33rd Annual International Conference of the IEEE EMBS
Boston, Massachusetts USA, August 30 - September 3, 2011

Ordering Samples along Environmental Gradients using Particle Swarm Optimization

Steven Essinger, Robi Polikar, Gail Rosen

***Abstract*—Due to the enormity of the solution space for sequential ordering problems, non-exhaustive heuristic techniques have been the focus of many research efforts, particularly in the field of operations research. In this paper, we outline an ecologically motivated problem in which environmental samples have been obtained along a gradient (e.g. pH), with which we desire to recover the sample order. Not only do we model the problem for the benefit of an optimization approach, we also incorporate hybrid particle swarm techniques to address the problem. The described method is implemented on a real dataset from which 22 biological samples were obtained along a pH gradient. We show that we are able to approach the optimal permutation of samples by evaluating only approximately 5000 solutions - infinitesimally smaller than the 22! possible solutions.**

***Index Terms*—Particle Swarm Optimization, Environmental Gradients, Sequential Ordering**

I. Introduction

One of the most notable Sequential Ordering Problems (SOP) in optimization theory is the canonical Traveling Salesman Problem (TSP) [1]. In the most general case, a figurative salesman must travel between cities ensuring that he visits all the cities on his list only once and does so by traversing the shortest distance. Therefore, the goal of the optimization is to complete his itinerary by listing the cities in order for which he must visit. There are numerous papers dedicated to the topic, correspondingly, we have referenced several of them [2], [3], [4], [5], [6]. This paper focuses on a biologically motivated SOP.

The specific SOP addressed within this paper focuses on ordering samples that were obtained along an environmental gradient. We note upfront that in the field of community ecology, the term *sample* has diverged from its usage in signal processing or statistics, and refers to an observation. Environmental gradients are defined as a spatially varying aspect of the environment that is expected to influence the composition of species present [7]. Examples of these gradients include pH, temperature, salinity, moisture, etc. There are generally only a few gradients of 'large effect' present at a time and location that are affecting the respective species abundances in the environment. For example, it has been shown that salinity strongly influences the distribution of bacteria in an environment [8]. There are innumerable environments in which an ecological study may focus upon; examples include soil [9], sea [10], human [11], [12] and ant guts [13]. Given a bundle of samples obtained from an ecological study, can we provide the ordering of samples correlated with an unknown environmental gradient present?

II. Background

Investigations of environmental gradients are a component of the broader class of study known as ordination in the field of community ecology [14]. The term *ordination* derives from attempts to order a group of objects in any number of dimensions, preferably few, that approximates some pattern of response of the set of objects. The usual objective of ordination is to help generate hypotheses about the relationship between the species composition at a site and the underlying environmental gradients [15]. In our study we are concerned with unconstrained gradient analysis; gradients are unknown *a priori* [7]. In other words, the species tell us what the gradient are. Traditionally, this is performed using an ordination technique such as Detrended Correspondence Analysis [16]. The sample locations are recorded along the unknown gradient prior to analysis, from which the shape and scale of the gradients are inferred. The problem focuses on analysis when the sample locations are not recorded and we wish to recover the correct order.

Bacteria are identified at each sample by using the universal bacterial barcode, 16S rRNA gene [17]. These genes are sequenced for each sample using DNA sequencing technology such as pyrosequencing [18]. Sequences are classified using readily available online tools such as the RDP classifier [19]. Once the sequences are classified we are able to identify the species present in each sample and their respective abundances. Once arranged in a matrix, we are able to process the samples using the proposed algorithm.

Particle swarm optimization (PSO) methods have been in development since the mid-1990s [20]. Since their inception it has been a common practice to combine PSO methods with other techniques forming a hybrid approach. For example, a hybrid algorithm coupling ant colony optimization with local search procedures was developed to address problems in sequential ordering [6]. A very recent paper described an algorithm developed for the traveling salesman problem, combining PSO with a local search method known as SOP k-Exchange [5]. Motivated by the potential fit of this algorithm to our environmental sampling problem, we have sought to appropriately model our problem for this framework. Herein, we describe our model and the set of modified procedures to address our environmental sampling dilemma.

III. Methods

A. Model Setup

We begin with a vector array of unordered samples obtained along an environmental gradient,

$$\mathbf{x} = \{x_1, x_2, ..., x_n\},$$

in which, $\{x_i \, \epsilon \, \mathbb{Z} : (1, n)\}$ and $x_i \neq x_j \, \forall \, i \neq j$,

where $\mathbb{Z} : (1, n)$ are all discrete integer values over range 1 to n.

Department of Electrical & Computer Engineering
Drexel University, Philadelphia, PA, 19104, USA
Correspondence: sessinger@drexel.edu

This work was supported by the National Science Foundation 708 CAREER award #0845827 and DOE award DE-SC0004335.

Each one of these samples contain the abundance of each species found at the sample site,

$$\mathbf{sp}_i = \{sp_{i1}, sp_{i2}, ..., sp_{im}\},\ i = 1:n.$$

Since we are sampling the bacteria present at each sample, and there is incredible diversity within the bacterial community, we generally have $m >> n$, so we form a 'tall' *m* x *n* data matrix which will be input to our algorithm.

In order to determine if our environmental samples are correctly ordered, we rely on biological assumptions of the bacterial responses to the underlying gradient. It is assumed that this gradient is affecting the populations of bacteria [7]. Particularly, we assume that each type of bacteria prefers a particular environmental condition [21]. As this condition varies spatially across the gradient, so will the abundance of each respective bacteria. For example, biologist use a unimodel distribution to model the abundance of bacteria across a pH gradient [22]. In this particular study, there is an immense amount of diversity between bacteria, having pH preferences ranging from of 4 to 8 [9].

To assess the feasibility of a permutation of ordered samples to reflect the true ordering we separately observe the distributions of all species. To this end we have chosen the Wald-Wolfowitz runs test, $w_i(\mathbf{x})$ [23]. For a given permutation, we test the null hypothesis that the abundance of each species is randomly distributed across the samples. When the null hypothesis cannot be rejected for most species we infer that the samples are incorrectly ordered. However, when the null hypothesis is rejected for many species we infer that the samples are correctly ordered. This is in accordance with the biologists' selection of the gaussian model of species along a gradient [21]. Even in cases where there is dispute among biologists interchanging the gaussian model for other distributions [24], our model is still effective since we are testing for randomness versus any distribution.

Formally, we set up our objective/fitness function as follows,

$$y = \textstyle\sum_{i=1}^{M} w_i(\mathbf{x}),\ \{y\,\epsilon\,\mathbb{Z} : (0,m)\} \text{ and } \{w_i\,\epsilon\,\mathbb{Z} : (0,1)\}$$

in which our goal is to chose the permutation such that,

$$\mathbf{x}_{correct} = \max_{\mathbf{x}}(y).$$

B. Algorithm Implementation

PSO algorithms were inspired by the behavior of insects in a population [20]. We model each insect as a particle representing one solution to our optimization problem. The collection of particles, referred to as the swarm, exist in the solution space. Each particle has a position and a velocity. With each iteration of the PSO algorithm, a particle moves in the solution space based on its performance on the objective/fitness function and that of the entire swarm. The benefit of this behavior is that in many cases the optimization avoids local minima. The optimization terminates once the maximum number of iterations has been reached or the score of the fitness function passes a predetermined threshold.

Hybrid PSO Algorithm Pseudocode

Input:
- Data matrix of species abundance: $[(\#\ of\ Species)\ x\ (\#\ of\ Samples)]$
- Fitness test: Wald-Wolfowitz
- m: # of particles for PSO
- k: # of iterations
- Free-parameters: {w,c_1,c_2}

Initialization:
1. Input seed, $\mathbf{x}_s$: random permutation of integers 1:n
2. Compute initial particle velocities, $\mathbf{v}_i^0$:
   a. Choose # of IM{} for each particle's initial velocity:
      - Drawn from: $U[n/4, n/2]$
   b. Select the node and displacement for each move:
      - Node from $U[1, n]$
      - Displacement from: $U[-n/3, n/3]$
3. Compute initial particle positions: $\mathbf{x}_i^0 = \mathbf{x}_s + \mathbf{v}_i^0$, i = 1:m
4. Set each particle best to initial: $\mathbf{p}_i = \mathbf{x}_i^0$
5. Run fitness test on all $\mathbf{x}_i^0$
6. Set global best position: $\mathbf{g} = \mathbf{x}_i^0$ s.t. $\max_{\mathbf{x}_i^0}(fitness)$

Iteration:

For iter = 1:k

For particle$_i$ = 1:m

A. Compute velocities: $\mathbf{v}_i^{k+1}$:
   1. $w\mathbf{v}_i^k$

   Compute r_1, r_2, then:
   2. $c_1 r_1(\mathbf{p}_i - \mathbf{x}_i^k)$ distance of current to particles best
   3. $c_2 r_2(\mathbf{g} - \mathbf{x}_i^k)$ distance of current to global best

B. Add velocities sequentially to $\mathbf{x}_i^k$: $\mathbf{x}_i^{k+1} = \mathbf{x}_i^k + \mathbf{v}_i^{k+1}$

C. Compute fitness of each $\mathbf{x}_i^{k+1}$:
   if $fit(\mathbf{x}_i^{k+1}) > \mathbf{p}_i$, then $\mathbf{p}_i = \mathbf{x}_i^{k+1}$
   if $fit(\mathbf{x}_i^{k+1}) > \mathbf{g}$, then $\mathbf{g} = \mathbf{x}_i^{k+1}$

D. Perform Local Enhancement on $\mathbf{g}$ using SOP-3 Exchange

E. Update $\mathbf{g}$

End

End

Definitions:
- $\mathbf{x}_s$: position (permutation) seed for all particles
- $\mathbf{x}_i^k$: position for particle i at iter k
- $\mathbf{v}_i^k$: velocity for particle i at iter k
- $\mathbf{p}_i$: best position (permutation) for particle i
- $\mathbf{g}$: best position (permutation) of all particles
- r_1, r_2: random values from U[0,1]
- Free parameters:
  - w: current particle
  - c_1: particle best
  - c_2: global best

Fig. 1. Pseudocode of the proposed hybrid PSO algorithm

The traditional formulation of a PSO algorithm begins with velocity, *(please refer to Figure 1 for notation definitions)*

$$\mathbf{v}_i^{k+1} = w\mathbf{v}_i^k + c_1 r_1(\mathbf{p}_i - \mathbf{x}_i^k) + c_2 r_2(\mathbf{g} - \mathbf{x}_i^k)$$

and position,

$$\mathbf{x}_i^{k+1} = \mathbf{x}_i^k + \mathbf{v}_i^{k+1}.$$

Since the particles are each a permutation of unique integers, the velocities are defined as insertion moves, IM. Each IM(i,d), moves the value i in the particle by a displacement, d, either left or right. Each of the three components of the velocity are computed separately and added sequentially to the current particle position in equation. Greater detail on this procedure may be found in the references [5].

Our full implementation of the hybrid PSO algorithm is described by pseudocode in Figure 1. With the input of our data matrix we also choose the number of particles to evaluate on each iteration and the number of iterations to complete. The velocity, or rather the IM for each particle is created by sampling from uniform distributions. The respective velocities are added to a randomized seed vector containing the integers from 1 to the number of samples, thereby rearranging the seed permutation for each particle's initial position so that each one is unique. The best respective particle position, $\mathbf{p}_i$, is set to the initialized position. The fitness test is run on each particle and the one performing the best is stored in the global best position vector, **g**.

Once all of the particles are initialized the algorithm computes the velocity for the next iteration. The velocity for each particle is comprised of three components. The first component, weighted by the free-parameter, w, contributes the respective particle's previous velocity to the current iteration. The second component, weighted by the free-parameter, c_1, contributes the difference between the particle's best and current positions. The third component, weighted by the free parameter, c_2, contributes the difference between the swarm's best and the respective particle's current position. Each velocity is added sequentially to the particle's position and the fitness test is run. The particle's best position, $\mathbf{p}_i$, is updated if the fitness test score exceeds those previous. Likewise, the swarm's best position, **g**, is updated when the fitness test score of any particle exceeds the global best.

Before we continue to the next iteration we perform a local enhancement search on the global best position using the SOP 3-Exchange algorithm [6]. Essentially, this algorithm swaps groups of samples with one another moving iteratively through the sample first in the forward direction, starting with index 1, and then in the backward direction, starting with index n. After each swap the fitness is run on the current permutation and the exchange algorithm terminates once the fitness has increased due to a swap. The global best position is updated and the PSO algorithm continues again with the next iteration. More detail on this procedure may be found in the references [5].

In addition to the PSO algorithm described above, we have implemented an adaptive c_2 parameter algorithm to avoid stagnation of multiple particles converging on the global best position. The procedure is described in Figure 2.

Adaptive c_2 Parameter Pseudocode

· Implemented for stagnation avoidance
· Update parameter c_2 after checking fitness of all particles on each iteration

Initialize:

· Set $c_2 = 1$

Iteration:

For iter = 1:k

1. Set counter $nrk = -1$
2. for each $fit(\mathbf{x}_i^k) = fit(\mathbf{g})$
   $nrk = nrk + 1$
   $\mathbf{x}_i^k = randperm(n)$, (i.e. restart particle)
   end
3. $c_2 = c_2 - 0.01 * nrk$

End

Fig. 2. Pseudocode of the adaptive c_2 procedure

IV. Results

To evaluate the performance of the algorithm on a pragmatic dataset we have used data from a recent publication in which environmental samples were collected along a pH gradient in soil [9]. There were 22 samples in this study selected for analysis, each from soil with a different pH value. The measured pH values ranged from 4.1 to 8. Over 5000 different bacteria were identified by the study with the abundances of each recorded at each sample. We preprocessed this data to remove bacteria that were present in less than 4 samples. The remaining 568 bacteria were included for the study.

Before we ran the PSO algorithm we checked the fitness of the true ordering of the samples. We found that 338 bacteria out of the 568 rejected the null hypothesis that the bacteria were randomly distributed. We then randomly swapped two samples causing a slight reordering of the permutation, and found that the worst-case swap dropped the number of bacteria rejected in the null hypothesis from 338 to 336. We also checked up to 50,000 random permutations and recorded the fitness score as shown in Table I. The highest score obtained was nearly 60% less than our goal of 338. The mean value was 10. Therefore, we believe we have found the global maximum of bacteria rejecting the null hypothesis to be 338.

We ran the algorithm on this dataset using 25 particles and 20, 50, 100, 300 and 500 iterations. We set the free-parameters w, c_1, and c_2 to 1, 1, and 1, respectively. These default values were chosen based on previous implementations of PSO [5]. We ran the algorithm on these conditions both with and without the local search procedure, SOP 3-Exchange. The results are summarized in Table II.

V. Discussion

There are an enormous number of potential solutions to a sequential ordering problem. In our test dataset we have 22 samples, corresponding to a solution space containing 22! possible permutations. We are searching for the permutation that exhibits the best fitness with regard to some specified criteria. It is

TABLE I
Performance of Random Permutations on pH Dataset

	Fitness Score	
Iterations	$High$	$Mean$
5000	121	10
50000	151	10

TABLE II

PERFORMANCE OF PROPOSED ALGORITHM ON PH DATASET

	Fitness Score	
Iterations	$w/o\ LS$	$w/\ LS$
20	86	111
50	161	149
100	156	140
200	186	212
300	190	201
500	175	203

unrealistic to evaluate all of the solutions even for a small number of samples as in this case. Therefore, we have evaluated the potential of using non-exhaustive, heuristic based approaches to our biological sample ordering problem.

From our experiments we have found that iterating the algorithm 20 times, without the LS implemented, yields a fitness score of 86. This is already encouraging since we have only observed 500 different permutations (25 particles, 20 iterations) for this test. In contrast, we observed the fitness of 5000 random permutations resulting in a mean score of 10 and a random high of 121.

Note that in PSO all of the particles are initialized randomly. We may observe cases where fewer iterations score higher. This is exemplified by comparing 50 to 100 iterations. However, we find that increasing the number of iterations for PSO beyond 100 generally provides a solution of increased fitness. We find the optimization is getting 'stuck' around fitness scores of 200. Including the LS method boosted performance with both the 20 and 200 iteration experiments.

We have found that 50,000 random permutations results in a mean fitness of 10 and a random high of 151. These results are exciting since we consistently obtain a fitness of 212 at 200 iterations with only having to check 5000 solutions, plus the local searches. Due to time constraints we have not performed any parameter tuning and are thus using default values for our free parameters. Reasonably, we are optimistic that we may reach our fitness goal of 338 using the described method with parameter tuning and algorithm refinement.

VI. CONCLUSION

In this study we evaluated the feasibility of using optimization techniques to solve sequential ordering problems for the detection of environmental gradients. We have shown that the algorithm as implemented performs consistently given at least 200 iterations. Notably, this is considerably superior than attempting to obtain the correct permutation randomly as Table I suggests. The algorithm's performance, as described in Table II, was obtained without any parameter tuning or further modification. We predict that we will reach the specified fitness goal after all parameters have been tuned.

REFERENCES

[1] G. A. Croes, "A Method for Solving Traveling-Saleman Problems," *Operations Research*, vol. 6, no. 6, pp. 791–812, 1958.

[2] M.W.P. Savelsbergh, "An efficient implementation of local search algorithms for constrained routing problems," *European Journal of Operational Research*, vol. 47, no. 1, pp. 75–85, 1990.

[3] X. Hu, R.C. Eberhart, and Y. Shi, "Swarm intelligence for permutation optimization: a case study of n-queens problem," in *Swarm Intelligence Symposium, 2003. SIS'03. Proceedings of the 2003 IEEE*. 2003, pp. 243–246, IEEE.

[4] LF Escudero, "An inexact algorithm for the sequential ordering problem," *European Journal of Operational Research*, vol. 37, no. 2, pp. 236–249, 1988.

[5] Davide Anghinolfi, Roberto Montemanni, Massimo Paolucci, and Luca Maria Gambardella, "A hybrid particle swarm optimization approach for the sequential ordering problem," *Computers & Operations Research*, vol. 38, no. 7, pp. 1076–1085, July 2011.

[6] Luca Maria Gambardella and Marco Dorigo, "An Ant Colony System Hybridized with a New Local Search for the Sequential Ordering Problem," *INFORMS Journal on Computing*, vol. 12, no. 3, pp. 237–255, July 2000.

[7] C.J.F. Ter Braak and I.C. Prentice, "A theory of gradient analysis," *Advances In Ecological Research*, vol. 34, no. 03, pp. 271–317, 1988.

[8] Devon J Mohamed and Jennifer Bh Martiny, "Patterns of fungal diversity and composition along a salinity gradient.," *The ISME journal*, vol. 5, no. 3, pp. 379–388, Sept. 2010.

[9] Johannes Rousk, E. Bååth, P.C. Brookes, C.L. Lauber, C. Lozupone, J.G. Caporaso, R. Knight, and N. Fierer, "Soil bacterial and fungal communities across a pH gradient in an arable soil," *The ISME Journal*, vol. 4, no. 10, pp. 1340–1351, 2010.

[10] JC Venter, Karin Remington, JF Heidelberg, AL Halpern, and D, "Environmental Genome Shotgun Sequencing of the Sargasso Sea," *Science*, vol. 304, no. 5667, pp. 66–74, 2004.

[11] Ruth E Ley, Fredrik Bäckhed, Peter Turnbaugh, Catherine a Lozupone, Robin D Knight, and Jeffrey I Gordon, "Obesity alters gut microbial ecology.," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 102, no. 31, pp. 11070–5, Aug. 2005.

[12] Peter J Turnbaugh, Ruth E Ley, Micah Hamady, Claire M Fraser-Liggett, Rob Knight, and Jeffrey I Gordon, "The human microbiome project.," *Nature*, vol. 449, no. 7164, pp. 804–10, 2007.

[13] Jacob a Russell, Corrie S Moreau, Benjamin Goldman-Huertas, Mikiko Fujiwara, David J Lohman, and Naomi E Pierce, "Bacterial gut symbionts are tightly linked with the evolution of herbivory in ants.," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 106, no. 50, pp. 21236–41, Dec. 2009.

[14] D. W. Goodall, "Objective methods for the classification of vegetation. III. An essay in the use of factor analysis," *Austral. J. Bot.*, vol. 1, pp. 39–63, 1954.

[15] R. A. Digby, P. G. N., and Kempton, *Population and Community Biology Series: Multivariate Analysis of Ecological Communities.*, Chapman and Hall, London, 1987.

[16] M.O. Hill and HG Gauch, "Detrended Correspondence Analysis: An imporved ordination technique," *Plant Ecology*, vol. 42, no. 1, pp. 47–58, 1980.

[17] N. R. Pace, "A Molecular View of Microbial Diversity and the Biosphere," *Science*, vol. 276, no. 5313, pp. 734–740, May 1997.

[18] Zongzhi Liu, Todd Z DeSantis, Gary L Andersen, and Rob Knight, "Accurate taxonomy assignments from 16S rRNA sequences produced by highly parallel pyrosequencers.," *Nucleic acids research*, vol. 36, no. 18, pp. e120, 2008.

[19] B L Maidak, G J Olsen, N Larsen, R Overbeek, M J McCaughey, and C R Woese, "The RDP (Ribosomal Database Project).," *Nucleic acids research*, vol. 25, no. 1, pp. 109–11, Jan. 1997.

[20] James Kennedy and Russell Eberhart, "Particle Swarm Optmization," *Neural Networks*, vol. 4, pp. 1942–1948, Jan. 1995.

[21] C. L. Mohler, "Effect of sampling pattern on estimation of species distributions along gradients," *Vegetatio*, vol. 54, no. 2, pp. 97–102, Oct. 1983.

[22] Justin Kuczynski, Zongzhi Liu, Catherine Lozupone, D. McDonald, Noah Fierer, and Rob Knight, "Microbial community resemblance methods differ in their ability to detect biologically relevant patterns," *Nature Methods*, vol. 7, no. 10, 2010.

[23] A. Wald and J. Wolfowitz, "On a test whether two samples are from the same population," *The Annals of Mathematical Statistics*, vol. 11, no. 2, pp. 147–162, 1940.

[24] M.P. Austin, "On non-linear species response models in ordination," *Plant Ecology*, vol. 33, no. 1, pp. 33–41, 1976.