15th Triennial World Congress, Barcelona, Spain

# ARMA MODEL SELECTION USING PARTICLE SWARM OPTIMIZATION AND AIC CRITERIA

**Mark S. Voss[a] and Xin Feng[b]**

[a] *Department of Civil and Environmental Engineering*
*voss@rocketmail.com*
[b] *Department of Electrical and Computer Engineering*
*fengx@marquette.edu*

*Marquette University, Milwaukee Wisconsin*

Abstract: This paper presents a new method for determining ARMA model parameters using Particle Swarm Optimization (PSO). PSO is a new optimization method that is based on a social-psychological metaphor. Each ARMA model is represented as a particle in the particle swarm. Particles in a swarm move in discrete steps based on their current velocity, memory of where they found their personal best fitness value, and a desire to move toward where the best fitness value that was found so far by all of the particles during a previous iteration. PSO is applied for determining the ARMA parameters for the Wolfer Sunspot Data. The method is extended using Akaike's Information Criterion (AIC). PSO is used to simultaneously optimize and select an estimated "best approximating ARMA model" based on AIC. Several plots are included to illustrate how the method converges for various PSO parameter settings.

Keywords: ARMA Models, ARMA Parameter Estimation, Identification Algorithms, Agents , Genetic Algorithms, Model Approximation. Particle Swarm Optimization.

## 1. INTRODUCTION

System Identification (Ljung, 1999; Pandit, 1983; Söderström, 1989) is concerned with the derivation of mathematical models from experimental data. When given a data set one typically applies a set of candidate models and chooses one of the models based on a set of rules by which the models can be assessed. One of the simplest System Identification models is the Autoregressive Moving Average (ARMA) model as shown in equation (1).

$$\begin{aligned} & y_t + \phi_1 y_{t-1} + \phi_2 y_{t-2} + \cdots + \phi_n y_{t-n} \\ & = \theta_0 + \theta_1 a_{t-1} + \theta_2 a_{t-2} + \cdots + \theta_m a_{t-m} \end{aligned} \quad (1)$$

where $\phi_i$, $i = 1,n$ and $\theta_j$, $j = 1,m$ are the parameters for an ARMA(n,m) model. For a given ARMA(n,m) model the model parameters $\phi_i$ and $\theta_j$ are selected such that equation (2) is minimized,

$$\sum_{t=1}^{N} \left( y_{t\_data} - y_{t\_ARMA} \right)^2 \quad (2)$$

where,

$$N = \text{ number of data points.} \quad (3)$$

## 2. PARTICLE SWARM OPTIMIZATION (PSO)

The Particle Swarm Algorithm is an adaptive algorithm based on a social-psychological metaphor (Kennedy and Eberhart, 2001a). A population of individuals adapt by returning stochastically towards previously successful regions in the search space, and are influenced by the successes of their topological neighbors. Most particle swarms are based on two sociometric principles. Particles fly through the solution space and are influenced by both the best particle in the particle population and the best solution that a current particle has discovered so far. The best particle in the population is typically denoted by (global best), while the best position that has been visited by the current particle is denoted by (local best). The (global best) individual conceptually connects all members of the population to one another. That is, each particle is influenced by the very best performance of any member in the entire population. The (local best) individual is conceptually seen as the ability for particles to remember past personal successes.

Particle Swarm Optimization is a relatively new addition to the evolutionary computation methodology, but the performance of PSO has been shown to be competitive with more mature methodologies (Eberhart and Shi, 1998a; Kennedy and Spears, 1998b). Since it is relatively straightforward to extend PSO by attaching mechanisms employed by other evolutionary computation methods that increase their performance; PSO has the potential to become an excellent framework for building custom high-performance stochastic optimizers (Løvbjerg, *et al*., 2001). It is interesting to note that PSO can be considered as a form of continuous valued Cellular Automata. This allows its hybridizations to extend into areas other than computational intelligence (Kennedy and Eberhart, 2001a).

### *2.1 PSO Equations*

The ith particle is represented as,

$$X_I = (x_{i1}, x_{i2}, ..., x_{iD}) \quad (4)$$

where D is the dimensionality of the problem. The rate of the position change (velocity) of the i[th] particle is represented by,

$$V_I = (v_{i1}, v_{i2}, ..., v_{iD}) \quad (5)$$

where $v_{ik}$ is the velocity for dimension "k" for particle "i". The best previous position (the position giving the best fitness value) of the i[th] particle is represented as,

$$P_I = (p_{i1}, p_{i2}, ..., p_{iD}) \quad (6)$$

The best previous position so far achieved by any of the particles (the position giving the best fitness value) of the i[th] particle is recorded and represented as,

$$P_G = (p_{g1}, p_{g2}, ..., p_{gD}) \quad (7)$$

On each iteration the velocity for each dimension of each particle is updated by,

$$v_{ik} = w_k v_{ik} + c_1 \varphi_1 p_{ik} + c_2 \varphi_2 p_{gk} \quad \{k, g\} \in \{1, 2, ..., D\} \quad (8)$$

where $w_k$ is the inertia weight that typically ranges from 0.9 to 1.2. $c_1$ and $c_2$ are constant values typically in the range of 2 to 4. These constants are multiplied by $\varphi$ (a uniform random number between 0 and 1) and a measure of how far the particle is from its personal best and the global best particle so far. From a social point of view, the particle moves based on its current direction ( $w_k$ ), its memory of where it found its personal best ( $p_{ik}$ ), and a desire to be like the best particle in the population ( $p_{gk}$ ).

### *2.2 PSO - Position Update Rule*

After a new velocity for each particle is calculated, each particle's position is updated according to:

$$x_{ik} = x_{ik} + v_{ik} \quad (9)$$

It typically takes a particle swarm a few hundred to a few thousand updates for convergence depending on the parameter selections within the PSO algorithm (Eberhart and Shi, 1998b).

### *2.3 Particle Swarm Parameter Settings*

Particle Swarm Optimization was used to determine the ARMA parameters for the Wolfer Sunspot Data (1770-1869). The PSO parameter values that were used are given in Tables 1. and 2.

Table 1.
Particle Swarm
Parameter Settings

| Parameter | Setting |
|---|---|
| Population Size | 80 |
| Number of Iterations | 1000 |
| $c_1$ and $c_2$ | 2.0 |
| Inertial Weight Figs. 1., 2. and 3. | 0.7 |
| Inertial Weight Figs. 4. | 0.7 to 0.9 |

Table 2.
Particle Variable Settings

| Dim | Variables | $x_{min}$ | $x_{max}$ | $\lvert v_{max} \rvert$ |
|---|---|---|---|---|
| 1 | n | 1 | 10 | 3 |
| 2 | m | 0 | 10 | 3 |
| 3 | $\theta_0$ | -25 | 25 | 5 |
| 4 | $\phi_1$ | -25 | 25 | 5 |
| 5 | $\phi_2$ | -25 | 25 | 5 |
| … | … | … | … | … |
| 13 | $\phi_{10}$ | -25 | 25 | 5 |
| 14 | $\theta_1$ | -25 | 25 | 5 |
| 15 | $\theta_2$ | -25 | 25 | 5 |
| … | … | … | … | … |
| 23 | $\theta_{10}$ | -25 | 25 | 5 |

Table 2. illustrates data structure for a particular particle. Note that a each variable has a minimum and maximum "x" value along with a maximum absolute velocity. Having separate limits for each particle dimension is an extension to the simple PSO. Dimension 1 is used for the regressive order which has been set to be between 1 and 10 inclusive. Similarly Dimension 2 is used for the order of the moving average order. That is each particle represents an ARMA(n,m) model where n and m can assume values 1 through 10 and 0 through 10 respectively. The values that the ARMA parameters $\phi$ and $\theta$ can assume have also been limited to ±25 with a maximum absolute velocity of 5. It is apparent that having expert knowledge about specific variable domains when setting up the particle variable settings can increase the performance of the algorithm. Setting the PSO particle variable settings is analogous to the procedure for setting up variable representations within a binary genetic algorithm; where one needs to choose the interval and a string length for each variable (Angenline, 1998).

### *2.4 Results: PSO + ARMA = SwARMA*

Initially "n" and "m" were fixed to test the ability of PSO to determine the ARMA parameters for a *fixed ARMA model*. The results from the study are shown in Fig. 1. This model was chosen to demonstrate the use of PSO on a well understood System Identification problem. The authors where somewhat surprised at the results. The PSO method converged in less that a minute on a 200 Mhz Pentium PC. In all cases the solution found was substantially better that those found using the Wolfer Sunspot example provided with the IMSL Libraries. It is recognized that the IMSL Libraries may not be "optimal" and are used here to provide a benchmark value.

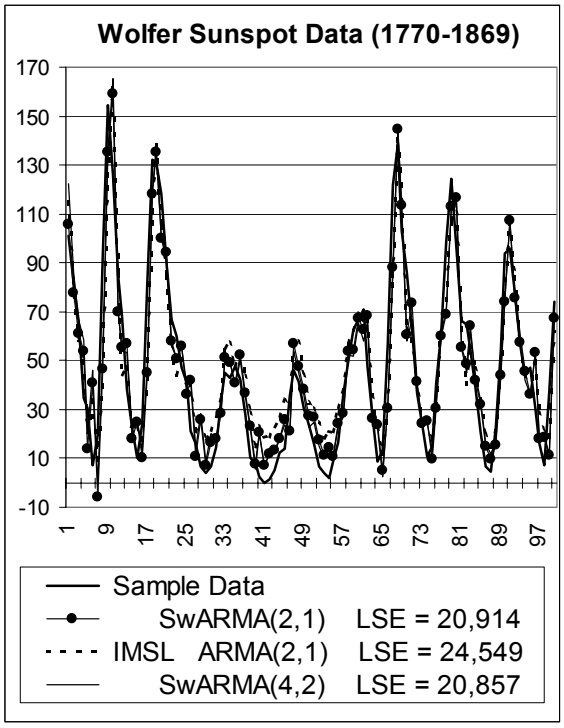


Fig. 1. Wolfer Sunspot Data.
Fixed SwARMA models. Weight = 0.7.

For an ARMA(2,1) model the PSO solution was 15% better than the IMSL solution. In light of these results the authors chose to call the combination of PSO with ARMA: SwARMA (Voss and Feng, 2001). These results suggest some interesting future research with regards to traditional System Identification.

### *2.5 Results: Static SwARMA convergence*

The convergence of the static ARMA(2,1) and ARMA(4,2) models, using the particle variable settings given in Table 2., are shown in Fig. 2. The dependant variable is the sum squared error SSE. This is also the fitness of the best particle in the population. The independent variable is show is the number of fitness function evaluations. This can be calculated as the number of particle iterations times the population size. Plotting the number of function evaluations is more informative than plotting the iterations. The fixed ARMA(2,1) model converges faster that the fixed ARMA(4,2) model as expected. Most of the fitness improvement is achieved after only 1000 function evaluations. The plateau after 1000 iterations suggests that the algorithm might have a more optimal parameter configuration.

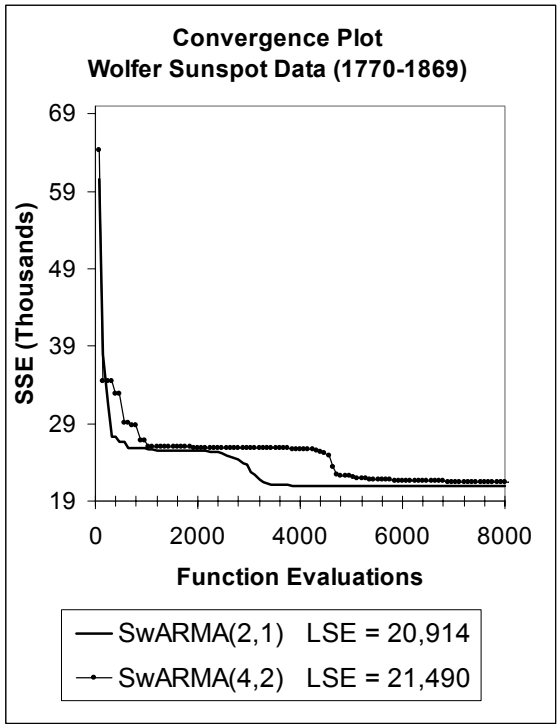


Fig. 2. Convergence plot for fixed SwARMA models. Weight = 0.7.

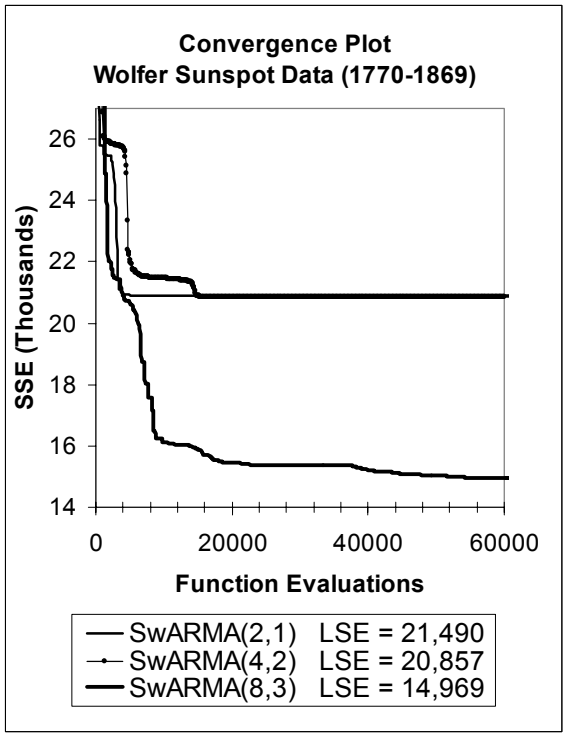


Fig 3. Convergence plot for variable SwARMA model. Weight = 0.7.

*2.6 Results: Dynamic SwARMA convergence*

Fig. 3. illustrates the effect of allowing dimensions "1" and "2" (which correspond to n and m in an ARMA(n,m) model) participate in the Particle Swarm Algorithm.

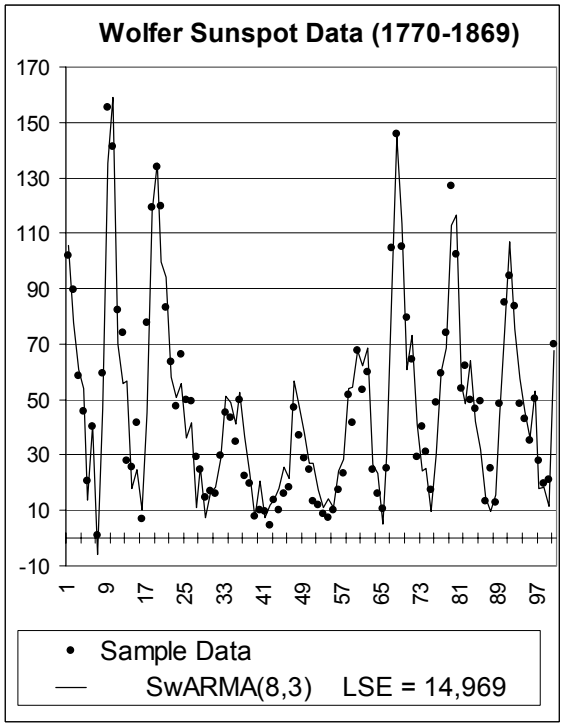


Fig. 4. Wolfer Sunspot Data. Variable SwARMA model. Weight = 0.7.

The best model after 60,000 function evaluations (750 iterations) was an ARMA(8,3) model. This model was plotted against the fixed ARMA models discussed in Fig. 2.. The constant improvement is caused by the model simultaneously optimizing, while at the same time, increasing the number of ARMA parameters. The observation of this behaviour is necessary in order to extend the SwARMA model to incorporate Akaike's Information Criterion (AIC) (Burnham and Anderson, 1998 ). into the model selection process. AIC will be discussed in the next section. The ARMA (8,3) model is plotted in Fig. 4. This model is somewhat large ( 8+3 = 11 parameters ) for the 100 data points used. The next section discusses selecting models using a constraint that takes both model fit and model complexity into consideration.

## 3. AKAIKE'S INFORMATION CRITERION (AIC)

Akaike's Information Criterion is an information-theoretic approach for selecting the estimated best approximating ARMA model.

AIC can be expressed as,

$$AIC = n\log(\hat{\sigma}^2) + 2K \tag{10}$$

where,

$$\hat{\sigma}^2 = \frac{\sum_{t=1}^{N}\left(y_{t\_data} - y_{t\_ARMA}\right)^2}{N}, \tag{11}$$

where,

$K = n + m$, sum of the ARMA(m,n) model parameters. (12)

### *3.1 AIC- A Heuristic Interpretation*

AIC is sometimes interpreted as the sum of two terms; the first is a measure of model fit, while the second is a penalty for the number of model parameters. This is an acceptable heuristic interpretation as long as one does not forget that the second term is not an arbitrary penalty term, but rather it is derived based on a link between information theory and the Kullback-Liebler (K-L) distance (Burnham, and Anderson, 1998).

### *3.2 Results: Dynamic ARMA convergence*

AIC was easily integrated into the SwARMA methodology. This was accomplished by simply replacing the particle fitness function (equation 2) with equation 10. It was demonstrated in Fig. 3. that the variable SwARMA model became more complex with increasing iterations. Fig. 5 demonstrates the ability for AIC to restrict model complexity. Fig. 5 also demonstrates the advantageous use of dynamic inertial weights. For the variable SwARMA(2,1) model the particles inertial weight was linearly decreased from 0.9 to 0.7 during the run. The effect of this was to delay convergence during the beginning of the run. That is, using a higher initial inertial weight prevented the swarm from prematurely converging on a sub-optimal solution. This can be observed as the shift to the right exhibited by the SwARMA(2,1) run. Fig. 5. illustrated the importance of understanding the behavior of particles (Kennedy, 1998c) based on the parameter settings.

## 4. CONCLUSION

Particle Swarm Optimization (PSO) was demonstrated as a viable method for determining the ARMA model parameters for a fixed size ARMA model. The PSO methodology was extended to solve variable ARMA models based on a sum squared error criteria (SSE). The variable ARMA model demonstrated the ability to use PSO to dynamically select the model size while decreasing the (SSE). Akaike's Information Criterion (AIC) was then integrated into the variable SwARMA model to allow for ARMA model selection based on an information-theoretic approach (SwARMA-AIC). Convergence plots for SwARMA-AIC demonstrated the ability for this hybrid algorithm to use AIC to select an estimated best approximating ARMA model. Particle Swarm Optimization has been shown to be a potential tool for optimising ARMA models.

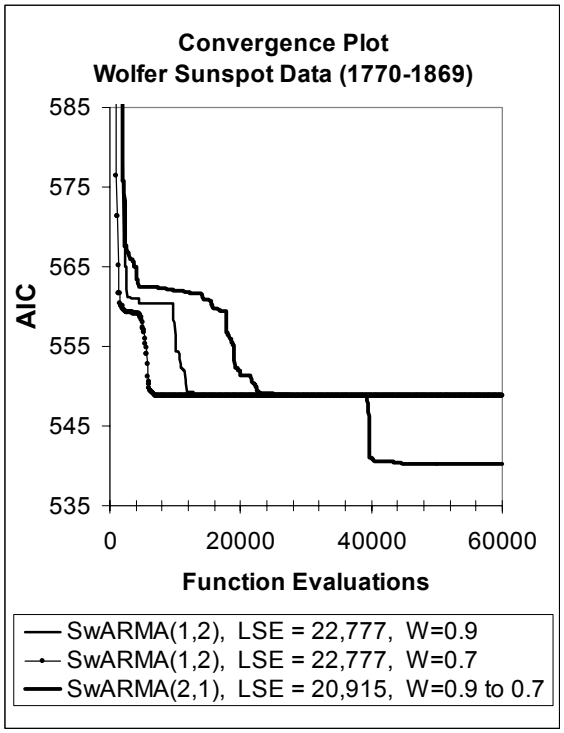


Fig. 5. Convergence plot for variable SwARMA-AIC models.

## REFERENCES


Angenline, P.J., (1998). Evolutionary Optimization Versus Particle Swarm Optimization: Philosophy and Performance Differences, *Evolutionary Programming VII*, Porto, V.W., Saravanan, N. Waagen, D., Eiben, A.E., 1447, 611-618, Springer.

Burnham, K.P., Anderson, D.R., (1998). *Model Selection and Inference*, Springer-Verlag, New York, NY.

Eberhart, R.C., Shi, Y., (1998a). Comparison between Genetic Algorithms and Particle Swarm Optimization, *Evolutionary Programming VII*, Porto, V.W., Saravanan, N. Waagen, D., Eiben, A.E., 1447, 611-618, Springer-Verlag.

Eberhart, R.C., Shi, Y., (1998b). Parameter Selection in Particle Swarm Optimization, *Evolutionary Programming VII*, Porto, V.W., Saravanan, N. Waagen, D., Eiben, A.E., 1447, 591-600, Springer-Verlag.

Kennedy, J., Eberhart, R.C., (Contributor), Shi, Y., (2001a). *Swarm Intelligence*, Morgan Kaufmann, San Francisco, CA.

Kennedy, J., and Spears, W.M., (1998b). Matching Algorithms to Problems: An Experimental Test of the Particle Swarm and Some Genetic Algorithms on the Multimodal Problem Generator", *Proceedings of the 1998 International Conference on Evolutionary Computation*, pp. 78-83.

Kennedy, J., (1998c). The behavior of particles, *Evolutionary Programming VII*, Porto, V.W., Saravanan, N. Waagen, D., Eiben, A.E., 1447, 581-590, Springer-Verlag.

Løvbjerg, M., Rasmussen, T.K., Krink, T., (1998). Hybrid Particle Swarm Optimizer with Breeding and Subpopulations, *Proceedings Genetic and Evolutionary Computation Conference*, GECCO-2001, Morgan Kaufmann, San Francisco, CA.

Ljung, L., (1999). *System Identification - Theory for the User*, Prentice Hall.

Pandit, S.M., Wu, S., (1983) *Time Series and System Analysis with Applications*, John Wiley & Sons.

Söderström, T., Stoica, P., (1989). *System Identification*, Prentice Hall.

Voss, M.S., Feng, X., (2001). Emergent System Identification using Particle Swarm Optimization, *Proceedings Complex Adaptive Structures Conference*, SPIE, Bellingham, WA.