Advancement in Scientific and Engineering Research
Vol. 1(2), pp. 22-34, October 2013
ISSN: 2384-7336
Research Paper

Hybrid neural network models for rainfall runoffs: Comparative study

A. A. Ojugo[1]* • R. Abere[1] • A. O. Eboka[2] • M. O. Yerokun[2] • R. E. Yoro[3] • C. C. Onochie[2] • D. Oyemade[2]

[1]Department of Mathematics and Computer Science, Federal University of Petroleum Resources Effurun, Delta State, Nigeria.
[2]Department of Computer Science, Federal College of Education Technical, Asaba, Nigeria.
[3]Delta State Polytechnic Ogwashi Uku, Delta State, Nigeria.

*Corresponding author. E-mail: ojugo_arnold@yahoo.com.

Accepted 12th August, 2013

Abstract. The study aims to compare hybrid Artificial Neural Networks that have been employed as appropriate models in the prediction of rainfall with historical data as explanatory variables. The selected area is the Chad River Basin in Nigeria with collected data set over the period from 1996 to 2007. A structured analysis of the existing system (SIMHYD) shows bottleneck such as excessive data requirement, large computational demand, and the fact that validation is still an on-going process. The 12-years historical data each on rainfall, relative humidity, cloud cover, temperature difference, and sunshine from the National Metrological Center Oshodi in Nigeria, with lag plot of the cross correlation values of rainfall with each of the other variables to choose explanatory variables that exerts significant influence on rainfall. The dataset was split into three dataset for the developed ANN model into these percentages: training (50%), Cross Validation (25%) and Testing (25%). This was used to validate results obtained, which shows significant correlation of rainfall as established for relative humidity, cloud cover and temperature difference (that were used as explanatory variables in this study). The Tansig activation function was adopted for the three explanatory variables and two parameter weights for each variable identified as the most appropriate model for modeling and predicting rainfall in Chad. Various ANN hybrids have been successful in their implementation, showing high degree of accuracy with many practical implications to water resource operations as well as provide lead time warning in flood management. Results show computed COE as 58, 24, 56 and 42% respectively for the various stations. Observed annual rainfall variations from long-term runoff, is an effect of variation cycle with significant correlation between rainfall and runoff (as indicative in the dataset used). The study implementation will create a synergy between Artificial Intelligence and other fields, which in this case, hydrology via the hybrid ANN models, so that the trained system will help simulate future flood and provide, lead time warning in flood management.

Keywords: Catchment, stochastic, algorithms, evolutionary, fitness function.

INTRODUCTION

Rainfall as an environmental factor can change fast, and can have significant influence in flood/stream hydrology with downstream implication of erosion, water quality, and design of engineering structures. And in turn, affects quality of life, agriculture, sewage system and tourism among others. Long term projections of such dynamic phenomena are often prone to error, its adaptation quite expensive due to its expansive and dynamic nature, making its prediction technically difficult (Conway et al., 1998). These predictions improvise to propagate input data (with noise, ambiguity and assumptions) as applied to the model to yield an output via optimization methods (Govindaraju, 2000; Haykin, 1999).

Various studies have successfully employed stochastic models for enhancing accurate rainfall prediction via optimization (that aims to find an optimal solution in a task, chosen from set of possible solution or search space), to yield an output guaranteed of high quality, void of ambiguity and perfects assumptions made. Model tuning have adapted advances in artificial intelligence to yield Evolutionary models, that is capable of performing quantitative processing to ensure qualitative knowledge and experience, as a new form of natural language (Abarghouei et al., 2009; Gaume and Gosset, 2003).

Evolutionary model via soft computing (SC) aims at synergy between Artificial Intelligence and other fields, and dedicated to solve problems via exploit of data and human knowledge, that is expressed inmathematic models and symbolic reasoning, to yield a methodtolerant to imprecision, uncertainty, partial truth and noise at its input (Coello et al., 2004; Abbott et al., 1986). They are metaheuristic method for constraint satisfaction problems in vector space with intelligent agents that searches the space for optimal fitness as inspired by evolution, behavioural patterns in biological populations and natural laws. They mimics agents seeking food, and have proven efficient in complex optimization (French et al., 1992). These include Genetic Algorithm, Artificial Neural Network, Ant Colony, Particle Swarm, Annealing, Fuzzy logic etc (Branke, 2001a; Beven, 2001a).

Robust optimization in its attempt to explore dynamic processes has 3-feats: robustness, continuous adaptation and flexibility. Study considers output feats – with uncontrollable constraints modeled in ANN not explicitly presenting a space but confined to real parameters and often limited by boundary values (Burnash, 1995; Campolo et al., 1999).

Artificial neural networks are trials in an attempt to translate into mathematical models, principles of biological processing so as to generate in the fastest time period, implicit and predictive model evolution of a system. Thus, derives from experience its ability to recognize feats and behaviours from historic data, and can suggest optimal fitness of high quality and void of *over-fitting*, irrespective of modification via other approximations with multiple agents. These constantly affect the quality of any solution (Dawson and Wilby 2001a). This work showcases ANN hybrids for RR – exploring structural differences and implications of multi-agent populated models (agents create their own behavioral rules based on historic RR dataset).

LITERATURE REVIEW

Models are tools for insight and knowledge about a future state. Its reliability can be questioned as results rather viewed as prophesy (as it presents likely future of variables that are important for optimal operation) than certainty (Govindaraju, 2000).

Rainfall-Runoff prediction estimates are made via various mathematical models that can be grouped into: (a) knowledge models – has its long-standing application that focuses on runoff quantification. The increasing awareness and dynamic nature of these environmental problems, has given additional impetus to hydrological modeling. Such models must meet new requirements, when they are intended to deal with other tasks such as erosion, land degradation, leaching of pollutants, irrigation, sustainable water/flood resource management, land-use possible consequence and climate changes (French et al, 1992). Despite efforts in the last two decades, hydrological models are still faced with the fundamental problem of calibration and validation (Ojugo et al., 2013a) due to limited data availability and natural heterogeneity of RR-process (Shamseldin, 1997) and (b) Data (AI) model – that have arose from the need to learn feats in time that is not possible via knowledge driven models.

Also, many problems are related to model testing such that traditional tests like split-sample are often insufficient to evaluate a model's validity and assess its pros/cons of the different model approaches. The need for additional data has been emphasized, advent of more powerful tests required and different dimensionality of model adopted via data driven models that employ evolutionary method to yield such dimensionality (Oreskes et al., 1984).

Runoff model modes

Validation in RR implies a model is tested with independent data. It demonstrates that a given model is capable of making accurate predictions for periods outside a calibration, and asserts if an underlying model's concept is adequate for a certain catchment to allow for discrimination between good and bad model hypotheses. It also is used to test if a model can be applied (fitted) to a catchment or for a sensitivity analysis of the model parameters (Openshaw, 2013; Rientjes, 2004).

Hjemfelt and Wang (1993) notes that only a model application can be validated, but a general non-site specific model validation is not possible. Also, the primary aim of a model application is to demonstrate that the model works for this particular application and its suitability for similar problems. Often the implicit argument can be found that a model is assumed to be 'valid' because it has been successfully applied in various studies. Model usage by research groups, expresses some confidence level, though the model of choice is based on non-scientific reasons such as freely distributed, user-friendly etc (Varoonchotikul, 2003; Seibert, 2000).

Groundwater models mode

Validation has been used extensively in groundwater models in the use of models in assessing the safety of underground disposal of nuclear and toxic waste. Khondker et al. (1998) asserts validation as impossible, and that models only can be invalidated. They provide examples demonstrating the limited accuracy of model predictions and argue that verification and validation are misleading. These terms cannot be used as they convey an impression of correctness, which cannot be justified scientifically to the public. Validation in groundwater modeling is used for assessing a model's goodness of fit and such an assessment is possible. On groundwater modeling and models in other earth sciences, Halff et al. (1993) argue that model verification and validation is impossible; and rather, that models can only be confirmed by demonstrating that their simulations agree with observations. This confirmation is only partly possible and thus, concludes that the main benefit of models is heuristic, as models are preliminary hypotheses assisting in gaining better understanding.

Ecological models

Nishimura and Kojiri (1996) reviewed such studies and conclude that much of the confusion and the mutually exclusive statements about model validation arise from varying semantic and philosophical perspectives and from different validation procedures. Tarboton (2003) compare techniques for validation, which they interpret as the comparison of simulated and observed data without the specification whether this data has been used for model development or calibration or not. De Vos et al. (2005) define validation as the process of evaluating the level of confidence in the model's ability to represent the problem entity, and emphasize that a model cannot be expected to be absolutely valid.

Seibert (1999) surveyed but note these distinctions between three types of validation as thus: (a) Replicative validity ensures prediction agrees with observed data used for model design and parameter estimation, (b) predictive validity ensures model can accurately simulate a variable or time period, which has not been used in model development and calibration, and structurally valid if it reflects the main couplings and behaviour of a real system, and (c) specific validity ensures the generally accepted standard for model testing and validating is adequate for a special purpose.

The latter, note to some degree that all models are unrealistic as well as emphasize that parameter calibration and the use of *ad hoc* model features often makes validation less rigorous, so that even inadequate models are likely to pass the tests (Gaume and Gosset, 2003).

Validation: Semantic/philosophy in study

Alternative terms to validation have been proposed. Dibike and Solomantine (1999) suggests that 'history matching', but it does not distinguish between historical data used in training, and those used for independent testing. He further propose "confirmation" – one of the words listed as explanation of 'to validate' in the dictionary (Hsu et al., 1995) and 'to confirm' points to, among others, 'to make definitely valid'. Thus, 'confirmation' is not less ambiguous than the term 'validation'.

Dibike and Solomantine (2001) suggests "corroboration" and the degree of corroboration describes the degree to which a hypothesis has passed the tests. Similar to 'confirmation' the word 'corroboration' does not in common use, express the limited and provisional acceptance better than 'validation' does. The word 'valid' is Latin for *validus* (strong, powerful, well-grounded) and means sound or defensible, and differs from words like true or correct which are connected to the process of verification (Latin word *verus*, true). There is a need for a generally accepted semantics to describe the qualifications of model predictions.

Here, we adopt a more concise (though generally criticized) term validation to be appropriate for use in connection to model testing. With reference to the false 'impression of correctness' it seems to be of importance to clearly state what is (not) meant by validation rather than to define a new term. In this study, "verification" is inappropriate in model testing, though found in some studies – to imply establishment of truth that is hardly possible in science and absolutely not in modeling. Thus, the philosophy of our use of validation in this study – implies predictive validity as proposed by Gaas (1983).

Problem statement

The need arose from these:

a) The dynamic nature of RR and its conceptual models are flawed with unfounded results.
b) Data driven models that have adopted ANN most often use hill-climbing method whose solution may get stuck at local minima, because their speed shrinks as model approach optima.

Purpose/significance of study

Study implements 3-hybrid ANN algorithm for RR model to help speed up the final stages of ANN and find robust optima in a shorter amount of time, in large and complex tasks. Hybrids, though difficult to implement, yields better selection and are encoded via structured learning (to address problem of existing statistical dependencies amongst data variables) and yield better generation with crossover, mutation and temperature function etc. Its application will yield computational intelligence for dynamic multipoint search in CSPs and be adapted to other areas such as image/video analysis, communication, control/design, OS task scheduling, parallel processing, medicine, security, etc.

MATERIAL AND METHODS

Selected area is BORDA (Nigeria) with landmass of

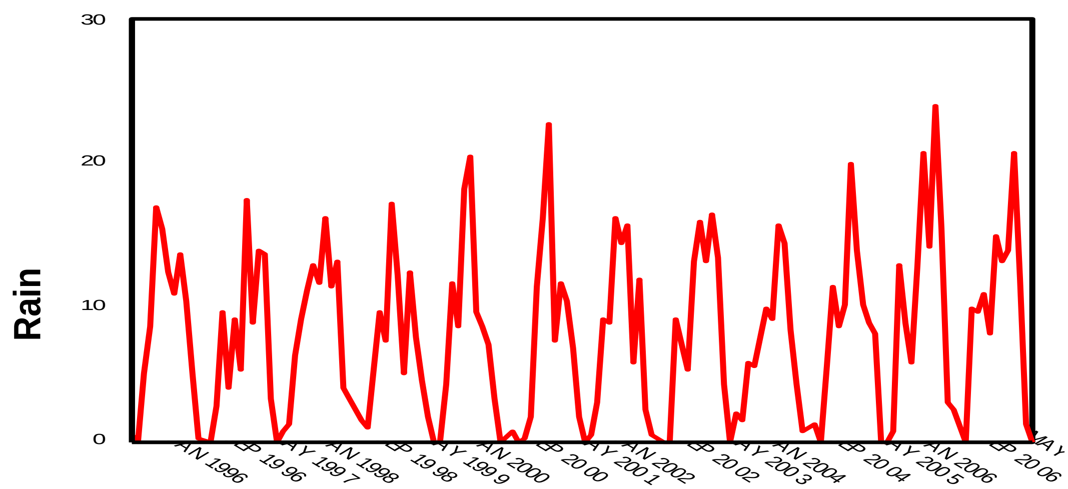

Figure 1. Time plot of rain.

22045 km^2, mean rainfall of 846 mm annually and perennial discharge of 3.8 m and 1.5 m^3/s respectively for dry and peak periods. Dataset collected (2003 to 2013) is split into 3-sets: *training* (45%), *cross-validation* (25%) and *validation* (30%). All fragment starts at period of constant low rainfall.

Pre-data analysis

The time plot for data collected within the period 1996 to 2007 is as seen in Figure 1 and Table 1.

Artificial neural network

Artificial neural network (ANN) as a data processing model is inspired by biological neurons of the human brain, and consists of interconnected neurons, whose major feat is in their ability to *learn* by example via simulation, making them universal estimators (Abraham, 2005). The brain learns in its behavior to process data, as neuron shares electrochemical signals amongst themselves via dendrites (Caudill, 1987; Fausett, 1994). These signals via synapses axon are converted so that learning occurs by adjusting synapse's weight, whose input is summed by an *adder*. This depends on the task and an activation function, can limit its output amplitude (Mandic and Chambers, 2001). The simple mathematical model for a synapse as weight connections that modulate its associated inputs and the nonlinear feats exhibited in neurons via transfer/activation function (sum of weighted input) is given by Equation 1:

$$\emptyset = f(net) = f\sum_{i=1}^{m} Xi * Wij \qquad (1)$$

Encoded, ANN has three basic layers: input, hidden and output, and two network configurations namely: (a) feed-forward in which data flows from input to output with no feedback, but extends over multiple layers, and (b) recurrent has a feedback with dynamic feats that undergoes relaxation to evolve a network to stable state where activation values and output changes no more. For some tasks, the output change is significant and dynamic behavior constitutes its output (Ojugo et al., 2013b).

The configurations are dependent on the application area, feats and system requirement (Beven, 2001b; Bishop, 1995). Various methods are used to set the connection strengths so that learning can take place namely: (a) explicit connection via a priori knowledge, and (b) implicit connection via *post-priori* in which the network is trained to learn patterns that changes its weight in a learning rule (Abraham, 2005; Beven and Binley, 1992).

Learning is grouped into: (a) supervised in which an input vector with a set of desired responses, one for each node, is relayed to the output. A forward pass is done and errors between *desired* and *actual* response for each node in the output is found, and then used to determine weight changes in the net based on the learning algorithm (Gupta et al., 1998; Hall, 2001). Thus, desired signal on output is provided by external teacher via back-propagation, delta rule and perceptron rule, (b) unsupervised or self organization, in which an output is trained to respond to clusters of patterns at its input so the system discovers statistically, salient feats of an input population with no prior knowledge how patterns are grouped; rather the system develops its own representation of input (Hsu et al., 1995; Heppner and Grenander, 1990), and (c) reinforcement in which network learns what to do, map states to actions to help maximize a numerical reward data.

Network must discover the actions that yield most reward by trying them. Sometime, such actions affect not only the immediate data, but also the rest states (Jang, 1993; Kennedy et al., 2001).

The two feats, trial/error search and delayed reward are its two distinguishing feats (Gupta et al., 1998; Jain and Srinivasulu, 2004; Jang, 1993; Lindstrom et al., 1997). Two basic architectures used in this study are feed forward and recurrent, which enables training of the network to predict future runoff via real-time data retrieved from an online data acquisition system that flows strictly from input to its output with historic analysed data as feedback connection.

ANN adopted architecture

The Multilayer Perceptron is interconnected via correlated weights, sums weighted input via Tansig activation functions as in Equations 2 and 3 to compute its output, from signals sent to all nodes in the hidden layer. $W_{i,j}$ is weight between input and hidden layers, $W_o j$ is bias weight and x_i is rainfall input signal sent via activation function to produce result. Thus, we adopt a tangent/sigmoid function (Minns, 1998).

$$Z_{ij} = w_o j + \sum_{i=1}^{m} x_i * w_i j \tag{2}$$

$$F(Z_{ij}) = \frac{2}{1+ e^{-2*Zij}} - 1 \tag{3}$$

The Time-Lag Recurrent Network is adopted as it is an extended MLP with short-memory that have local recurrent connections, requiring a smaller network to learn temporal tasks (compared to MLP that use extra inputs to represent the past samples). TLRN is more plausible and computationally more powerful than other adaptive models. It uses back propagation in time (an advanced training algorithm) learning so that its output at time t is used along with a new input to compute the network's output at time t+1 in response to dynamism (Mandic and Chambers, 2001). Also, the Elman in which the hidden unit's activation values are fed-back as extra set of input, and Jordan in which output values are fed back to hidden units (Rajurkar et al., 2004; Karunanithi et al., 1994). Thus, output is computed via Tansig activation function given as y^k, sums input, receives target value of input training pattern, computes error data, weight correction updates in layers (c_j^k) and bias weights correction updates (c_o^k). This error is sent from output layer back to input nodes via error back propagation, to correct the weights. Back propagation is the most used training algorithm in a multilayer feed-forward networks, whose goal is to find the weight that approximates target values of output with selected accuracy. The weights are modified by minimizing error between target and computed outputs at the end of each forward pass as in Equation 7. If error is higher than the selected value, process continues with a reverse pass; else, training stops. Weights in BP are updated via mean square error continuously until minimal error is achieved (Ursem et al., 2002).

Radial basis function

Recurrent networks are nonlinear time series, identification and pattern classification. A simple recurrent net is constructed by modifying the multilayered feed forward with addition of 'context layer' to help the net retain data between observations. At each move, new inputs are fed to the network. Previous contents of hidden layer is passed into context layer and later fed back into the hidden layer in the next time step (Regiani and Rientjes, 2005). The context layer contains nothing initially and output from the hidden layer after the first input to the network, will be the same as if there is no context layer (Perez and Marwala, 2011). Weights are calculated same way for the new connections from and to the context layer from the hidden layer. Its weight and Euclidean distance measure input distance from its center to the best, in this curve-fitting high dimensional space (Shamseldin, 1997). Its training finds weights with learning of best fit to train data, as its Guassian transfer functions that assumes an approximation influence which data points have at the center, so that function decreases with distance from its center. The Euclidean length (rj) measures the distance between datum vector y = (y1, ..., ym) and center (w_{1j}, ..., w_{mj}):

$$r_j = \left|\left|y - Y^j\right|\right| = \left\{\sum_{i=1}^{m}(y_i - w_i j)^2\right\} ½ \tag{4}$$

The suitable transfer function is applied to r_j:

$$\emptyset(r_j) = \emptyset||y - Y^j|| \tag{5}$$

Finally, output k receives weighted combination as:

$$y^k = w_o + \sum_{j=1}^{n}\left(c_j^k * \emptyset(r_j)\right) = w_o + \sum_{j=1}^{n}(c_j^k * \emptyset||y - Y^j||) \tag{6}$$

Experimental models

Study adoptsTLRN architecture with *unsupervised* learning and RBF a control model to compare the results obtained by training the network to generate satisfactory results and provide a fail-safe to eradicate noise in data-stream processed in real-time. The network learns from experiences, generalized from previous datasets to new ones with abstract feats, at its inputs containing irrelevant data (Sedighizadeh and Masehian, 2009; Seibert, 2000).

Trial-error is used in selecting number of hidden layers and nodes in each hidden layer. Previous results have shown that ANN with a hidden layer outperforms those with two/more, as this only increases the number of parameter that only complicates training (Ursem et al., 2002). The optimal hidden layer size is found by systematically increasing the number of hidden node until network's performance shows no further improvement or it longer improves significantly. The network is complex enough to accurately simulate dynamic, nonlinear feats. Model performance for our study adopts a *single* hidden layer with 18-hidden nodes (Tokar and Johnson, 1999) with respect to rainfall, previous discharge and evo-transpiration) as supported in Sajikumar and Thandaveswara (1999), Bishop (1995) and Reynolds (1994).

Gravitation search algorithm

Gravitation search algorithm (GSA) is based on Newton's laws of gravity and motion with its main idea, being to consider isolated system of masses, where every mass represents a solution to a certain problem. Law of gravity states that every particle attracts another and the gravitational force between particles are directly proportional to the product of their masses and inversely proportional to distance between them (Varoonchotikul, 2003; Ojugo, 2009). Thus, an agent's performance depends on its mass as they attract each other via gravitational force (a pull towards those of heavier masses). N agents initializedat start point, are randomly located in space so that gravitational force is defined as thus:

$$Fij = G(t) = \frac{Mi(t) * Mj(t)}{Rij(t) + \varepsilon}\{Xj(t) - Xi(t)\} \tag{7}$$

R_{ij} is the Euclidean distance between masses for the objects (i and j) masses, G(t) is gravitation force at time t with small constant ε – which decreases in time to control the population and search's accuracy. Thus, the total force acting on an agent is:

$$F_i^d = \sum_{j \in kbest,\, j \neq 1} rand(i) * Fij \tag{8}$$

rand randomizes agents'initial state at intervals [0,1]. Acceleration of i at time t, in *d* dimension is directly proportional to force acting on agent I, and inversely proportional to agent's mass:

$$Aid(t) = \frac{Fid(t)}{Mij(t)} \tag{9}$$

The next velocity of an agent is a function of its current velocity plus current acceleration, which updates next position given by X as thus:

$$Vid(t+1) = rand(i) * Vid(t) + Aid(t) \tag{10}$$

$$Xid(t+1) = Xid * Vid(t+1) \tag{11}$$

$V_i^d(t)$ is agent velocity in *d* at time t, rand is a random number between [0,1]. Mass is updated as fitness value of agent i at time t given as:

$$Mi(t) = \frac{Fit(i) - worst(t)}{best(t) - worst(t)} \tag{12}$$

Best(t)/Worst(t) are strongest/weakest agents from their fitness route. For a Max task, they are defined:

$$worst(t) = \max_{j \in \{1,2 \ldots N\}} Fit(t) \tag{13}$$

$$best(t) = \min_{j \in \{1,2 \ldots N\}} Fit(t) \tag{14}$$

At start, agents are located as solution points trained in ANN, and then passed over to GSA so that with each cycle, agent velocity and position is updated via Equations 11 and 12; while G and M are computed at each of the iterations via Eq. 8 and 13. The algorithm is stopped with a stop criterion (computational expensive), or if an optima is found. GSA use exploration ability to navigate that guarantees the choice of value of random agents, and exploitation ability that allows agents of heavier masses to move slower in order to attract those of lesser mass and to locate optima around a good solution in the shortest time as in Figure 4 (Dawson and Wlby, 2001b; Seibert, 1999; Haykin, 1999).

ANN-cultural genetic algorithm

GA as inspired by Darwinian evolution and genetics (survival of fittest), consists of a population (data) chosen for natural selection with potential solutions to a specific task. Each potential solution is an individual for which an optimal is found using four operators: initialize, select, crossover and mutation (Coello et al, 2004; Reynolds, 1994). Individuals with genes close to its optimal, is said to be fit. Fitness function determines how close an individual is to the optimal solution.

Ojugo et al. (2013a, 2013b) notes the operators as:

a) Initialize: From the population, individual data are encoded into format suitable for selection. Each encodings has its merit/demerit. Binary encoding is computationally more expensive to achieve. Decimal encoding has greater diversity in chromosome and greater variance of pools generated; float-point encoding or its combination is more efficient than binary. Thus, it encode as fixed length vectors for one or more pools of different types. The *fitness* function evaluates how close a solution is to its optimal – after which they are chosen for reproduction. If solution is found, function is *good*; else, is *bad* and not selected for crossover. The fitness function is the only part with knowledge of task. The more solutions are found, the higher its fitness value.

b) Selection: Good fit individuals close to optimal are chosen to mate. The larger the number of selected, the better the chances of yielding fitter individuals. This continues until one is chosen, from the last two/three remaining solutions, to become selected parents to new offspring. Selection ensures the fittest individuals are chosen for mating but also allows for less fit individuals from the pool and the fittest to be selected. A selection that only mates the fittest is *elitist* and often leads to converging at a local optima.

c) Crossover: Ensures genes of fitter individuals are exchanged to yield a new, fitter pool. There are two crossover types (depends on encoding type used) as: (a) simple crossover for binary encoded pool via particular- or multi- point; and all genes are from one parent, and (b) arithmetic crossover allows new pool to be created by adding an individual's percentage to another.

d) Mutation alters chromosomes by changing its genes or its sequence, to ensure that a new pool converges to global minima (instead of local optima). Algorithm stops if optimal is found or after number of runs (though computationally expensive) if a number of new pools are created or once no better solution is found. Genes may change based on probability of mutation rate. Mutation improves the much needed diversity in reproduction and its algorithm is as thus:

Mutation Algorithm
1. Input: A chromosome rule
2. Output: Mutated solution, a fns of mutation rate
3. Set mutation threshold (between 0 and 1)
4. For each network attribute in chromosome
5. Generate a random number between 0 and 1
6. If random number > mutation threshold then
7. Generate random value
8. Set solution attribute value with
9. Generated attribute value
10. End if: End for Each

Cultural GA is one of the many variants of GA with a belief space as: (a) Normative (has specific range of values which an individual is bound), (b) Domain (data about task domain), (c) Temporal (data about events' space is available), and (d) Spatial (has topographical data). In addition, an influence function mediates between belief space and the pool – to ensure and alter individuals in the pool to conform to belief space. CGA is chosen so as to yield a pool that does not violate its belief space and reduces number of possible individuals GA generates till an optimum is found (Reynolds, 2004).

Once initialized, ANN computes individual fitness and 30-individual are selected as the new sub pool via tournament, to determine mating individuals. Thus, after training, selected data are moved to the CGA model – with only crossover and mutation applied, to help the network model learn dynamic and non-linear feats in the historic, obtained data.

With GA, only crossover (single point) and mutation is carried out, and data between 1 and 30 is randomly generated via Gaussian distribution, corresponding to crossover points (since prior now, all genes are from a parent). Now, other parents contribute the rest to yield new individuals, whose genetic makeup is combination of both parents. They are then allowed to undergo mutation from which 3-random genes are selected for another mutation and are allocated new random values that still conforms to the belief space. The number of mutation applied depends on how far CGA is progressed (how fit is the fittest individual in the pool). Thus, number of mutations equals fitness of the fittest individual divided by 2. New individuals replace old ones in pool, with low fitness values (creating a new pool). This continues until individual with a fitness value of 0 is found, indicating that the solution has been reached (Branke, 2001).

Initialization and selection via ANN ensures first 3-beliefs are met; while mutation ensures the fourth is met. Also, an influence function helps influence how many mutations takes place. Knowledge of solution (how close task is to solution) has direct impact on how algorithm is processed. Algorithm stops when best individual has a fitness of 0 (Campolo et al., 1999; Dawson and Wilby, 2001b).

Hybrid ANN-simulated annealing

SA as inspired by annealing, to strengthen glass and crystals, so that a glass is heated until it liquefies and allowed to slowly cool so that the molecules settles into lower energy states. Thus, it tracks and alters the state of an individual, constantly evaluating its energy via its energy function. Its optimal point is found by running series of Markov chain under different thermodynamic state (Perez and Marwala, 2011). This *neighbouring* state is determined by randomly changing an individual's current state via a neighbourhood function. If a state with lower energy is found, individual moves to it; else, if neighbourhood state has a higher energy, individual moves to that state only, if an acceptance probability condition is met. If not met, individual remains at current state (Kitanidis and Bras, 1980).

The acceptance probability is difference in energies between current and neighbouring states, and temperatures. Temperature is initially set high, so individual is more inclined towards higher energy state, allowing the individual to explore a greater portion of the space and preventing it from being trapped in local optimum. As model progresses, temperature reduce with cooling and individuals converge towards lowest energy states till an optimum point (Ojugo et al., 2013b). Its algorithm is:

1. Initialize individual state, energy and temperature
2. Loop until temperature is at minimum
3. Loop until maximum number of iterations reached
4. Find neighbourhood state via neighbourhood function
5. If neighbourhood state has lower energy than current
6. Then change current state to neighbouring state
7. Else if the acceptance probability is fulfilled
8. Then move to the neighbouring state
9. Else retain the current state
10. Keep track of state with lowest energy
11. End inner loop: End outer loop

This hybrid combines ANN's exploratory search of the space via multiple individuals and SA's flexibility in finding a better optimal point, even when a local minimum are found and present (Dibike and Solomantine, 1999; Dawson and Wilby, 2001).

ANN first yields candidates with low fitness via training and calibration of model. Thus, the model needs to be more robust, so that if a better individual is not found, best individual is chosen after a number of runs for a series of random walks until an optimal solution is found. Some factors must be defined: (1) On ANN: the number of runs, the dataset used for calibration, the population representation within the dataset, the size and cross validation function; Conversely, (2) on SA (with ANN stages complete), SA is run on the chosen "fittest" candidates or individual until a solution is found and what is the neighbourhood size and function. As applied, the initialized dataset is used for training cum calibration of the network (Ojugo et al., 2013a).

Temperature schedule is applied that randomly re-initializes the network for the series of Markov chain about to be run. Neighbourhood function is then applied to randomly change individual energy states and compute fitness function as best fitness and such individual is tracked until a fitness of 0.6 is found, which experimentally is found that ANN finds a 0.6 fitness quickly (Govindaraju, 2000).

ANN finds individuals with low energy, then enters SA cycle early enough to apply the temperature schedule as needed. Thus, a moderated Markov chain is used that accepts the states with energies of lower or equal to current state's energy. This runs till state with energy of 0 is reached (to imply that the solution is found). SA and ANN, shares the same fitness function (Ojugo et al., 2012).

Model performance evaluation

Model performance is computed via mean square error (MSE), mean absolute error (MAE) and mean relative error (MRE), coefficient of efficiency (COE) and determination (r^2). As the most commonly used measures in hydrology, MSE/MRE/MAE has an ideal value 0; while COE and r^2 has ideal value 1 as in Eqs.16-20 with outputs and n observations (Nash and Sutcliffe, 1970):

$$MSE = \, ^1/_n \sum_{i=1}^{m} \{(Ypi - Yio)2\}^{1/2} \tag{15}$$

$$MAE = \, ^1/_n \sum_{i=1}^{m} |Ypi - Yio| \tag{16}$$

$$MRE = \, ^1/_n \sum_{i=1}^{m} \frac{|Ypi - Yio|}{Yio} \tag{17}$$

$$COE = 1 - \frac{(Q_{obs} - Q_{sim})^2}{(Q_{sim} - Q_{obs})^2} \tag{18}$$

$$r^2 = \frac{[(Q_{obs} - (1 - Q_{obs}))(Q_{obs} - (1 - Q_{obs}))]^2}{(Q_{obs} - (1 - Q_{obs}))^2 (Q_{sim} - (1 - Q_{sim}))^2} \tag{19}$$

RESULTS

Tables 2 and 3 are comparative performance values between the hybrids for RBF and TLRN. After testing all hybrids as in Figures 2 and 3 respectively, the results analysis is as presented below:

a) ANNCGA took 21 s to find the solution after 98 iterations (at best). ANNCGA was run 15 times (to eradicate non-biasness), it found optima each time – and the time varied significantly between 21 seconds and 4 min – as its convergence time depends on how close the initial population is to the solution as well as on mutation applied to the individuals in the pool.

b) ANNGSA (at best) 18 s after 321 iterations. GSA employed a gravitational pull and mass update of 282 iterations before finding a solution. It was run 25 times and solution was found each time on a range between 4 s and 3 min. Its convergence time depends on initialization, gravitational cum mass updates.

c) ANNSA arrived at solution 2.112 s after 401 iterations. SA used a Markov chain of 387 iterations to find a solution. On 25 runs, solution time is between 3 s and 3 min – and its convergence time depends on initialization and random swaps from temperature schedule as applied

Table 1. RR parameter for Oshodi (1996 - 2007).

Area	Mean	Std Dev	Coeff. Var	Max Rainfall	Min rainfall
Benin	823	359	58	4532	142
Ekpoma	732	299	24	1034	102
Sapele	962	420	56	4320	127
Agbor	734	343	42	1354	156

Table 2. Simulated values for RBF network.

Item	ANNGSA	ANN-CGA	ANNSA
Training/calibration phase			
MAE	0.4512	0.6645	0.6665
MRE	0.5234	0.5468	0.6518
MSE	0.6799	0.6938	0.5329
COE	1.5622	1.5902	1.8320
R^2	1.4390	1.3456	1.4523
Testing/validation phase			
MAE	0.3527	0.2196	0.1265
MRE	0.4982	0.3762	0.2618
MSE	0.3392	0.3298	0.1203
COE	0.9456	1.1023	1.1209
R^2	1.3430	1.0023	1.1102

Table 3. Simulated values for TLRN.

Item	ANNGSA	ANN-CGA	ANNSA
Training/calibration phase			
MAE	0.6344	0.6045	0.6501
MRE	0.7234	0.7230	0.6815
MSE	0.7199	0.6953	0.5796
COE	1.7220	1.7002	1.7324
R^2	1.5401	1.5450	1.5023
Testing/validation phase			
MAE	0.4209	0.4387	0.6215
MRE	0.4701	0.6420	0.6108
MSE	0.5401	0.6253	0.2003
COE	0.9906	1.3056	1.3009
R^2	1.5234	1.0923	1.2102

and supported by (Haykin, 1999).

DISCUSSION

With a common solution space, fitness function and selection criteria, note that with weights set between 0.2/0.35, and biases = 0.75 yields a better and faster convergence, before they are crossed over to CGA, GSA and SA. Other values, led to a slower convergence and sometimes, non-convergence.

The RBF network as a control models with a hidden layer(s) and various nodes within the hidden layer as in (Abraham, 2005) reflect a model's performance that is feasible for prediction. Its inaccuracy is clarified via longer training and larger dataset (as long as overtraining does not occur). Result showed performance improved at testing with greater efficiency (proper training and parameter selection – void of over fitting, over parameterization and over training), and predicted values yielded better result for stations with larger size. RBF was also found to be more easily trained to learn data feats and consistently outperforms other techniques used – as ANN performance is hardly influenced by non-linearity and data selection).

The number of nodes in hidden layer also influenced performance. If it is small, network may not achieve its accuracy. If it is too many, it may result in overtraining. Large stations generate higher peaks at training, and the use of two hidden layers in such substations is a merit; while smaller catchment is sufficiently handled by single hidden layer. The general pattern of rainfall over the period, and annual totals indicate spatial and temporal variability due to cohesive relationship between rainfall and runoff. Factors affecting runoff are more uniform for smaller catchments and their coefficients of determination increased with decrease in area.

Result tradeoffs

Ojugo (2009, 2013b) notes that various trade-offs in prediction result often fall under these categories:

Result presentation

Researchers often display flawed results, modify/build new models rather than re-test limitations, biasness and inabilities of existing ones – since negative results are less valuable. Data driven model aim to curb non-linearity and dynamism in historic datasets, used to train/test it, unlike knowledge models.

Efficiency

Modelers use figure to show how well their simulations is in agreement with observations (even with their limited data that is squeezed) with lines for observed and simulated runoff that are not easily distinguishable. Some

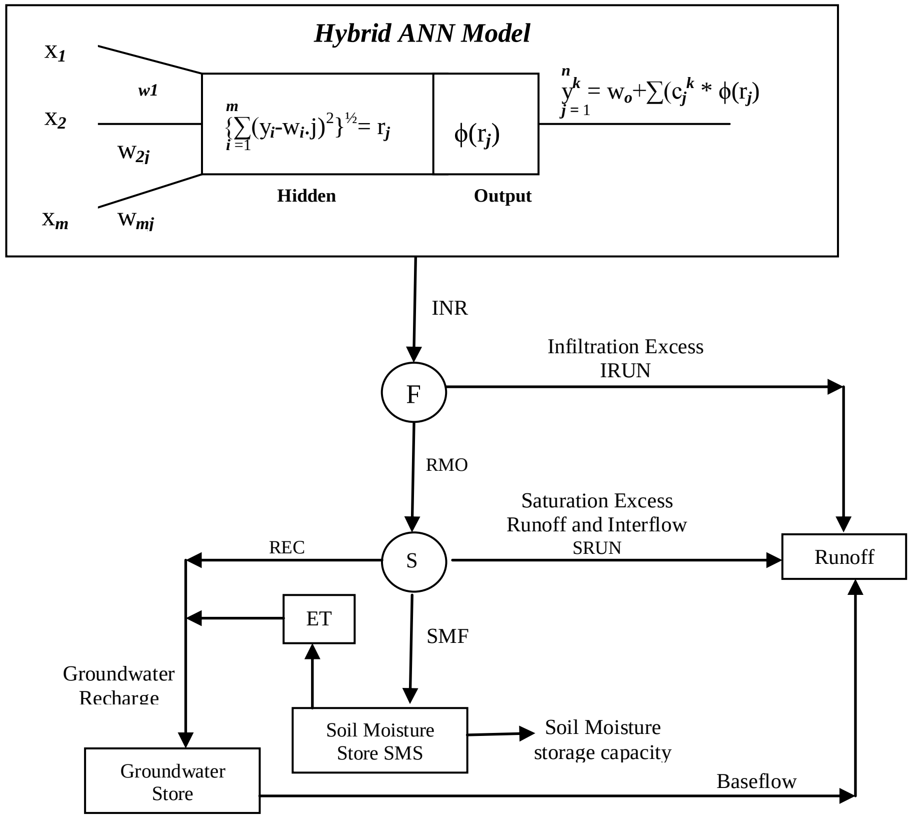

Figure 2. Block diagram of proposed hybrid ANN model design.

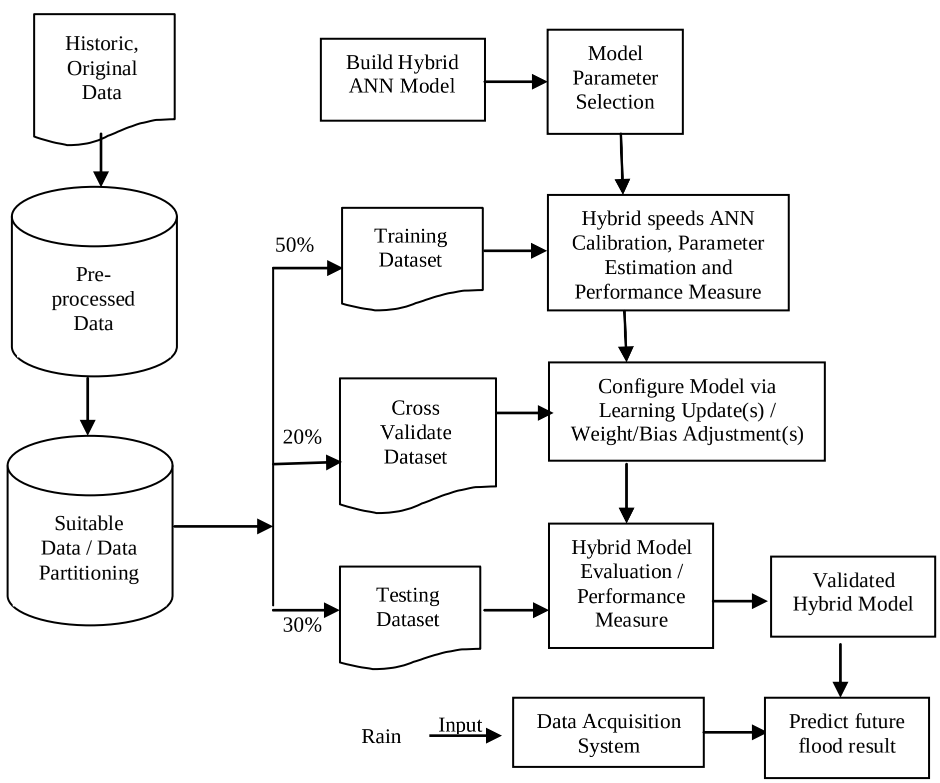

Figure 3. Data flow diagram of the hybrid model.

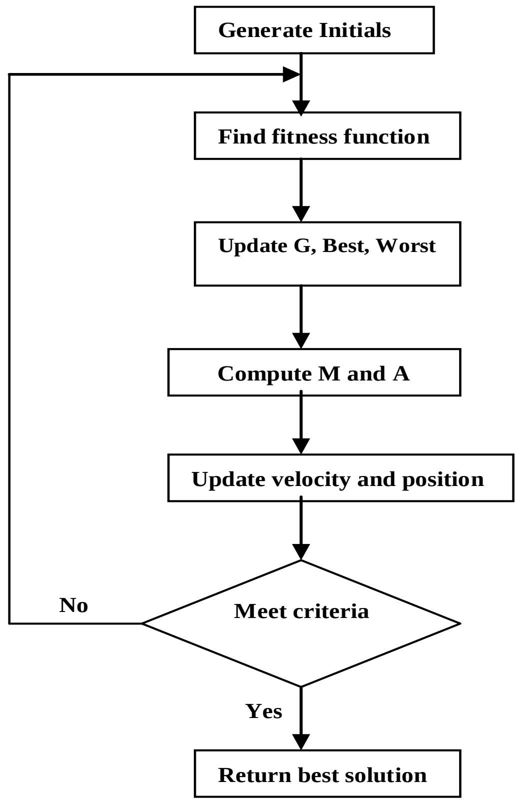

Figure 4. Steps for gravitational search algorithm.

do not provide numerical data; but their model is in 'good agreement' with observations. Some measure of goodness does not provide the relevant information.

Insufficient testing

Validation is a comparison of computed versus observed values, and many studies suffer from inadequate data. If a model aims to simulate more than runoff, such ability is demonstrated in unfounded results with limited data and misleading results and conclusions.

Model validation

Model validation is not an undertaking to be carried out by a researcher or research group; but rather, a scientific dialogue. Improper model applications and ambiguous results often impede such dialogue. The aim of this thesis to a greatly minimize confusion in hydrology models as well as proffer data driven models that will aid in simulation of RR.

CONCLUSION AND RECOMMENDATIONS

Models are useful fictions and/or representation of reality as their primary value is their use as an intellectual tool, to help us better understand and reflect reality. Thus, they support experts in making estimates about the future. As thus, the model is recommended for use in RR-model simulation.

Models are used for prediction, and serve as educational tools to compile existing knowledge about a task, as a language to communicate hypotheses, and to gain better insight or understanding of a task. Model development study of its failure or sensitivity analysis helps reflect theories on functioning of natural systems. A detailed model may not be operationally applicable in larger scale, but allow for study and thus, helps to develop other reasonable and applicable model, for larger scales. Models are used to examine varied hypotheses about a catchment and to investigate which parameter or input data are most crucial to be estimated accurately.

Very simple models do not provide enough new data, whereas very complex models are not understandable. A model's application as an intellectual tool requires less accurate numerical agreement between simulations and observations, but rather requires feedback mechanisms, as more important.

Hybrids are valuable in comprehending such RR processes, and may not necessarily be suitable tool for concrete predictions. Only understood and manageable models are fully explored. There must be a balance for complexity and simplicity, which is crucial for studying RR processes.

Thus, these recommendations are made:

1. Parameters are a major source of uncertainty. Model should have input ranges as computed via Monte-Carlo Integral methods.
2. Multi-criteria training with adequate datasets can help to reduce parameter uncertainty.
3. Prediction is of limited practical use, without clear data about reliability and accuracy.

REFERENCES

Abarghouei A, Ghanizadeh A, Shamsuddin S (2009). "*Advances in* soft computing methods in edge detection", J. Adv. Soft Comput. ISSN: 2074-8523, 1(2):23-30

Abbott M, Bathurst J, Cunge J, O'Connell P, Rasmussen J (1986). "Introduction to Syst`emeHydrologiqueEurop´een 1 and 2: History and philosophy of a physically-based, distributed modeling system", J. Hydrol. 87:45-77.

Abraham A (2005). "Handbook of Measuring System Design", John Wiley and Sons Ltd, ISBN: 0-470-02143-8, pp. 901-918.

Beven KJ, Binley NJ (1992). "Knowledge driven models and computations", Wiley and Sons, Chichester, ISBN: 0-465-08245-3:234-289.

Beven KJ (2001a). "Rainfall-Runoff Modeling: a Primer", Wiley and Sons,Chichester, ISBN: 0-465-01145-7, pp 22 – 54.

Beven KJ (2001b). "How far can we go in distributed hydrological modeling?", Hydrol. Earth Syst. Sci. 5:1-12, SRef-ID: 1607-7938/hess/ 2001-5-1.

Bishop CM (1995). "Neural networks for pattern recognition", Oxford University Press, ISBN: 0-342-45230, pp. 253-294.

Branke J (2001a). "Evolutionary Optimization in Dynamic Environments", Kluwer.

Branke J (2001b). "Reducing the sampling variance when searching for robust solution", In Proceed. Man and Cybern.14(6):234-242, Springler-Verlag.

Burnash RJ (1995). "NWS River Forecast System – catchment modeling, in: Computer Models of Watershed Hydrology", In Singh, V. P., Colorado Water Resour. pp. 311-366.

Campolo M, Andreussi P, Soldati A (1999). "Flood forecasting with a neural network", Water Resources, 35(4):1191-1197.

Caudill M (1987). "Neural Networks Primer, Part I", AI Expert December pp. 46-52.

Coello C, Pulido G, Lechuga M (2004). "Handling multiple objectives with particle swarm optimization", In Proceed. Evol. Comput. 8:256-279.

Conway AJ, Macpherson KP, Brown J (1998). "Delayed time series predictions with neural networks", J. Neurocomput.18:81-89.

Dawson C, Wilby R (2001a). "Comparison of neural networks in river flow forecasting", J. Hydrol. Earth Sci. SRef-ID: 1607-7938/hess/2001 -3-529.

Dawson C, Wilby R (2001b). "Hydrology modeling using neural network", Journal of Programming in Phys. Geogr. 25:80-108.

De Vos NJ, Rientjes TH, Pfister L (2005). "Groundwater levels as indicator in rainfall-runoff modeling using Artificial Neural Networks', In Proceedings of National Council for Rainfall, Australia.

Dibike YB, Solomatine DP (1999). "River Flow Forecasting Using MLP Neural Network", J. Phys. Chem. Earth(EGS1.1-0002):24(5):14-27.

Dibike YB, Solomatine DP (2001). "River flow forecasting using artificial neural networks", J. Phys. Chem. Earth (EGS1.1-0002):26(1):1-7.

Fausett L (1994). "Fundamentals of Neural Networks", New Jersey: Prentice Hall p. 240.

French M, Krajewski W, Cuykendall R (1992). "Rainfall forecasting in space and time using a neural network", J. Hydrol. 137:1-31.

Gaas SI (1983). "Decision-aiding models: validation, assessment, and related issues in policy analysis", Oper. Res. 31:603-631.

Gaume E, Gosset R (2003). "Overparameterization, obstacles of artificial neural networks in hydrology?", J. Hydrol. Earth Syst. Sci. 7:693-706, SRef-ID: 1607-7938/hess/2003-7-69.

Govindaraju RS (2000). "Artificial Neural Networks In Hydrology, Part II: Hydrologic Applications", J. Hydrol. Eng. 5(2):124-135.

Gupta H, Sorooshian S, Yapo P (1998). "Improved calibration of hydrologic model: Multiple, non-commensurable measures of data", Water Resour. 34(4):751-763.

Halff AH, Halff HM, Azmoodeh M (1993). "Predicting from rainfall using neural networks", In Proceed. Eng. Hydrol. 760-765.

Hall MJ (2001). "How well does your model fit data?, J. Hydroinform. 3(1):49-55.

Haykin S (1999). "Neural networks, comprehensive foundation", Prentice Hall: New Jersey pp. 23-89.

Heppner H, Grenander U (1990). "Stochastic non-linear model for coordinated bird flocks", In Krasner, S (Ed.), The ubiquity of chaos. Washington: AAAS pp. 233-238.

Hjemfelt A, Wang M (1993). "Artificial neural networks as unit hydrograph applications", In Proceed. Eng. Hydrol. pp. 754-759.

Hsu K, Gupta H, Sorooshian S (1995). "ANNfor rainfall-runoff process", Water Resources, 31(10), pp.2517–2530.

Jain, A. and Srinivasulu, S., (2004). "Development of effective, efficientrainfall-runoff models using integration of deterministic, real-coded GA and ANN", Water Resour. 40(4):23-45.

Jang JS (1993). "Adaptive fuzzy inference systems". IEEE Trans. Syst. Man and Cybern. 23:665-685.

Karunanithi N, Grenney W, Whitley D, Bovee K (1994). "Neural network for river prediction", J. Comput. Civil Eng. 8(2):201-220.

Kennedy J, Eberhart R, Shi Y (2001). "Swarm intelligence". ISBN: 1-974-3564, San Francisco: Kaufmann, pp. 35-87.

Khondker MH, Wilson G, Klinting A (1998). "Application of neural networks in real time flash flood forecasting", In Proceed. Hydroinform. Conf. pp. 777-781.

Kitanidis P, Bras R (1980). "Real-time forecasting with a conceptual hydrologic model, 2, applications and results", Water Resour. 16(6):1034-1044.

Lindstrom G, Johansson B, Persson M, Gardelin M, Bergstrom S (1997). "Development and test of distributed HBV-96 hydrological model, J. Hydrol. 201:272-288.

Mandic D, Chambers J (2001). "Recurrent Neural Networks for Prediction: Learning Algorithms, Architectures and Stability", Wiley & Sons: New York pp. 56-90.

Minns AW (1998). "Artificial neural networks as sub-symbolic process descriptors", published Ph.D. Thesis, A. Balkema, Rotterdam, The Netherlands.

Nash JE, Sutcliffe JV (1970). "River flow forecasting through conceptual models; part I – a discussion of principles", J. Hydrol. Sci. 10:282-290.

Nishimura S, Kojiri T (1996). Real-time rainfall prediction using neural network and genetic algorithm with weather radar data, In Proceeding of the 10th. Congress of Asia and Pacific Div. Int. Assoc. Hydraulic Res. 204-211.

Ojugo A (2009). "Artificial neural networks gravitational search algorithm for rainfall runoff modeling in hydrology", unpublished PhD Thesis, Computer Science Department: Ebonyi State University Abakiliki, Nigeria.

Ojugo A, Eboka A, Okonta E, Yoro R, Aghware F (2012). "GA rule-based intrusion detection system", J. Comput. Inform. Syst. 3(8):1182-1194.

Ojugo AA, Yoro R (2013a). "Computational intelligence in stochastic solution for Toroidal Queen problem", Progress Intell. Comput. Appl. 2(1),doi: 10.4156/pica. 2(14):46-56.

Ojugo AA, Emudianughe J, Yoro RE, Okonta EO, Eboka A (2013b). "Hybrid artificial neural network gravitational search algorithm for rainfall runoff modeling in Hydrology", Progress Intell. Comput. Appl. 2(1),doi: 10.4156/pica. 2(1.2):22-33.

Openshaw W (2010). "Rainfall runoff processes: Workbook for online module", [online]:www.engineering.usu.edu/dtarb/rrp.html, last accessed Feb 2013.

Oreskes N, Shrader-Frechette K, Belitz K (1984). "Verification, validation, and confirmation of numerical models in the earth *sciences*", Earth Sci. 263:641-646.

Perez M, Marwala, T (2011). “Stochastic optimization approaches for solving Sudoku”, Trans. Evol. Comput. (IEEE) pp. 256-279.

Rajurkar MP, Kothyari U, Chaube U (2004). “Modeling of daily rainfall-runoff relationship with artificial neural network”, J. Hydrol. 28:96-113.

Reggiani P, Rientjes T (2005). “Flux parameterization in the representative elementary watershed approach: Application to basin”, Water Resour. 41(4):18-27.

Rientjes T (2004). “Inverse modeling of the rainfall-runoff relation: a multi objective model calibration approach”, published Ph.D thesis, Delft University of Technology, Delft, Netherlands.

Reynolds R (1994). “An introduction to cultural algorithms”, Trans. Evol. Programm. (IEEE), pp. 131-139.

Sajikumar N, Thandaveswara BS (1999). “A non-linear rainfall runoff model using an artificial neural network”, J. Hydrol. 216:32-55.

Sedighizadeh D, Masehian E (2009). “Particle Swarm Optimization: Taxonomy, methods and application”, J. Comput. Theor. Eng. ISSN: 1793-8201, 1(5):127-145.

Seibert J (1999).“Conceptual runoff models - fiction or representation of reality?”Acta University publications, comprehensive summaries of Uppsala Dissertations from the Faculty of Science and Technology 436, Uppsala. ISBN 91-554-4402-4.

Seibert J (2000). “Multi-criteria calibration of conceptual model”, J. Hydrol. Earth Syst. Sci. 4:215-224, SRef-ID: 1607-7938/hess/2000-4-215.

Shamseldin AY (1997). “Application of a neural network to rainfall-runoff modeling”, J. Hydrol. 199:272-294.

Tarboton DG (2003).“Rainfall runoff processes: A workbook for online module, [online]: www.engineering.usu.edu/dtarb/rrp.html, last retrieved: 16-8-2012.

Tokar AS, Johnson PA (1999). “Rainfall-runoff modeling using neural networks”, J. Hydrol. Eng. 4(3):232-239.

Ursem R, Krink T, Jensen M, Michalewicz Z (2002). “Analysis and modeling of controls in dynamic systems”. IEEE Trans. Memetic Syst. Evol. Comput. 6(4):378-389.

Varoonchotikul P (2003). “Flood forecasting using artificial neural networks”, published Ph.D. thesis, Swets&Zeitlinger, Lisse, The Netherlands.

http://sciencewebpublishing.net/aser