+1}^i|x_k^i, y_{k+1})$,, resampling strategy and the number of particles N.

Initialization: Generate $x_1^i \sim p_{x_0}, i = 1, \dots, N$ and let initial weights to be 1/N.

For loop *k* = 1, 2…End of Observations

Start

1. Measurement Update: For i = 1,2,…, N

$$w_{k|k}^i = \frac{1}{c_k} w_{k|k-1}^i p(y_k|x_k^i) \tag{7}$$

Where the normalization weight is given by:

$$c_k = \sum_{i=1}^N w_{k|k-1}^i p(y_k|x_k^i) \tag{8}$$

2. Estimation:

The filtering density is approximated by

$$p(x_{1:k}|y_{1:k}) = \sum_{i=1}^N w_{k|k}^i \,\delta(x_{1:k+1} - x_{1:k+1}^i) \tag{9}$$

And the mean is approximated by:

$$\dot{x} \approx \sum_{i=1}^N w_{k|k}^i \, x_{1:k}^i \tag{10}$$

```
    IF (Sample depletion occurs, call a pre-defined resampling function)
        Reset the weights and copies of each particle.

3.  Time Update:
    Generate predictions according to the proposal distribution
        $x_{k+1}^{i} \sim q(x_{k+1} | x_k^i, y_{k+1})$                                (11)
    And compensate for the importance weights
        $w_{k+1|k}^{i} = w_{k|k}^{i} \frac{p(x_{k+1}^i | x_k^i)}{q(x_{k+1}^i | x_k^i, y_{k+1})}$      (12)
4.  IF ( $y_k$ is not last observation)
    $k = k + 1$
    Go to step 1.

     Else End For loop.
END
```

4.2. The Parameter Learning Algorithm

This particular algorithm was given by Raggi et al [15], and was specifically designed for the estimation of stochastic volatility and parameters of the model. The particle learning algorithm resembles the algorithm given by Liu & West [9] with an additional MCMC step after 50 iterations.

```
$[\{\{x_k^i, w_k^i\}\}_{i=1}^{N_s}] = PLA[\{x_{k-1}^i, w_{k-1}^i\}_{i=1}^{N_s}, z_k]$
 For $i = 1 : N_s$
    -   Calculate $\mu_k^i$
    -   Calculate $w_k^i = q(i | z_{1:k}) \propto p(z_k | \mu_k^i) w_{k-1}^i$          (13)

END For
Calculate total weight: $t = SUM(\{w_k^i\}_{i=1}^{N_s})$
For $i = 1 : N_s$
    -   Normalize: $w_k^i = t^{-1} w_k^i$                                          (14)

END For
    Resample by assigning equal weight to each particle
For $j = 1 : N_s$ .
    -   Draw  $x_k^i \sim q(x_k | i^j, z_{1:k}) = p(x_k | x_{k-1}^{ij})$              (15)
    -   Assign weights $w_k^j \propto p(z_k | x_k^j) / q(z_k | \mu_k^{ij})$          (16)

END For
Calculate total weight: $t = SUM[\{w_k^i\}_{i=1}^{N_s}]$
For $i = 1 : N_s$
    -   Normalize: $w_k^i = t^{-1} w_k^i$                                          (17)

END For
```

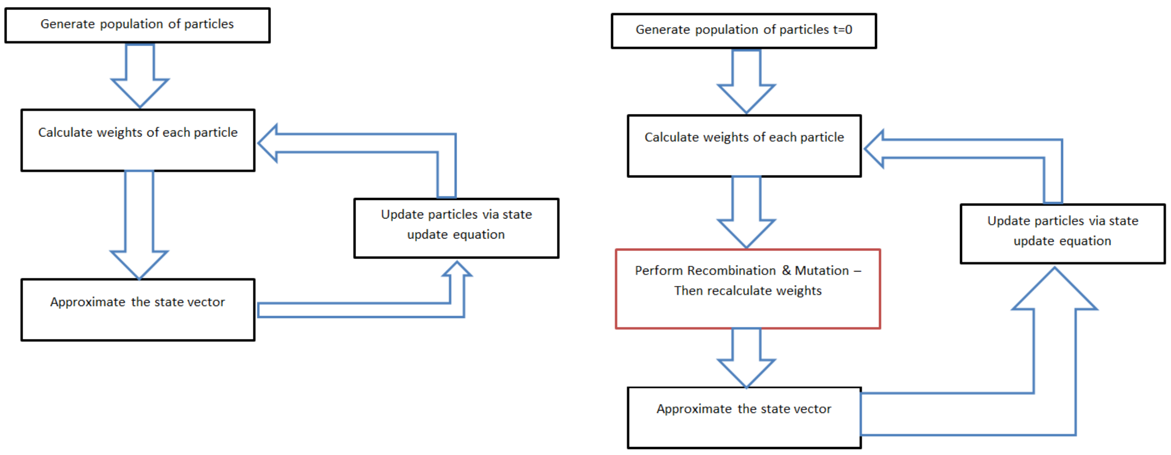
(Optional: After 50 iterations) Move the former particles according to MCMC with invariant distribution the posterior and update the sufficient statistics according to the MCMC move [15].

5. Our Proposed Real Coded Genetic Algorithm Particle Filter

Genetic algorithms are stochastic population based metaheuristics [1, 8].We add a real recombination and mutation layer in a generic SIR particle filter. The convergence theorem for real coded genetic algorithms was given by Goldberg in [4]. The flow chart of our algorithm compared to a generic SIR is shown below:

Fig. 1. Flow chart of a generic particle filter (left) and a RGAPF (right) with an additional recombination and mutation layer.

The real recombination operators we use in our experiments are:


Generate population of particles
Calculate weights of each particle
Update particles via state update equation
Approximate the state vector
Generate population of particles t=0
Calculate weights of each particle
Perform Recombination & Mutation – Then recalculate weights
Update particles via state update equation
Approximate the state vector

1. Arithmetic recombination
2. Simulated Binary Crossover (SBX)
3. Uni-modal normal distribution crossover (UNDX)
4. Parent centric crossover (mPCX)

The mutation operator is Gaussian mutation. For more details of these operators see [14].

6. Design of Experiment and Results

We simulate the values of the volatility and price process by using equations (5-6), setting the values of parameters as:

$$k = \theta = \varepsilon = 0.9,$$
$$\Delta = 0.1$$

Now we have the price process (observation equation) and the volatility process (state equation). Given the observation process, we will now estimate the values of the hidden volatility by using the particle filters that we proposed, and compare their performance with an SIR particle filter and the parameter learning algorithm (PLA). We will use 2000 and 200 particles for our estimation.

We then calculate the root mean square error (RMSE) of the actual volatility compared to the estimated volatility for that filter. This procedure is carried out 50 times for each algorithm. The average time taken by the algorithm for the each run is also calculated. We are using an acer aspire 5742 system with 4 GB DDR3 memory and Intel Core i5-450M processor.

The recombination and mutation probabilities are:

$$p_c = 0.45$$
$$p_m = 0.01$$

The parameter values for the recombination and mutation operator are as follows:

1. For Arithmetic Recombination, choose α = 0.7
2. For Simulated Binary Crossover, choose n = 2.
3. Set   = 0.5 in the Gaussian mutation operator.

For 2000 and 200 particles, the results for the 50 runs are shown in the tables below:

Table 1. Performance Comparison for 2000 particles

PF Name	**RMSE**	**Average Time**
SIR PF	0.2578	94.716 seconds
PLA	0.1453	153.219 seconds
PF with ES-CMA	0.0645	165.740 seconds
RGAPF with Arithmetic recombination	0.0651	114.605 seconds
RGAPF with simulated binary cross over (SBX)	0.0945	125.785 seconds
RGAPF with UNDX recombination	0.16456	147.654 seconds
RGAPF with mPCX recombination	0.07456	143.429 seconds

Table 2. Performance Comparison for 200 particles

PF Name	**RMSE**	**Average Time**
SIR PF	0.6578	9.176 seconds
PLA	0.2453	15.329 seconds
PF with ES-CMA	0.0845	16.074 seconds
RGAPF with Arithmetic recombination	0.0951	11.462 seconds
RGAPF with simulated binary cross over (SBX)	0.1945	12.578 seconds
RGAPF with UNDX recombination	0.1605	13.165 seconds
RGAPF with mPCX recombination	0.0965	13.342 Seconds

7. Conclusion

We proposed RGAPF for the estimation of the stochastic volatility of stocks. The performance of our proposed algorithms is compared with a generic SIR particle filter, and the particle filter proposed by Raggi et al[15] and an SIR with ES-CMA layer. The RGAPF performs relatively better than these three algorithms and its performance does not deteriorate when the number of particles is decreased. For future work, the dimension of the state process should be increased to see what affect this might have on the performance.

8. References

[1] D.E., Goldberg, (1989*) Genetic Algorithms in Search, Optimization and Machine Learning*. Addison Wesley, ISBN 0-201-15767-5.

[2] Eric Jacquier, Nicholas G. Polson and Peter E. Rossi; *Bayesian Analysis of Stochastic Volatility Models.* Journal of Business and Economic Statistics, 1994, 12(4), pp. 371-89

[3] Fearnhead, P. *Sequential Monte Carlo methods in filter theory*. Ph.D. dissertation, University of Oxford, UK, 1998.

[4] Goldberg, D.E. *Real Coded Genetic Algorithms, Virtual Alphabets and Blocking.* Complex Systems 5(1991) 139-167.

[5] Gordon, N., Salmond, D., and Smith, A. *A novel approach to nonlinear/non-Gaussian Bayesian state estimation*. In lEE Proceedings on Radar and Signal Processing, vol.140, 1993, 107-113.

[6] Gustafsson, F. *Particle filter theory and practice with positioning applications*. IEEE A&E Systems Magazine vol.25, No.7 July 2010

[7] Hammersley, J., and Morton, K.*Poor man's Monte Carlo.* Journal of the Royal Statistical Society, Series B, 16(1954), 23.

[8] J.H. Holland, (1998*) Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to*

biology, control and artificial intelligence. MIT Press, ISBN 0-262-58111-6.

[9] Liu, J.and West, M. *Combined parameter and state estimation in simulation-based filtering*. Sequential Monte Carlo in Practice, Springer July 2001, 197-217.

[10] Morrison, R. *Designing evolutionary algorithms for dynamic environments*. Springer 2004.

[11] Pantrigo, J., Duarte, A., Sanchez, A and Cabido, R. *Scatter Search Filter to solve Dynamic Travelling Salesman Problem.*

[12] Patrigo, J., Sanchez, A., Gianikellis, K. and Montemayor, A. *Combining Particle Filter and Population-based Metaheuristics for Visual Articulated Motion Tracking*. Electronic Letters on Computer Vision and Image Analysis 5(3):68-83, 2005

[13] Patrigo, J., Sanchez, A., Gianikellis, K. and Montemayor*, A. Heuristic particle filter: applying abstraction techniques to the design of visual tracking algorithms.* Expert Systems. Wiley. Vol. 28, Issue 1, Pages 49-69, Feburary 2011

[14] Talbi, E. Metaheuristics: From Design to Implmentation. (Wiley series on Parallel and Distributed Computing). Wiley-Blackwell July 2009. 208-213.

[15] Raggi, D., and Bordignon. S, *Sequential Monte Carlo Methods for Stochastic Volatility Models with Jumps*, 2006.

[16] Uosaki, K., Kimura, Y. and Hatanaka, T. *Nonlinear State Estimation by Evolutionary Strategies Based Particle Filters*. 0-7803-7804-0 /03/17.00 0 2003 IEEE