Gaussian processes applied to the study of chemotactic movements of droplets

Carlotta Porcelli, qbp693

August 21, 2017

Abstract

The research hereby presented investigates the use of a Gaussian Process regression model on a dataset collected from experiments lead in the context of Artificial Chemical Life for the EVOBLISS project [1] at the REAL laboratory in the IT University of Copenhagen [2] . The experiments performed examine the chemotactic behavior of 1-Decanol droplets placed in Decanoate solutions with different pHs and molarities when a chemical gradient is introduced in the external aqueous environment. The Gaussian process model has been fit to the data collected in order to look for a combination of pH and molarity values of the decanoate solution that would allow the droplet to increase the efficiency of its chemotactic movement.

1 Introduction

The movement of non-living objects in aqueous environments has been used to reproduce the chemotaxis process of a living cell, movement in a particular direction. [3]

The development of chemical systems that mimic the behavior of living systems can be useful in studying related processes in natural living systems.

The data exploited in this research have been collected with the usage of EvoBot [4], a liquid handling robot built in the context of the EVOBLISS [1] project. One of the goal of this project is to develop a robotic platform, which by using artificial evolution, can optimize the performance of a physicochemical system

and its environment.

The experiments are composed of two parts, one is fixed and the other one is variable. The droplet composition and volume along with the salt gradient created, form the fixed part: their values are in fact the same in all the experiments. On the other hand, the variable part is the water phase, generated by two continuous variables: the molarity and the pH of the decanoate (Decanoic Acid) solutions. The experiments carried out by EvoBot [4] span in continuous ranges of molarity, from 5mM to 20mM and pH from 7.0 to 12.3 to create the aqueous environment.

The aqueous combinations explored were created by mixing decanoate solutions at different pHs and adding water to get the molarity desired. Once the solutions were ready, a 1-Decanol droplet was added to the system and a salt gradient was created adding a solution of NaCl. The salt gradient allows the droplet to move from the point in which it was inserted into the system to the salt source, following the gradient created by the salt. In each experiment the droplet movement is tracked for a total time of 60 seconds. The implementation of the tracking assigns a *fitness value* based on the speed and the spatial precision of the droplet. High values represent good chemotactic movements.

Figure 1 shows an example of the tracking of one individual. The tracking starts at the initial position where the droplet is placed. The blue path is composed of the centroids of the droplet after its movement.

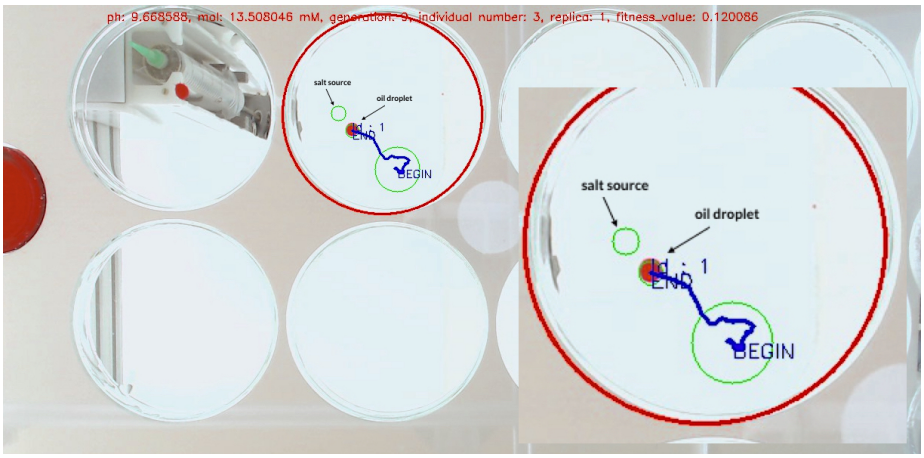

Figure 1: Example of the tracking of an individual. This image is saved for all the experiments and contains the information about the individual, including the pH and molarity of the solution tested, in this case pH=9.66 and molarity=13.50.

The 'END' point is where the program stopped tracking the droplet because it reached the proximities where the salt was added, *salt source*. The droplet reached efficiently the salt source and the fitness value registered is 0.120086.

The aim of the project hereby presented is to make an estimation on the values of pH and molarity forming a decanoate solution in which the droplet performs a smooth and precise chemotactic movement. In this mean, a Gaussian regression model is used to characterize a non-linear relationship between the two variables (pH and molarity). In a classic linear regression algorithm, it is assumed that a dependent variable y can be modeled as a function of one or more independent variables X in the form of $y = f(x) + \epsilon$. The choice of the GP approach relies on the facts that it finds a distribution over the possible functions *f(x)* consistent with the observed data. Moreover, the observations occur in a continuous domain.

Table 1: First 4 data points

pH	**molarity**	**fitness values**
10.07	11.35	-0.027
10.94	10.03	-0.017
9.72	8.97	-0.032
9.84	12.44	0.105

2 Materials and methods

2.1 The dataset

The data set used is composed of 73 experiments. As shown in Table 1 the first two columns contain the values of pH and molarity of the solutions tested, the last column specifies the fitness values computed after the tracking of the droplet. To obtain the data here studied, an evolutionary algorithm has been performed on the chemical system to optimize the ph and molarity of the aqueous solutions for the chemotactic movement.

2.2 Software used

The library from which the Gaussian process regression model has been implemented is the GPy framework from the Sheffield machine learning group. [5]

2.3 The Gaussian Processes

The task of building regression models to characterize non-linear relationships between variables often involves extensive model selection procedures to ensure that the most appropriate model is chosen. An alternative is to use a Bayesian strategy to model the unknown underlying function. Modeling data using Gaussian distributions in some cases would not seem to be any gain because of the reduced flexibility that those distributions have. A Gaussian process is defined as probability distribution over functions f(x) so that the set of values of f(x) evaluated on a set of points $x_1, ..., x_N$ jointly have a Gaussian distribution.
A Gaussian process (GP) assumes that a $p(f(x_1), ..., f(x_N))$ is jointly Gaussian with some mean $\mu(x)$ and covariance $\sum(x)$ given by $\sum_{ij} = k(x_i, x_j)$, where k is a positive definite kernel function [6]. The covariance represents a form of dis-

tance or similarity. Considering two input points x_i and x_j with corresponding observed values y_i and y_j, if the inputs are close to each other, it is expected that that y_i and y_j will be close as well. This measure of similarity is embedded in the covariance function. Since the key assumption in GP modeling is that the data can be represented as a sample from a multivariate Gaussian distribution, we have that

$$\begin{bmatrix} y \\ y_* \end{bmatrix} \sim N\left(0, \begin{bmatrix} K & K^T \\ K_* & K_{**} \end{bmatrix}\right) \tag{1}$$

Where y is the observed value, y_* is the prediction and K is the covariance matrix. The interest is on the conditional probability $p(y_*|y)$ that follows a Gaussian distribution:

$$y_*|y \sim N(K_* K^{-1} y, K_{**} - K_* K^{-1} K_*^T) \tag{2}$$

2.3.1 Choice of Kernel

The kernel used to describe the covariance function expected in the dataset is the Radial-basis function (RBF) kernel defined on two samples x and x' as:

$$K(x, x') = \sigma^2 exp\left(-\frac{\|x - x'\|^2}{2l^2}\right) \tag{3}$$

Where $\|x - x'\|^2$ is the squared Euclidean distance between the two samples and l is the length-scale free parameter describing how smooth the function has to be and σ^2 the variance. The kernel used in the GP model has been implemented as follows:

kg = GPy.kern.RBF(input_dim=2, variance=0.5, lengthscale=1., ARD=True)

Where the **input_dim** parameter represents the dimension of the input space, equals to two variables; the **variance**, determining the variation of function values from their mean, has been set to 0.5 and the **lengthscale** to 1. The flag **ARD=True** indicates having one lengthscale parameter per dimension.

2.3.2 Model parameters and optimization

The model has been implemented using the kernel in the above section and the noise variance parameter set to 1. This parameter specifies how much noise is

expected to be present in the data.

model = GPy.models.GPRegression(XTrain, y_train_set_, kernel=kg, normalizer=False, noise_var=1.)

The model has been later fit to the data using the **model.optimize()** function with maximum number of function evaluations equals to 1000. A value of 10 has been specified for the parameter **n_restarts_optimizer**, which rusn the optimization algorithm as many times as specified, using randomly-chosen starting coordinates to avoid finding a local, rather than a global, maximum in the marginal likelihood.

3 Results

The plot in Figure 2 shows the distribution over the experimental space of the data set. The points are colored accordingly to their fitness value. The plot in Figure 3 shows the Gaussian density distribution of the data before fitting the model.

The plot in Figure 4 shows the Gaussian density distribution after fitting the model on the data set.

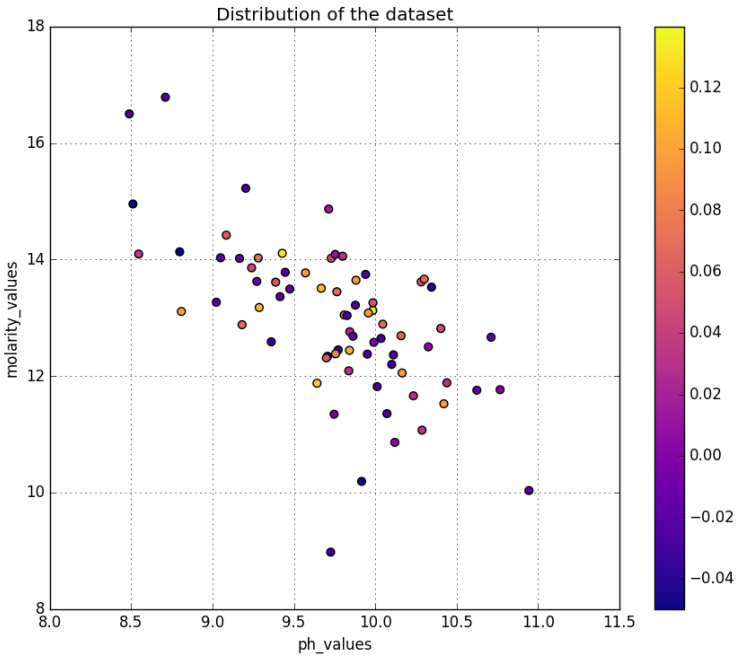

Figure 2: Distribution of the data set colored by fitness values

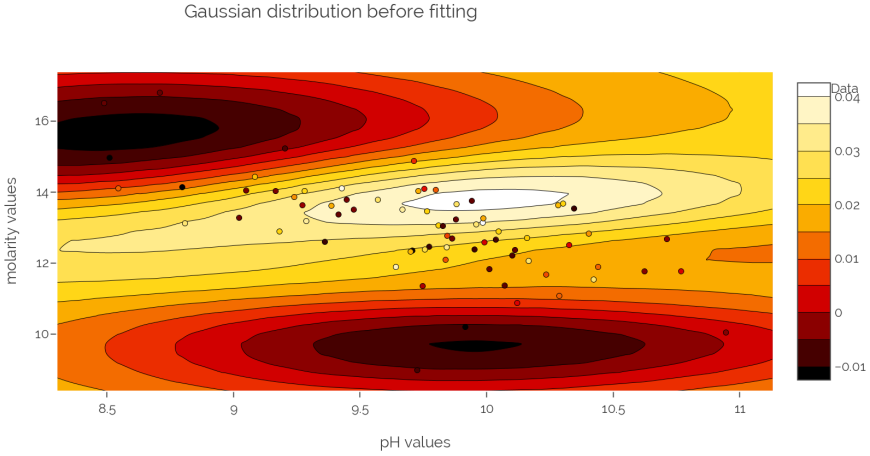

Figure 3: Gaussian distribution before fitting the non normalized data

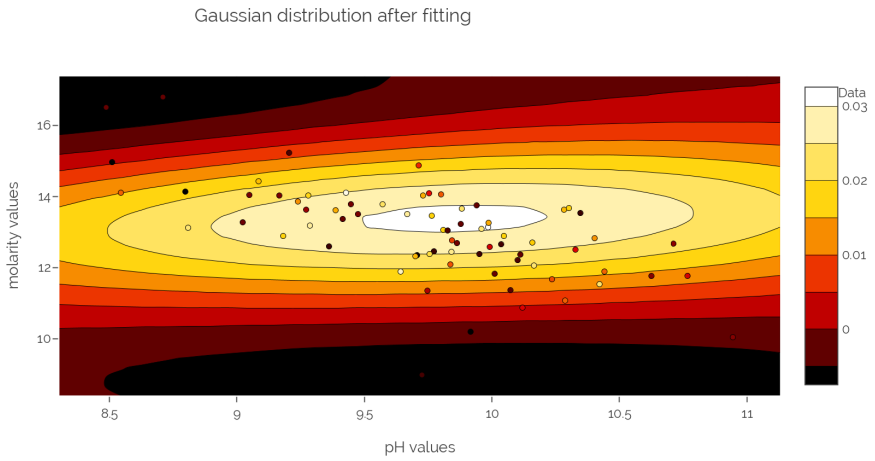

Figure 4: Gaussian distribution after fitting the non normalized data

4 Discussion

From the plot in Figure 4 it is possible to observe the lighter area bounded by values $9.5 < pH < 10.2$ and $12 < molarity < 14$. In this area the samples should get higher fitness values than the darker areas depicted around it. However, it can be observed that many samples in that area have darker color, meaning lower fitness values. This could be explained by the fact that a high number of tested individuals, even having very similar pH and molarity values, produced very different results. This is also shown in the plot in Figure 2 where many points close to each other, or almost overlapping, have different fitness values. It has to be said that, being the data a result of evolutionary experiments, some areas of the search space are oversampled, this is where the evolutionary algorithm converged. The chemical system studied is unfortunately highly unstable. In many cases it has been noticed that Marangoni flows, superficial tensions between droplets and the edge of the Petri dish or air flows, affected the movements of the droplet resulting in low fitness values even though the same aqueous phase might have produced higher results in the previous experiment. A collection of data from previous experiments, not presented here, showed that a high level of noise was present. To smooth the noise in the system the number of experiments for each individual was increased to four and the final fitness

value of each experiment was chosen to be the median of the four replicas instead of the average, in order to minimize the influence of the outliers. Regrettably it seems that there still is a high level of noise in the evaluation.

4.1 Future work

To solve the problems related to the chemical system, future work has been programmed to improve the results of these experiments. A change in the protocol for making the solutions is being studied and a covering to avoid air flows from the outside has been designed. On the computational side, different type of kernels or combinations of them could be explored to check for different results. Even though the system is noisy, experiments within the above found range of pH and molarity could be lead in order to check if the fitness values match the ones indicated from the model.

References

[1] https://blogit.itu.dk/evoblissproject/overview/progress/artificial-chemical-life/, August 19, 2017.

[2] https://real.itu.dk/projects/evobliss/, August 19, 2017.

[3] J. Cejkova, M. Novák, F. Stěpánek and M.M. Hanczyc (2014) Dynamics of tactic droplets in salt concentration gradients. *Langmuir* 30(40): 11937-44.

[4] Andres Faina, Farzad Nejatimoharrami, Kasper Støy, Pavlina Theodosiou, Benjamin Taylor, Ioannis Ieropoulos. "EvoBot: An Open-Source, Modular Liquid Handling Robot for Nurturing Microbial Fuel Cells" in *Proceedings of the Artificial Life Conference 2016* .United States: MIT Press, 2016, pp. 626–633.

[5] GPy: A Gaussian process framework in python - http://github.com/SheffieldML/GPy, August 19, 2017.

[6] Robert, Christian. "Machine Learning, a Probabilistic Perspective." (2014): 62-63.