

A PARAMETER ESTIMATION TECHNIQUE FOR THE NONLINEAR MUSKINGUM FLOOD ROUTING MODEL

Gurhan GURARSLAN¹, Halil KARAHAN²

- (1) Department of Civil Engineering, Pamukkale University, Denizli, Turkey, email: gurarslan@pau.edu.tr
(2) Department of Civil Engineering, Pamukkale University, Denizli, Turkey, email: hkarahan@pau.edu.tr

ABSTRACT

A parameter estimation technique based on particle swarm algorithm is proposed for the nonlinear Muskingum flood routing model. In the proposed technique, a penalty function approach is imposed on the numerical solution of the model in order to prevent negativity of outflows and storages. The performance of the proposed method is compared with reported techniques given in the literature through Wilson data. The results obtained for Wilson data demonstrate that the proposed algorithm can confidently be applied to estimate optimal parameter values of the nonlinear Muskingum flood routing model.

1 INTRODUCTION

Flood routing is important in the design of flood protection measures in order to estimate how the proposed measures will affect the behaviour of flood waves in rivers so that adequate protection and economic solutions can be found (Singh and McCann, 1980). Flood routing procedures may be classified as either hydrological or hydraulic. Hydrological methods use the principle of continuity and a relationship between discharge and the temporary storage of excess volumes of water during the flood period. Hydraulic methods of routing involve the numerical solutions of either the convective-diffusion equations or the one-dimensional Saint-Venant equations of gradually varied unsteady flow in open channels.

The hydraulic methods generally describe the flood wave profile more adequately when compared to hydrological methods, but practical application of hydraulic methods are restricted because of their high demand on computing technology, as well as on quantity and quality of input data (Singh, 1988). In practical applications, the hydrological routing methods are relatively simple to implement and reasonably accurate (Haktanir and Ozmen, 1997). An example of a simple hydrological flood routing technique used in natural channels is the Muskingum flood routing method (Karahan, 2009).

The Muskingum method was first developed by McCarthy (1938) for flood control studies in the Muskingum River basin in Ohio. The standard procedure for applying the Muskingum method involves two basic steps: calibration and prediction. In the calibration step, a parameter estimation problem is solved in which the parameter values for the Muskingum model of a river are determined by using historical inflow-outflow hydrograph data. The prediction step is the solution of a routing problem in which the outflow hydrograph for a given inflow hydrograph is determined by using the routing equations (Das, 2004).

The following hydrologic continuity and nonlinear storage equations are commonly used in the Muskingum model.

$$\frac{dS_t}{dt} = I_t - O_t \quad (1)$$

$$S_t = K [\chi I_t + (1 - \chi) O_t]^m \quad (2)$$

In the above equations S_t [L^3], I_t [L^3/T], and O_t [L^3/T] are simultaneous amounts of storage, inflow and outflow, respectively, at time t ; K [$L^{3(1-m)}T^m$] is storage-time constant and is greater than 0, and χ is a weighting factor usually varying between 0 and 0.5 (Tung, 1985); m is an exponent for considering the effects of nonlinearity and is greater than 1 for nonlinear models (the original linear model can be a special case of the nonlinear model where m is equal to 1). In the model, K , χ and m are unknown parameters and S_t and O_t must be handled as non-negative variables. Last decade, several researchers have proposed various heuristic algorithms such as genetic algorithm (Mohan, 1997), harmony search (Kim et al., 2001), particle swarm optimization (Chu and Chang, 2009) and immune clonal selection algorithm (Luo and Xie, 2010) to the parameter estimation of the nonlinear Muskingum model.

Recently, Chu and Chang (2009) proposed a parameter estimation technique for the nonlinear Muskingum model based on the particle swarm optimization (PSO). In their model, the authors claim that the proposed scheme can improve the accuracy of the Muskingum model for flood routing. However, Das (2009) discusses the optimal parameters obtained by the PSO are infeasible and give negative outflows. Das (2009) solved the hydrological continuity equation by the numerical scheme given by the authors for the parameters obtained by PSO to prove his claim.

In this paper, an efficient parameter estimation method for the nonlinear Muskingum model is proposed. In the proposed method, an indirect penalty function approach is used in the numerical model to prevent negativity of outflows and storages (or prevent infeasible solutions). The proposed algorithm finds the global or near-global minimum regardless of the initial parameter values with fast convergency and few control parameters. The performance of the method is compared with the reported techniques given in the literature through Wilson (1974) data. The results demonstrate that the proposed algorithm can confidently be applied to estimate optimal parameter values of the nonlinear Muskingum model.

2 ROUTING PROCEDURE OF THE NONLINEAR MUSKINGUM MODEL

Rearranging Eq. (2), the rate of outflow O_t can be obtained as (Tung, 1985; Geem, 2006),

$$O_t = \left(\frac{1}{1-\chi} \right) \left(\frac{S_t}{K} \right)^{1/m} - \left(\frac{\chi}{1-\chi} \right) I_t \quad (3)$$

Combining Eq. (3) and the continuity Eq. (1), the state equation can be obtained as,

$$\frac{dS_t}{dt} = - \left(\frac{1}{1-\chi} \right) \left(\frac{S_t}{K} \right)^{1/m} + \left(\frac{1}{1-\chi} \right) I_t \quad (4)$$

$$S_{t+1} = S_t + \Delta S_t \quad (5)$$

Note that the unit time step is used in the Eq. (5). If infeasible values of K , χ , m are selected, negative values of O_t and S_t can be obtained in the numerical solution of Muskingum model. Hence, an indirect penalty function approach is imposed to the numerical solution in order to prevent negativity (outflows and storages cannot be negative quantities) as follows:

$$S_{t+1}^* = \lambda_1 |S_{t+1}|, \text{ if } S_{t+1} < 0 \quad (6.a)$$

$$O_{t+1}^* = \lambda_2 |O_{t+1}|, \text{ if } O_{t+1} < 0 \quad (6.b)$$

where, $\lambda_{1,2}$ are penalty constants (used as 1000 for this study), S_{t+1}^* is penalized next storage, O_{t+1}^* is penalized next outflow. Note that, S_{t+1}^* and O_{t+1}^* are positive but unrealistic values.

The zero-order numerical routing procedure given by Tung (1985) and Geem (2006) is preferred for making a comparison. Main steps of this procedure are given as follows:

Step 1: Assume values for three parameters K , χ and m .

Step 2: Calculate S_t using Eq. (2), where initial outflow is the same as the initial inflow.

Step 3: Calculate the time rate of change of storage volume using Eq. (4).

Step 4: Estimate the next accumulated storage (S_{t+1}) using Eq. (5) or Eq. (6.a).

Step 5: Calculate next outflow (O_{t+1}) using Eq. (3) or Eq. (6.b).

Step 6: Repeat steps 2-5 for all times.

3 PARTICLE SWARM ALGORITHM

Particle swarm optimization (PSO) is population based stochastic optimization technique developed by Eberhart and Kennedy (1995), inspired by social behaviour of bird flocking or fish schooling. PSO shares many similarities with evolutionary computation techniques such as Genetic Algorithms (GA). The system initialized with a population of random solution and searches for optima by updating generations. However, unlike GA, PSO has no evolution operators such as crossover and mutation. Compared to GA, the advantages of PSO are that PSO is easy to implement and there are few parameters to adjust (PSO Tutorial, 2011). The book by Kennedy and Eberhart (2001) describes many philosophical aspects of PSO and swarm intelligence. An extensive survey of PSO applications is made by Poli (2007, 2008).

A basic variant of the PSO algorithm works by having a population (called a swarm) of candidate solutions (called particles). These particles are moved around in the search-space according to a few simple formulae. The movements of the particles are guided by their own best known position in the search-space as well as the entire swarm's best known position. When improved positions are being discovered these will then come to guide the movements of the swarm. The process is repeated and by doing so it is hoped, but not guaranteed, that a satisfactory solution will eventually be discovered.

Formally, let $f : \mathbb{R}^n \rightarrow \mathbb{R}$ be the fitness or cost function which must be minimized. The function takes a candidate solution as argument in the form of a vector of real numbers and produces a real number as output which indicates the fitness of the given candidate solution. The gradient of f is not known. The goal is to find a solution $\mathbf{a}$ for which $f(\mathbf{a}) \leq f(\mathbf{b})$ for all $\mathbf{b}$ in the search-space, which would mean $\mathbf{a}$ is the global minimum.

Let NP be the number of particles in the swarm, each having a position $\mathbf{x}_i \in \mathbb{R}^n$ in the search-space and a velocity $\mathbf{v}_i \in \mathbb{R}^n$. Let $\mathbf{lbest}_i$ be the best known position of particle i and let $\mathbf{gbest}$ be the best known position of the entire swarm (Wikipedia, 2011). A basic PSO algorithm is then:

- For each particle $i = 1, \dots, NP$ do:
 - Initialize the particle's position with a uniformly distributed random vector: $\mathbf{x}_i \leftarrow U(\mathbf{lb}, \mathbf{ub})$, where $\mathbf{lb}$ and $\mathbf{ub}$ are the lower and upper boundaries of the search-space
 - Initialize the particle's best known position to its initial position: $\mathbf{lbest}_i \leftarrow \mathbf{x}_i$
 - If $(f(\mathbf{lbest}_i) < f(\mathbf{gbest}))$ update the swarm's best known position: $\mathbf{gbest} \leftarrow \mathbf{lbest}_i$

- Initialize the particle's velocity: $\mathbf{v}_i \in U(-|\mathbf{ub} - \mathbf{lb}|, |\mathbf{ub} - \mathbf{lb}|)$
- Until a termination criterion is met (e.g. number of iterations performed, or adequate fitness reached), repeat:
 - For each particle $i = 1, \dots, NP$ do:
 - Pick random numbers: $r_1, r_2 \in U(0, 1)$
 - Update the particle's velocity:

$$\mathbf{v}_i \leftarrow w\mathbf{v}_i + c_1 r_1 (\mathbf{lbest}_i - \mathbf{x}_i) + c_2 r_2 (\mathbf{gbest} - \mathbf{x}_i)$$
 - Update the particle's position: $\mathbf{x}_i \leftarrow \mathbf{x}_i + \mathbf{v}_i$
 - If $(f(\mathbf{x}_i) < f(\mathbf{lbest}_i))$ do:
 - Update the particle's best known position: $\mathbf{lbest}_i \leftarrow \mathbf{x}_i$
 - If $(f(\mathbf{lbest}_i) < f(\mathbf{gbest}))$ update the swarm's best known position: $\mathbf{gbest} \leftarrow \mathbf{lbest}_i$
- Now $\mathbf{g}$ holds the best found solution.

Here, w is a factor of inertia ranging from 0.4 to 0.9 to control the impact that the history of velocities has on the current velocity (Shi and Eberhart, 1998; Yagiz and Karahan, 2011). w_1 and w_2 are the initial and final values of the inertia weight, respectively, k is the current iteration number and $k_{\max}$ is the maximum number of allowable iterations. c_1 and c_2 are cognitive and social coefficients, respectively. A relatively high value of the cognitive component results in excessive wandering of particles while a higher value of the social component causes premature convergence of particles (Kennedy and Eberhart, 1995). Hence, time-varying acceleration coefficients are employed to strike a proper balance between the cognitive and social component during the search. c_1 is changing from 2.5 to 0.5 and c_2 is changing from 0.5 to 2.5, over the full range of the search (Ratnaweera et al., 2004).

$$w = (w_1 - w_2) \frac{k_{\max} - k}{k_{\max}} + w_2 \quad (6)$$

$$c_1 = (c_{1f} - c_{1i}) \frac{k}{k_{\max}} + c_{1i} \quad (7)$$

$$c_2 = (c_{2f} - c_{2i}) \frac{k}{k_{\max}} + c_{2i} \quad (8)$$

4 APPLICATION TO WILSON (1974) DATA

In order to fairly compare the results, an example is solved through the developed model with the data given by Wilson (1974). The objective function to be minimized is the sum of the squared residuals (SSQ) between observed and calculated outflows as follows:

$$\text{minimize } SSQ = \sum_{t=1}^N \left[O_t - \hat{O}_t(K, \chi, m) \right]^2 \quad (9)$$

where, O_t denotes the observed outflow, $\hat{O}_t$ denotes the calculated outflow and N denotes the number of time step. The ranges of three parameters used in the applications are selected as $K = 0.01-1.0$, $\chi = 0.01-0.5$ and $m = 1.0-3.0$. The stopping condition can be defined as (Ali and Törn, 2004; Karahan, 2011):

$$|F_{\text{best}} - F_{\text{worst}}| \leq \varepsilon, \quad (12)$$

where, ε is a small number, selected as $1E-12$, F_{best} is the best fitness value in the population and F_{worst} is the worst fitness value in the population. Maximum iteration number was set to be 10,000. PSO algorithm was run 100 times to achieve average results. For each run, the initial population was randomly created by means of using different seed numbers.

The parameter estimation technique for the nonlinear Muskingum model using HS-BFGS is applied to data set given by Wilson (1974), which has been studied by Gill (1978), Tung (1985), Yoon and Padmanabhan (1993), Mohan (1997), Kim et al. (2001), Das (2004), Geem (2006), Chu and Chang (2009), Luo and Xie (2010), and Geem (2010).

In order to determine the suitable NP value, a sensitivity analysis was carried out thorough data set given by Wilson (1974). Statistical evaluation of the parameters, SSQ's, iteration number and CPU times for different NP value is presented in Table 1. As can be seen in Table 1, global optimum is never found for $NP=5$, is found 16 times for $NP=10$ and is found 98 times for $NP=20$ in the 100 runs. But, it is found 100 times for $NP=30$ and $NP=40$ in the 100 runs. The suitable NP value is selected as 30 for this problem in terms of requiring less CPU time and finding true global optimum.

As can clearly be seen in Table 1, there is extremely small difference between the best solution and the worst solution of 100 runs. Standard deviation of SSQ's of 100 runs is 3.23E-13. The best solution vector is obtained as a result of 500 iterations. The

proposed algorithm is free from infeasible starting vectors and computational divergence. Required CPU time of the proposed algorithm for 500 iterations is 3.30 second.

Table 1. Statistical evaluation of the parameters, SSQ's, iteration number and CPU times for different NP values

NP	Statistical Values	K	χ	m	SSQ	ITER	CPU (s)
5	Best	0.029898	0.113101	1.484230	36.768128	1007	1.14
	Worst	0.494480	0.297018	2.104218	370.743510	5220	5.87
	Mean	0.188958	0.273108	1.728968	71.376548	2086	2.35
	Std. Dev.	0.10	0.03	0.13	51.39	635	0.72
10	Best	0.079154	0.258792	1.864791	36.767888	590	1.40
	Worst	0.087638	0.291513	1.885623	47.822981	9082	20.24
	Mean	0.086414	0.287403	1.867704	36.966236	7588	16.91
	Std. Dev.	9.82E-04	3.73E-03	2.40E-03	1.20	1797	4.00
20	Best	0.086240	0.286886	1.858392	36.767888	524	2.29
	Worst	0.090389	0.300000	1.868109	39.327344	2445	10.84
	Mean	0.086291	0.287049	1.867989	36.793486	753	3.35
	Std. Dev.	4.14E-04	1.31E-03	9.69E-04	0.26	452	2.00
30	Best	0.086249	0.286917	1.868087	36.767888	500	3.30
	Worst	0.086249	0.286917	1.868087	36.767888	644	4.24
	Mean	0.086249	0.286917	1.868087	36.767888	558	3.71
	Std. Dev.	6.81E-09	1.42E-09	1.75E-08	3.23E-13	28	0.18
40	Best	0.086249	0.286917	1.868087	36.767888	510	4.56
	Worst	0.086249	0.286917	1.868087	36.767888	610	5.60
	Mean	0.086249	0.286917	1.868087	36.767888	554	4.95
	Std. Dev.	5.66E-09	1.29E-09	1.46E-08	2.47E-13	23	0.21

Table 2 shows the comparison of the best SSQ's for Wilson data obtained from various techniques such as genetic algorithm (GA), immune clonal selection algorithm (ICSA), discrete-variable harmony search (HS), parameter-setting-free harmony search (PSFHS), Broyden-Fletcher-Goldfarb-Shanno (BFGS) technique and PSO algorithm.

Chu and Chang (2009) found the best SSQ value as 36.89 by using PSO algorithm. The ranges of the parameters given by the authors are $K = 0.01 - 0.20$, $\chi = 0.2 - 0.3$

and $m = 1.5 - 2.5$. Their value of $\chi = 0.3330$ is greater than 0.3. Also, it should be noted that authors used PSO algorithm with different Muskingum model which utilizes an average inflow in Equation (2). So, their result was not compared in Table 2. As can be seen in Table 2, the SSQ (36.767888) values obtained by proposed PSO algorithm is better than all techniques given in the literature. Also, it can be seen that obtained three parameters are in feasible range.

Table 2. Comparison of the best SSQ's obtained by different methods for Wilson (1974) data

Method	K	χ	m	SSQ
GA	0.1033	0.2813	1.8282	38.2363
ICSA*	0.0884	0.2862	1.8624	36.8026
HS	0.0883	0.2873	1.8630	36.7829
PSF-HS	0.0864	0.2869	1.8677	36.7680
BFGS	0.0863	0.2869	1.8679	36.7679
PSO	0.086249	0.286917	1.868087	36.767888

*Luo and Xie (2010) reported that SSQ value was 35.64. But this value is not correct for parameters given by the authors. As reported by Geem (2010), the correct SSQ value is 36.8026.

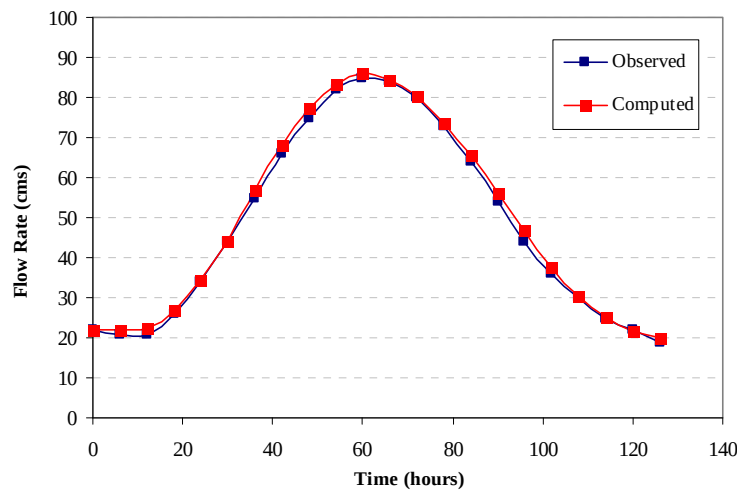


Figure 1. Comparison of the observed and computed hydrographs of Wilson data for the best solution vector

Computed outflows of Wilson data which were obtained by different methods are given in Table 3. As can be seen in Table 3, computed outflows from PSO are better than the other methods. The comparison of the observed and computed hydrographs of Wilson data for the best solution vector is presented in Figure 1. As can be seen in Figure 1, computed hydrograph is well suited to the observed hydrograph.

Table 3. Comparison of the Observed and Computed Outflows for Wilson Data

Time (h)	I_t (cms)	O_t (cms)	Computed Outflows (cms) obtained by different methods					
			GA	ICSA	HS	PSF-HS	BFGS	PSO
0.0	22.0	22.0	22.0	22.0	22.0	22.0	22.0	22.0
6.0	23.0	21.0	22.0	22.0	22.0	22.0	22.0	22.0
12.0	35.0	21.0	22.4	22.4	22.4	22.4	22.4	22.4
18.0	71.0	26.0	26.4	26.6	26.6	26.6	26.6	26.6
24.0	103.0	34.0	34.2	34.4	34.4	34.5	34.5	34.5
30.0	111.0	44.0	44.2	44.2	44.1	44.2	44.2	44.2
36.0	109.0	55.0	57.0	56.9	56.8	56.9	56.9	56.9
42.0	100.0	66.0	68.2	68.1	68.1	68.1	68.1	68.1
48.0	86.0	75.0	77.2	77.1	77.1	77.1	77.1	77.1
54.0	71.0	82.0	83.3	83.3	83.3	83.3	83.3	83.3
60.0	59.0	85.0	85.7	85.9	85.9	85.9	85.9	85.9
66.0	47.0	84.0	84.2	84.5	84.5	84.5	84.5	84.5
72.0	39.0	80.0	80.2	80.5	80.6	80.6	80.6	80.6
78.0	32.0	73.0	73.3	73.6	73.7	73.7	73.7	73.7
84.0	28.0	64.0	65.1	65.3	65.4	65.4	65.4	65.4
90.0	24.0	54.0	55.8	55.9	56.0	56.0	56.0	56.0
96.0	22.0	44.0	46.7	46.6	46.7	46.7	46.7	46.6
102.0	21.0	36.0	38.0	37.8	37.8	37.8	37.7	37.7
108.0	20.0	30.0	30.9	30.5	30.5	30.5	30.5	30.4
114.0	19.0	25.0	25.7	25.3	25.3	25.2	25.2	25.2
120.0	19.0	22.0	22.2	21.8	21.8	21.7	21.7	21.7
126.0	18.0	19.0	20.3	20.0	20.0	20.0	20.0	20.0

5 CONCLUSIONS

In this study, a PSO based parameter estimation technique for the nonlinear Muskingum model is proposed. In the proposed method, non-negativity restrictions are imposed on the numerical solution with an effective penalty approach. The proposed algorithm finds the global or near-global minimum regardless of the initial parameter values with fast convergency and few control parameters. The proposed algorithm found the best solution among 10 different methods. The results obtained for this application demonstrate that the proposed algorithm can confidently be applied to estimate optimal parameter values of the nonlinear Muskingum model.

REFERENCES

- Ali, M.M., and Törn, A. (2004). Population set-based global optimization algorithms: some modifications and numerical studies, *Computer & operations research*, 31: 1703-1725.
- Chu, H-J., and Chang, L-C. (2009). Applying particle swarm optimization to parameter estimation of the nonlinear Muskingum model, *Journal of hydrologic engineering*, 14: 1024-1027.
- Das, A. (2004). Parameter estimation of Muskingum models, *Journal of irrigation and drainage engineering*, 130: 140-147.
- Das, A. (2009). Discussion of “applying particle swarm optimization to parameter estimation of the nonlinear Muskingum model, *Journal of hydrologic engineering*, 14: 1024-1027.
- Geem, Z. W. (2006). Parameter estimation for the nonlinear Muskingum model using the BFGS technique, *Journal of irrigation and drainage engineering*, 132: 474-478.
- Geem, Z. W. (2010). Parameter estimation of the nonlinear Muskingum model using parameter-setting-free harmony search algorithm, *Journal of hydrologic engineering*, doi:10.1061/(ASCE)HE.1943-5584.0000352.
- Gill, M. A. (1978). Flood routing by Muskingum method, *Journal of hydrology*, 36: 353-363.
- Haktanir, T., and Ozmen, H. (1997). Comparison of hydraulic and hydrologic routing on three long reservoirs, *Journal of hydraulic engineering*, 123: 153-156.
- Karahan, H. (2009). Predicting Muskingum flood routing parameters using spreadsheets, *Computer applications in engineering education*, doi:10.1002/cae.20394.
- Karahan, H. (2011). Obtaining regional rainfall-intensity-duration-frequency relationship curves by using differential evolution algorithm, *Scientific research project of TUBITAK (108Y299)*, Final report, Denizli, Turkey (in Turkish).
- Kennedy, J., and Eberhart, R. (1995). Particle swarm optimization, *Proceedings of IEEE international conference on neural networks IV*, 1942-1948, 1995.
- Kennedy, J., and Eberhart, R.C. (2001). *Swarm Intelligence*, Morgan Kaufmann.
- Kim, J.H., Geem, Z.W. & Kim, E.S. (2001). Parameter estimation of the nonlinear Muskingum model using harmony search, *Journal of american water resources association*, 37: 1131-1138.
- Luo, J., and Xie, J. (2010). Parameter estimation for the nonlinear Muskingum model based on immune clonal selection algorithm, *Journal of hydrologic engineering*, 15: 844-851.
- McCarthy, G. T. (1938). The unit hydrograph and flood routing, *Conference of North Atlantic Div.*, U.S. Army Corps of Engineers.
- Mohan, S. (1997). Parameter estimation of nonlinear Muskingum models using genetic algorithm, *Journal of hydraulic engineering*, 123: 137-142.
- Poli, R. (2007). An analysis of publications on particle swarm optimisation applications, *Technical Report CSM-469*, Department of Computer Science, University of Essex, UK.

- Poli, R. (2008). Analysis of the publications on the applications of particle swarm optimisation, *Journal of artificial evolution and applications*, 1-10.
- PSO Tutorial (2011). <http://www.swarmintelligence.org/tutorials.php>
- Ratnaweera, A., Halgamuge, S.K. and Watson H.C. (2004). Self-organizing hierarchical particle swarm optimizer with time-varying acceleration coefficients. *IEEE Transactions on Evolutionary Computation*, 8(3): 240-255.
- Shi, Y., and Eberhart, R. (1998). A modified particle swarm optimizer. *Proceedings of the IEEE international conference on evolutionary computation*, Anchorage, Alaska, 69-73.
- Singh, V.P. (1988). *Hydrologic systems*, Vol. 1: rainfall-runoff modelling, Prentice Hall, NJ.
- Singh, V.P., and McCann, R.C. (1980). Some notes on Muskingum routing coefficients, *Journal of hydrology*, 48: 343-361.
- Tung, Y. K. (1985). River flood routing by nonlinear Muskingum method, *Journal of hydraulic engineering*, 111: 1447-1460.
- Wikipedia (2011). http://en.wikipedia.org/wiki/Particle_swarm_optimization.
- Wilson, E.M. (1974). *Engineering hydrology*, MacMillan Education Ltd., Hampshire, U.K.
- Yagiz, S., and Karahan, H. (2011). Prediction of hard rock TBM penetration rate using particle swarm optimization, *International journal of rock mechanics & mining sciences*, 427-433.
- Yoon, J. W., and Padmanabhan, G. (1993). Parameter-estimation of linear and nonlinear Muskingum models, *Journal of water resources planning and management*, 119: 600-610.