Dynamical Systems Prediction using Temporal Artificial Neural Networks and Multi-objective Genetic Algorithm

Praveen Koduru[1], William H. Hsu[1], Sanjoy Das[1], Stephen Welch[2] and Judith L. Roe[3]

Departments of Electrical Engineering and Computer Engineering and Computer Science
[2]Department of Agronomy [3]Division of Biology
Kansas State University
Manhattan, KS 66506

Abstract

We investigate the problem of learning to predict dynamical systems that exhibit switching behavior as a function of exogenous variables. The family of dynamical systems we present is significant to the modeling of gene expression and organismal response to environmental conditions and change. We first develop a framework for learning to predict events such as state or phase changes as a function of multiple dynamic variables. Next, we consider the more challenging problem of identifying parameters and the functional form of the dynamical systems *ab initio*. We then survey several applicable representations and inductive learning techniques for each task. We then describe a comparative experiment in learning a particular instantiation of the dynamical system for a plant genome modeling application. Finally, we evaluate the results using predictive accuracy of the differential equation parameters or accuracy on the event prediction task; consider the ramifications for modeling the metabolic processes of living systems; and outline future challenges such as multi-objective optimization and finding relevant exogenous and latent variables.

Introduction

Predicting the behavior of dynamical systems as a function of exogenous variables is an often-encountered problem in learning simulation models of living organisms. Gershenfeld and Weigend identify specific aspects of the problem of understanding historical time series data: *forecasting*, the inferential problem of predicting future values of some series given past values of the same or a different series; *modeling*, the problem of inductively learning to map from observations of one series to estimated values of another; and *characterization*, a term used to describe change of representation, meta-learning (inductive bias optimization) and high-level model selection. [GW94] Many learning approaches consider each of these subproblems as independent phases. In this paper, we consider the whole problem of capturing the behavior of a dynamical system, from identifying the functional form of the system, to estimation of low-level parameters in a system with known functional form, to predicting discrete events such as switching.

We consider a type of dynamical system that exhibits switching behavior and can thereby represent specific gene regulatory functions such as control of development in plants. For purposes of developing our algorithm and experiments, we use a system of differential equations whose form is not supplied to the learning mechanism. The equational model is based upon our preliminary study of real gene expression data and provides a way to generate synthetic data that can be validated using real data from the field and laboratory.

In this work, we focus on the general problem of predicting events in a time series with exogenous variables. The novel aspect of our approach is that we isolate aspects of the characterization and modeling tasks that provide a natural problem definition for symbolic regression. We also develop a general framework for learning models of dynamical systems, including the functional form and parameters of the predictor. We apply two methods of learning: first, predicting events directly from a partially observable multivariate stochastic process; and second, estimating parameters of a function and substituting these into an equational representation of the dynamical system. We discuss cases where the latter task involves feature selection and construction. Our general framework presents at least three design choices important to for learning to predict discrete behavior from time series – namely, whether the solution must:

1. be expressed using a **specific functional form**, whether learned or given as input
2. apply **single or multi-objective** optimization
3. use a function whose **structure is learned or pre-specified** except for parametric values

Our experimental design rationale is that these three design choices, if made in order, select from a flexible set of concrete representations and learning algorithms that allow systematic exploration. The results indicate that isolating the parameter estimation and prediction tasks can boost localization accuracy, and isolating the functional form identification task can provide an independent evaluation mechanism without loss of accuracy. We discuss the relevance to predictive modeling of dynamical systems and specification of learning algorithms. Future work includes generalization of this method beyond dynamical systems and learning from time series.

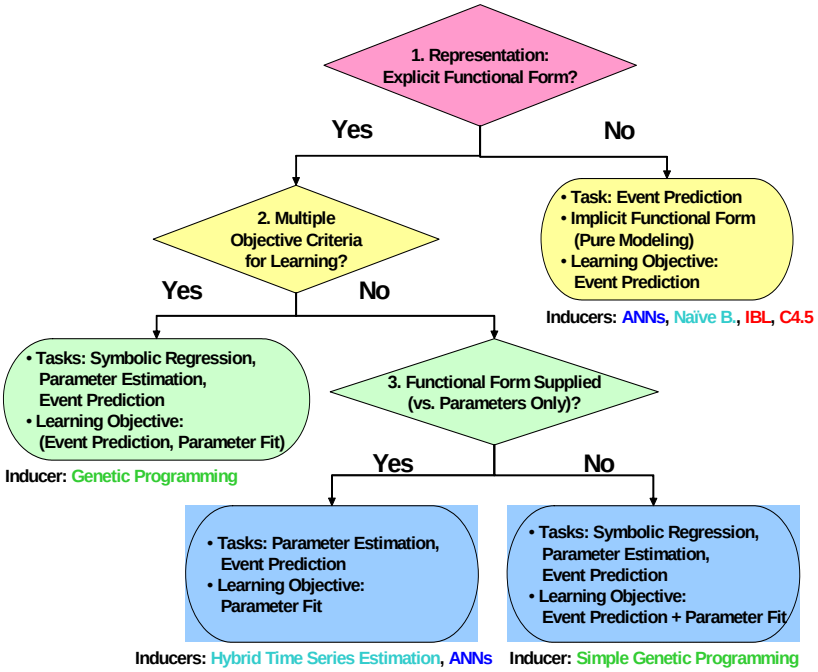

Figure 1. Overview of the learning framework: design choices, learning tasks and objective functions, and typical inducers.

Background

Figure 1 depicts a decision tree used to identify learning problems in our learning framework. Each internal decision nodes represents one of the three design choices listed in the introduction; each leaf, a functional specification of the learning task and the objective criterion used. Except for differences in the representation of hypotheses and the training algorithms, this framework applies equally to temporal and atemporal domains. In this paper, the **overall** performance element in our application is to predict the timing of discrete events, such as switching or other state and phase transitions. This is true regardless of the solution representation and the learning task(s) listed.

In this section we first define the dynamical systems modeling problem, then review representations, objective measures, and search algorithms from the literature. Taken together, these form the specific inducers listed below the leaves of the decision tree.

Implicit Representations

The first choice in Figure 1 determines whether the problem representation requires an explicit functional form, either supplied by domain experts or determined automatically using methods such as symbolic regression. On the "no" branch, therefore, are those inducers that produce implicit representations from which we are not necessarily able to extract a solution in the form of an algebraic expression or the parameters of ODEs. The isolation of the structural and parameter estimation phases makes it easier to apply techniques for model selection, or to implement hybrid and multistrategy time series learning techniques [HRW00], as is the case for our application.

To implement the far right-hand leaf of the tree in Figure 1, we use six basic inducers: multilayer perceptrons, RBFs, instance-based learning (IBL), *OneR*, Naïve Bayes, and decision tree learning. [FHT+04]

Explicit Representation: Equational Model

The "yes" (left) branch of the first decision node Figure 1 commits the designer of the learning system to provide the parameters of an equational model. If the functional form (e.g., polynomial of a given degree, ODE of a given order) is not previously known, the designer must also develop or apply methods that discover this form.

To implement parameter estimation, we use nonlinear time series predictors with the specific variables as values. In the experiments reported here, all of the back-end parameter estimators are recurrent ANNs. The front-end integrators may be ANNs or any of the base inducers listed in the previous subsection.

Objective Functions

The second decision node commits the designer to use either a simple measure of fit such as prediction accuracy for the parametric model or a composite (e.g., weighted) one that combines accuracy on the continuous and discrete prediction tasks. The continuous prediction task consists of predicting the value of the ODE variables, either by

finding an implicit representation (as in the rightmost leaf), or by finding an function approximator by symbolic regression (as in the bottom right hand and far left hand leaves) and testing quality of fit over fitness cases. We note that fitness cases can be pre-evaluated values of the ODE variables or generated dynamically. Dynamic values facilitate multi-objective optimization (by simultaneous substitution into the ODE and) active learning, which can be useful in domains that involve continuous time [NK03].

Learning the Functional Form

The third decision node asks whether the specified functional form is previously known or will actually be learned. In many cases, the form is supplied by a domain expert or some hyper parameters are known. For example, the order of the ODE system and the variables in the system might be determined by physical observation or experimentation. For purposes of our application, we consider the initial condition specification to be part of the training experience (i.e., input attributes) and not the functional form.

A simple genetic programming (GP) system [Ko92] can find not only the coefficients of the dynamical system but an algebraic approximator. Several Pareto optimization algorithms can be used to find non-dominated solutions given multiple simultaneous objective functions of the kind listed above. *FSGA* [KDWR04] is one such algorithm.

Parameter Estimation: Hybrid Approaches

Finally, the bottom left leaf requires parameter estimation given a **specified** functional form. In our application this means that the variables of the ODE are given and the problem consists of predicting their values given **only** the time and exogenous variables. We use a hybrid, modular approach consisting of the temporal ANN models as the predictor representation and the other inductive learning representations as the combiner for their outputs.

Methodology

We now present a synopsis of our dynamical systems modeling domain and the new application of a hybrid fuzzy simplex genetic algorithm (FSGA).

$$\frac{d}{dt} LFY = g(SOC1, TFL1) - LFY$$

$$\frac{d}{dt} AP1 = h_{\uparrow}(LFY) - AP1$$

$$\frac{d}{dt} TFL1 = h_{\downarrow}(AP1) - TFL1$$

Dynamical system to be modeled.

Application: Biological Dynamical System

We now present a synopsis of our dynamical systems modeling domain and the new application of a hybrid fuzzy simplex genetic algorithm (FSGA).

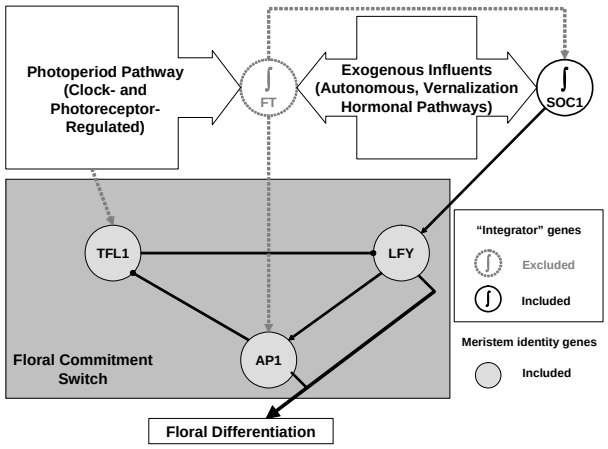

Figure 2. A bistable switch that captures floral development gene dynamics in many flowering plants.

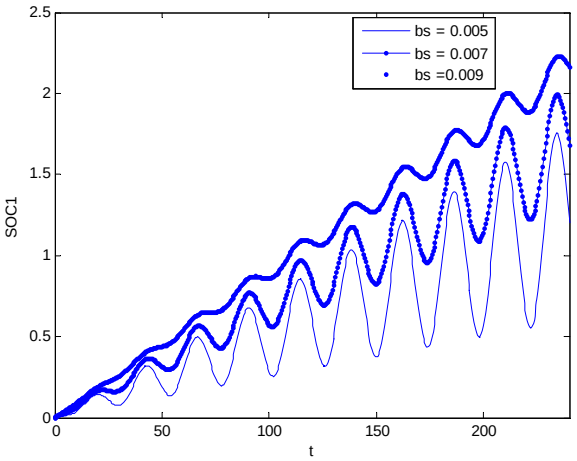

Figure 3. The exogenous variable with varying b_S
SOC1 **$(a_S, b_S, p_S, t) = b_S \cdot t + ½ (a_S - b_S) \cdot t \cdot \sin(2\pi\, t / p_S)$**

The Fuzzy Simplex Genetic Algorithm

Fuzzy dominance: Assume an overall minimization problem involving M objective functions $e_i(.)$, $i = 1 \ldots M$. The solution space is denoted as $\Psi \subset \Re^n$. Given a monotonically non-decreasing function $\mu_i^{dom}(\cdot)$, whose range is in [0, 1], $i \in \{1,2,\ldots,n\}$ such that $\mu_i^{dom}(0) = 0$, solution $u \in \Psi$ is said to i-dominate solution $v \in \Psi$, if and only if $e_i(u) < e_i(v)$. This relationship can be denoted as $u \succ_i^F v$. If $u \succ_i^F v$, the degree of fuzzy i-dominance is equal to $\mu_i^{dom}(e_i(v) - e_i(u)) \equiv \mu_i^{dom}(u \succ_i^F v)$. Fuzzy dominance can

be regarded as a fuzzy relationship $u \succ_i^F v$ between u and v. Solution $u \in \Psi$ is said to fuzzy dominate $v \in \Psi$ if and only if $\forall i \in \{1,2,\ldots,M\}$, $u \succ_i^F v$. This relationship can be denoted as $u \succ^F v$. The degree of fuzzy dominance can be defined by invoking the concept of fuzzy intersection and using a *t*-norm,

$$\mu^{dom}\left(u \succ^F v\right) = \bigcap_{i=1}^{M} \mu_i^{dom}(u \succ_i^F v) \qquad (1)$$

Given a population of solutions $S \subset \Psi$, a solution $v \in S$ is said to be fuzzy dominated in S iff it is fuzzy dominated by any other solution $u \in S$. In this case, the degree of fuzzy dominance can be computed by performing a union operation over every possible $\mu^{dom}\left(u \succ^F v\right)$, carried out using *t*-co norms as,

$$\mu^{dom}(S \succ^F v) = \bigcup_{u \in S} \mu^{dom}(u \succ^F v) \qquad (2)$$

In this manner, each solution can be assigned a single measure to reflect the amount it dominates others in a population. Non-dominated individuals within a solution will be assigned zero fuzzy dominance, as for any non-dominated individual $\mu_i^{dom}(u \succ_i^F v)$.

Further details of this concept can be found in [KDWR04]. In the present work, the membership $\mu_i^{dom}(u \succ_i^F v)$ is piecewise linear, and given by,

$$\mu_i^{dom}\left(\Delta e_i\right) = \begin{cases} 0, & \Delta e_i \le 0 \\ (\Delta e_i)/\Delta_i & 0 < \Delta e_i < \Delta_i \\ 1, & \Delta e_i \ge \Delta_i \end{cases} \qquad (3)$$

where, $\Delta e_i = e_i(v) - e_i(u)$. The union and intersection operators follow the standard min and max definitions [Me95].

Fuzzy dominance time makes it possible to assign a single measure of fitness to multiple individuals. Computing the mutual fuzzy dominance in a population of individuals enables local gradient descent based techniques to be applied in a multi-objective framework. It applies a local search, the Nelder-Mead simplex algorithm, in conjunction with a multi-objective genetic algorithm.

A simplex in n-dimensions consists of $n+1$ solutions u_k, $k = \{1,2,\ldots n+1\}$ which are its vertices. In a plane, this corresponds to a triangle. The solutions are evaluated in each step and the worst solution w is identified. The centroid of the simplex is then evaluated, excluding the worst solution and the worst point is reflected along the centroid. If

$$c = \sum_k u_k - w$$

is the centroid, the reflected solution is:

$$r = c + (c - w) \qquad (4)$$

Usually, the worst point w is replaced with the reflected point r in the simplex, but if the r is better than any solution in the simplex, the simplex is further expanded as,

$$r_e = c + \eta(c - w) \qquad (5)$$

Where η is called the expansion coefficient. However, if the reflected solution r is worse than w, the simplex is contracted and the reflected solution is placed on the same side of the centroid. When solution r is not worse than w, but worse than any other solution in the simplex, the simplex is still contracted, but the reflection is allowed to remain on the other side of the simplex's centroid. Reflection is carried out as follows,

$$r_c = c \pm \kappa(c - w) \qquad (6)$$

In the above equation, κ is called the contraction coefficient. Solution w is replaced with the new one, r, r_e, or r_c in the next step. The simplex algorithm is allowed to run for multiple steps before it converges. Fuzzy dominance is the objective function to which the Nelder-Mead simplex algorithm is applied. This allows the simplex algorithm to push solutions towards regions of lower dominance, i.e. the Pareto front.

One of the key features of current multi-objective algorithms is *elitism*, i.e. the reinsertion of the best individual solutions into the parents [DAPM02, ZT03]. This paper removes that shortcoming by maintaining an archive to store some of the non-dominated solutions found, i.e. the *elites*. Within each generation of the genetic algorithm, the archive is combined with the population, before selection, crossover and mutation are carried out.

Unlike in the previous algorithm [KDWR04], where a binary tournament selection was implemented, the present approach uses a rank based strategy. The solutions in a population are sorted according to fuzzy dominances and then assigned ranks. A Roulette wheel selection is done to pick the parents for crossover. An offspring t, was computed from two parents u and v in the following manner,

$$t = \zeta u + (1 - \zeta)v \qquad (7)$$

where ζ is a uniformly distributed random number in [0, 1].

Solutions were mutated with a probability of β, by adding a random number with zero mean, that followed a Gaussian distribution with a spread σ, according to,

$$u = u + N(0, \sigma) \qquad (8)$$

At the end of the mutation, the best $2(n+1)$ solutions found, in terms of the fuzzy dominance measure, are further improved by applying fixed number of iterations of the Nelder-Mead algorithm.

At the end of the iteration, the archive is updated by extracting the solutions from the offspring population which

are not dominated by the archive individuals and inserting them into the archive. Any archive individual that is now dominated is removed from it. Some more individuals are discarded also in order to limit the archive size to a tractable number of individuals and reduce the computational effort in comparing them with the population in each generation. This is accomplished by means of a hyper grid [ZT03], a feature that has been added to the present version of the algorithm and which did not exist in [KDWR04]. The hyper grid divides the solution space into different hypercube shaped regions. Only a single individual is picked at random from each cell to be included in the archive, while the rest are deleted.

Applying the hyper grid to the present algorithm has another added benefit. Although the algorithm in [KDWR04] was able to converge rapidly towards the Pareto front of the solution space, the spread of the final front obtained was highly localized. Much of the effort in current multi-objective optimization is targeted towards finding solutions that are not only close to the Pareto front, but also that sample the front at approximately regular intervals. By maintaining a single solution from each hypercube cell in the archive, the algorithm now guarantees sufficient diversity along the Pareto front.

Experimental Design

The desired output is the switching indicator function (floral commitment has or has not occurred).

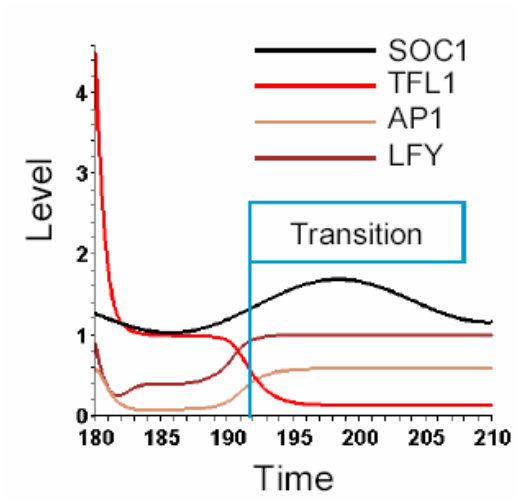

Figure 4. The target event: floral commitment.

As discussed in the previous section, we use a variety of inductive learning representations as implicit functional forms, and use either temporal ANNs (as parametric nonlinear models) or functions evolved using GP.

For fixed functional forms, we use a simple recurrent ANN with leaky integrator axons and 8 hidden units.

The desired output is a multivariate regression target consisting of the values of *LFY*, *TFL*, and *AP1* over 244 time quanta (122 observed hours over a period of about 9 days). The time *t* and *SOC1* are inputs to the predictor, whose output is used in our hybrid time series estimation scheme (shown in bottom right leaf of Figure 1).

In FSGA there are 9 parameters to be identified in system that are evolved over 25,000 function evaluations. Selection was done using rank based selection. The mutation rate is set at 0.1 and the crossover rate at 0.8. At each generation one simplex is formed and it is allowed to flip for 5 iterations. For the simplex the variables are set as α =10, η =1.5 and κ =.5. The parameters were evolved using synthetic data obtained using b_S=0.007 and then used to extrapolate over b_S=0.005 and 0.009.

Results

Table 1. Root mean squared error (RMSE) for Control, Ideal, Simple Recurrent (Artificial Neural) Network, and Fuzzy Simplex Genetic Algorithm

Inducer	Experiment			
	Control	Ideal	SRN	FSGA
IBL	0.0905	0.1109	0.1109	0.1109
C4.5	0.0907	0.0640	0.0640	0.0640
N.Bayes	0.1779	0.0877	0.0641	0.0640
RBF	0.0852	0.0998	0.0894	0.0905
OneR	0.0905	0.0640	0.0640	0.0640
MLP	0.0829	0.0864	0.0716	0.0874

Table 1 lists comparative results for the basic inducers from the bottom right leaf of Figure 1, applied to only *t* and *SOC1*. This establishes an experimental control or baseline. The "informed" event predictor is given the values of *t*, *SOC1*, and *LFY*, *AP1*, and *TFL1* as input. Not all of the basic inducers improve over the control on this straightforward prediction task, for which the concept *TFL1(t+1) < AP1(t+1)* is sufficient. However, all inducers achieve between 1 and 3 time units of localization error on the switching task, or 30-90 minutes.

We then used simple recurrent ANNs together with the base inducers to implement the method depicted in the bottom left leaf of Figure 1 – that is, given *SOC1* as input, predict the values of *LFY*, *AP1*, and *TFL1* and use the **predicted** values and *SOC1* as input to each of the six inductive learning algorithms listed. The results are shown in the last two columns of Table 1. Again, all inducers in the last two columns predict floral commitment to within 1-3 time units of accuracy, well below the (2-hour) threshold of real-world observation. As the Ideal and FSGA columns show, using FSGA to predict the gene expression levels as intermediate values results in performance identical to the ideal case where *LFY*, *AP1*, and *TFL1* are **given** for nearest neighbor (IBL), decision tree learning (C4.5), and *a priori* rule induction (OneR), and near-ideal performance in the case of feedforward artificial neural networks trained using back propagation (MLP).

Table 2. RMSE for three FSGA parameter estimation experiments using bS = 0.007.

Inducer	bS parameter value input to first stage		
	0.005	**0.007**	**0.009**
IBL	0.0640	0.1109	0.0905
C4.5	0.0640	0.0640	0.0640
N.Bayes	0.0905	0.0640	0.0905
RBF	0.0640	0.0905	0.1040
OneR	0.0640	0.0640	0.0905
MLP	0.0476	0.0874	0.0669

Finally, we wished to assess the generalization quality of the FSGA-based parameter estimation system. Table 2 shows the results obtained by training using the *SOC1* curve with the parameter b_S = 0.007 and then using b_S = 0.005, 0.007, and 0.009 as input to the resultant dynamical system.

$$SOC1\ (a_S, b_S, p_S, t) = b_S \cdot t + \tfrac{1}{2} (a_S - b_S) \cdot t \cdot \sin(2\pi\, t / p_S)$$

Conclusions and Future Work

We have developed a framework for learning parameters and, when required, the functional form of a dynamical.system from time-dependent data, and applied it with some success to predict a target event to within 30-90 minutes out of over 120 hours. This framework uses hybrid learning and prediction of ODE variables to achieve an improvement over basic inducers.

Continuing work focuses on: developing a robust multi-objective specification for the dynamical system and events of interest; statistical validation of extracted features in symbolic regression; multi-objective GEC; latent variable discovery; and exogenous variable selection.

References

[Do03] Dong, Z. 2003. *Incorporation of genomic information into the simulation of flowering time in Arabidopsis thaliana*. Ph.D. dissertation, Kansas State University.

[FHT+04] Frank, E., Hall, M., Trigg, L., Kirkby, R., Schmidberger, G., Ware, M., Xu, X., Bouckaert, R., Wang, Y., Inglis, S., and Witten, I. H. 2004. *Waikato Environment for Knowledge Analysis (WEKA) v.3: Machine Learning Software in Java.* Distributed from URL: http://www.cs.waikato.ac.nz/~ml/weka/.

[GW94] Weigend, A. S. and Gershenfeld, N. A 1994. *Time Series Prediction: Forecasting the Future and Understanding the Past.* Santa Fe Institute Studies in the Sciences of Complexity XV; Proceedings of the NATO Advanced Research Workshop on Comparative Time Series Analysis (Santa Fe, May 1992). Reading, MA: Addison-Wesley.

[HRW00] Hsu, W. H., Ray, S. R., and Wilkins, D. C. 2000. A multistrategy approach to classifier learning from time series. *Machine Learning*, **38**(1-2):213-236.

[KDWR04] P. Koduru, S. Das, S. M. Welch, J. L. Roe, "Fuzzy Dominance Based Multi-objective GA-Simplex Hybrid Algorithms Applied to Gene Network Models", Lecture Notes in Computer Science: Proceedings of the Genetic and Evolutionary Computing Conference, Seattle, Washington, (2004).

[LPB+04] Luke, S., Panait, L., Balan, G., Skolicki, Z., Bassett, J., Hubley, R., and Chircop, A. 2004. ECJ 12: A Java-based Evolutionary Computation and Genetic Programming Research System. Distributed from URL:

[ZT03] Zitzler, E., Thiele, L., Laumanns, M., Fonseca, C.M., da Fonseca, V.G.: Performance assessment of multiobjective optimizers: An analysis and review, IEEE Transactions on Evolutionary Computation, Vol. 7, no. 2 (Apr. 2003) 117-132.

http://cs.gmu.edu/~eclab/projects/ecj/.

[DAPM02] Deb, K., Agrawal, S., Pratap, A. and Meyarivan, T.: A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Transactions on Evolutionary Computation 6(2): 182-197 (2002).

[Me95] Mendel, J.M.: Fuzzy logic systems for engineering: A tutorial, Proceedings of the IEEE, Vol 83, No. 3 (March 1995) 345-377.

[Mu02] Murphy, K. J. 2002 *Dynamic Bayesian Networks: Representation, Inference and Learning.* Ph.D. dissertation, University of California – Berkeley.

[NK03] Nodelman, U. and Koller, D. 2003. Continuous Time Bayesian Networks. In *Proceedings of the International Conference on Uncertainty in Artificial Intelligence (UAI-2003)*, Acapulco, MEXICO, August, 2003. San Mateo, CA: Morgan-Kaufmann.

[Wa60] Wang, J.Y. 1960. A critique of the heat unit approach to plant response studies. *Ecology*, **41**:785-90.