List of Publications

Bachelor's Thesis

G. Qian, J. Mei, H. H. C. Iu and S. Wang, "Fixed-Point Maximum Total Complex Correntropy Algorithm for Adaptive Filter," in IEEE Transactions on Signal Processing, vol. 69, pp. 2188-2202, 2021, doi: 10.1109/TSP.2021.3067735. keywords: Convergence;Adaptive filters;Signal processing algorithms;Filtering;Adaptation models;Kernel;Filtering algorithms;Complex domain;errors-in-variables;adaptive filtering;fixed point,

Master's Thesis

J. Mei, L. Gulyás, J. Botzheim, "Comparing Lamarckian and Baldwinian Approaches in Memetic Optimization," in Advances in Computational Collective Intelligence. ICCCI 2023. Communications in Computer and Information Science, vol 1864. Springer, Cham, 2023, doi: 10.1007/978-3-031-41774-0_41.

Patents

CN117236233A **Inventor: Mei Jiaojiao, Gong Ding, Liu Yaxiong**

A Method for Initial Value Evaluation in Semiconductor Devices

CN117236234B **Inventor: Mei Jiaojiao, Gong Ding, Liu Yaxiong**

A Method for Predicting Iteration Steps in Semiconductor Device Simulation

List of Publications

Bachelor's Thesis

1.1 Fixed-Point Maximum Total Complex Correntropy Algorithm for Adaptive Filter

Link to the paper: Fixed-Point Maximum Total Complex Correntropy Algorithm for Adaptive Filter

Abstract: The core idea of this paper is to introduce flexible learning rates instead of a fixed learning rate in the MTCC algorithm (prior work). MTCC utilizes gradient descent for optimization. A known drawback of classical gradient descent is the fixed learning rate (also known as step length), where a large fixed learning rate can result in overshooting the minimum, while a small one may slow down the convergence speed, as illustrated in Figure 1.1 and Figure 1.2. The enhancement proposed in this paper involves initially setting a large learning rate to rapidly narrow down the search space, followed by a smaller learning rate to refine solution precision.

To comprehend the paper, readers also need some background knowledge on Complex Correntropy, signal processing, and adaptive filtering. Other related topics may also be pertinent. The mathematical proofs therein were completed by my bachelor advisor, while my work primarily focuses on coding and paper writing.

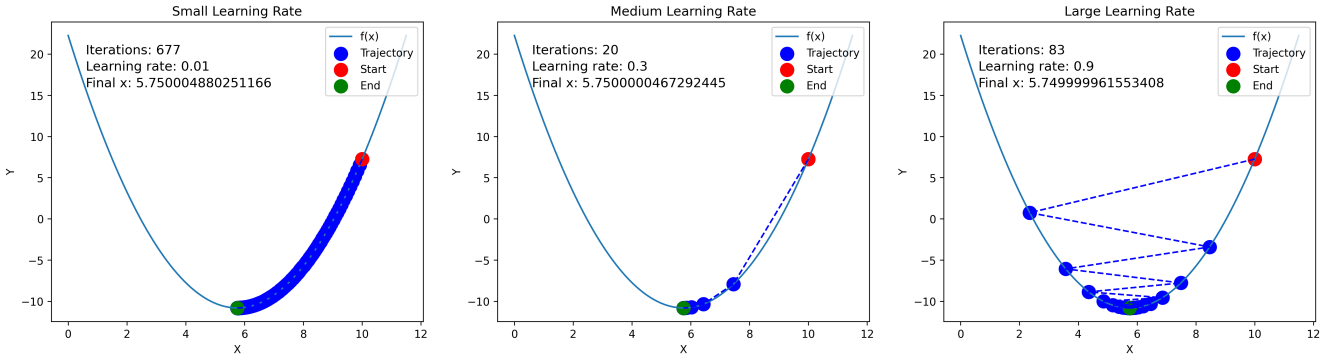

Figure 1.1: Gradient Descent with different learning rates $f(x) = (x - 3.5)^2 - 4.5x + 10$

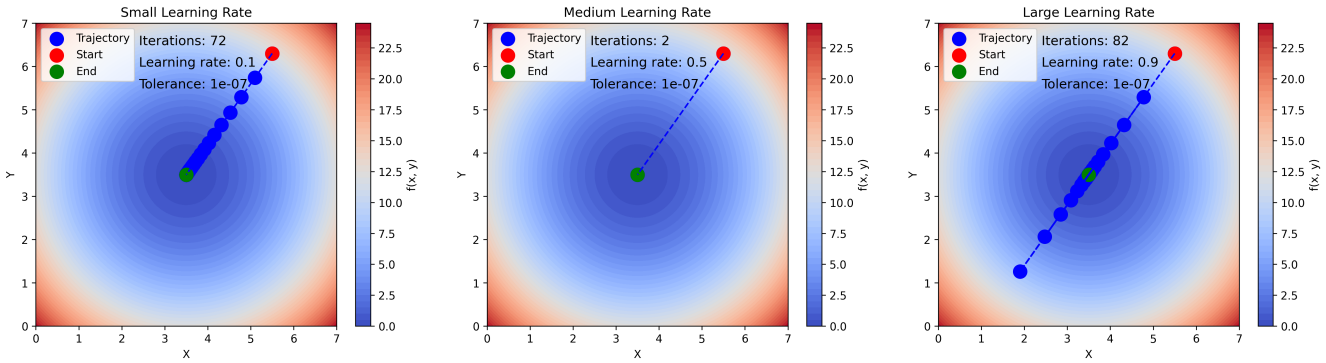

Figure 1.2: Gradient Descent with different learning rates $f(x, y) = (x - 3.5)^2 + (y - 3.5)^2$

Master's Thesis

2.1 Comparing Lamarckian and Baldwinian Approaches in Memetic Optimization

Link to the paper: Comparing Lamarckian and Baldwinian Approaches in Memetic Optimization

Abstract: The main focus of this paper is to enhance genetic algorithms with effective local search procedures. For simplicity, let's denote the genetic algorithm as **A** and the genetic algorithm augmented with local search (referred to as memetic algorithm) as **B**. The goal is to find the global minimum(s) of a given function f(x) within a limited number of evaluations. Given the same number of evaluations, we compare the overall performance of **A** and **B** on a range of benchmark functions with diverse properties and dimensions, part of which is shown in Figure 2.1. After finding effective local search methods which makes **B** superior to **A**, we further compare the performance of Lamarckian and Baldwinian approaches (two different variants of **B**) utilizing the same effective local search. It's worth noting that while both are categorized as **B**, they employ distinct methodologies for utilizing this local search.

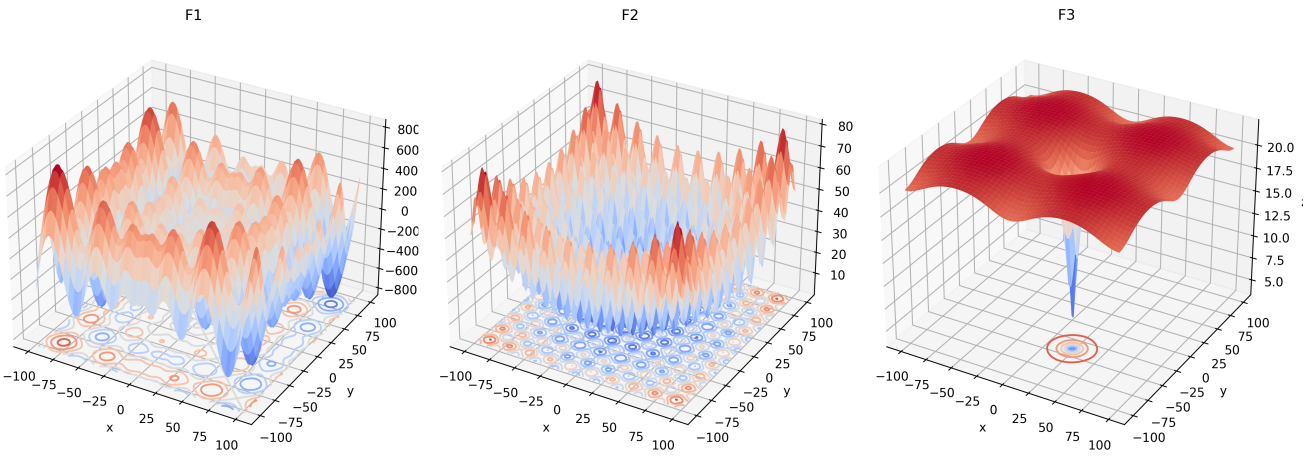

Figure 2.1: Part of the CEC-BC-2017 benchmark functions $F_1(x)$, $F_2(x)$, and $F_3(x)$ when $n = 2$

Algorithms like A or B, which are inherently stochastic and do not guarantee finding the optimal solution, are commonly utilized for optimizing parameters. Often, parameters interact with each other, and it becomes necessary to find relatively good combinations of parameters. In such cases, these algorithms can be considered.

$$F_1(x) = -\sum_{i=1}^{n} x_i \sin(\sqrt{|x_i|})$$

$$F_2(x) = \sum_{i=1}^{n} \left(x_i^2 - 10\cos(2\pi x_i) + 10\right)$$

$$F_3(x) = \left(-20\exp\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}\right) - \exp\left(\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)\right) + 20 + e\right)$$

Patents

3.1 A Method for Initial Value Evaluation in Semiconductor Devices

Link to the patent: A Method for Initial Value Evaluation in Semiconductor Devices

Some background knowledge: Newton's method, renowned for its quadratic convergence in root-finding, heavily relies on the initial value's proximity to the root or the system's conditioning for efficacy. If this condition is not met, convergence may fail. To understand this, let's examine fixed-point iteration. If the absolute value of the derivative of the iteration function at the fixed point, denoted as $|g'(x)| > 1$, the fixed point repels solutions; if $|g'(x)| < 1$, it attracts solutions. When $|g'(x)| = 1$, a more detailed analysis of the fixed point's behavior is required. In essence, the rate of convergence towards the fixed point depends on $|g'(x)|$. The closer it is to 0, the faster the convergence of the iterates. We can transform $f(x) = 0$ into a fixed-point problem by defining $g(x) = x - \frac{f(x)}{f'(x)}$, where $f'(x)$ represents the derivative of $f(x)$. The derivative of $g(x)$ at the fixed point x is $g'(x) = 1 - \frac{f'(x)^2 - f(x)f''(x)}{f'(x)^2}$. Obviously, $g'(x) = 0$ when $f(x) = 0$, so we should have pretty fast convergence here. When $f'(x) = 0$, convergence is not guaranteed, rendering Newton's method ineffective.

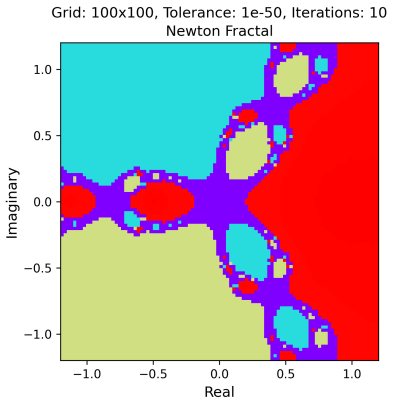

Figure 3.1: Grid:100x100, Iterations:10, Tol:1e-50

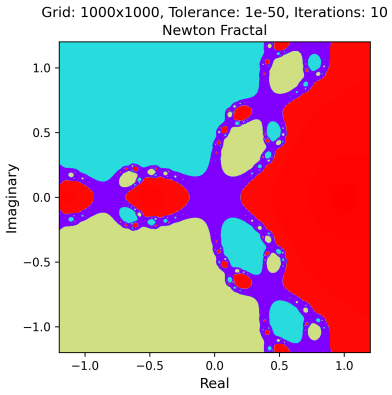

Figure 3.2: Grid:1000x1000, Iterations:10, Tol:1e-50

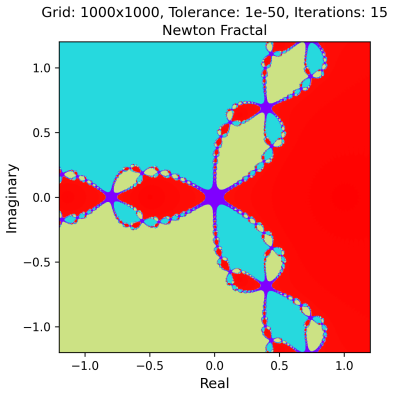

Figure 3.3: Grid:1000x1000, Iterations:15, Tol:1e-50

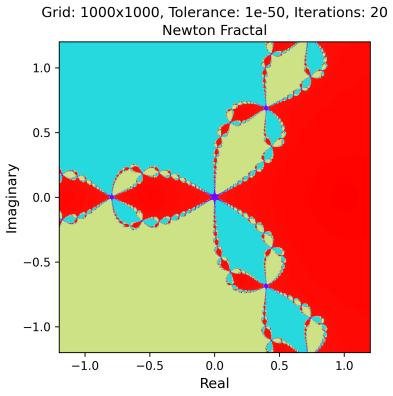

Figure 3.4: Grid:100x100, Iterations:20, Tol:1e-50

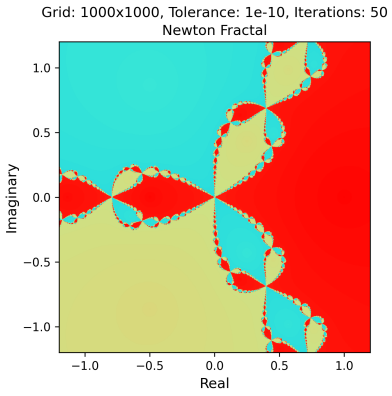

Figure 3.5: Grid:1000x1000, Iterations:10, Tol:1e-10

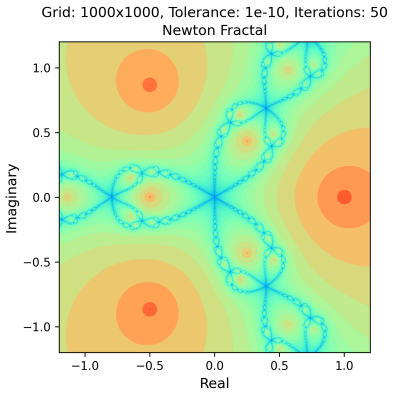

Figure 3.6: Grid:1000x1000, Iterations:15, Tol:1e-50

Abstract: Figure3.1, Figure3.2, Figure3.3, Figure3.4, and Figure3.5 illustrate the Newton fractals generated by the function $f(x) = x^3 - 1$ with different tolerances, grid sizes, and iterations. Actually, whether a point can converge or not is also influenced by the precision of the floating-point arithmetic. But no matter what, the nature of the function is the most important factor. This simple function has 3 roots which are $r_1 = 1, r_2 = -\frac{1}{2} + \frac{\sqrt{3}}{2}i, r_3 = -\frac{1}{2} - \frac{\sqrt{3}}{2}i$. It just serves as an example here. Each root is rendered with a distinct color. If starting from a certain initial value leads to convergence to one of the roots, the color of the initial value matches that of the corresponding root. Otherwise, if no convergence is achieved, the initial value is represented in purple. As you can imagine, the distribution of points that do not converge should be roughly as Figure3.6 shows, though I only distinguish between convergence and non-convergence Figure3.6. For the convergent parts represented in orange, the darker the color, the fewer iterations are required.

The purpose of this patent is to utilize machine learning methods to perform binary classification on points that can converge and points that cannot converge. By training a binary classification model, the goal is to predict whether a new point can converge, that is all. (Now, perhaps you understand why machine learning relies heavily on big data. Without sufficient data, patterns may not emerge.) $f(x) = x^3 - 1$ serves as an example. Most engineering problems, regardless of complexity, can be reduced to linear algebra: $Ax = b$, $Ax \approx b$, $Ax = \lambda x$. Newton's method plays a crucial role in root-finding and is perhaps the most widely used one. Hence, there is a need to optimize it.

3.2 A Method for Predicting Iteration Steps in Semiconductor Device Simulation

Link to the patent: A Method for Predicting Iteration Steps in Semiconductor Device Simulation

Abstract: This patent, similar to its predecessor, centers on the development of a sophisticated multiclass classification model. Specifically, the model is designed to predict the number of iterations required for convergence, given a particular initial value. Training data is like $(x, \text{number of iterations}[1-5]) \rightarrow$ Good, $(x, \text{number of iterations}[6-10]) \rightarrow$ Medium, $(x, \text{number of iterations}[11-15]) \rightarrow$ Bad, $(x, \text{number of iterations}[16, +\infty]) \rightarrow$ Unconverged. Of course, one can modify the intervals and the number of classes.

Drawbacks of these two patents: Figure 3.7 illustrates the performance of different classifiers. The decision boundaries portrayed by these classifiers are too coarse for fractals. This is primarily due to the inherent complexity of fractals, which are infinite patterns characterized by constant repetition. In summary, because of the intricate and self-similar nature of fractals, the classifiers face challenges in accurately delineating their intricacies. The binary classification model is sufficient engough to use by achieving a high accuracy 85% to 95% on test data, but the multiclass classification model is not mature enough to be used in practice.

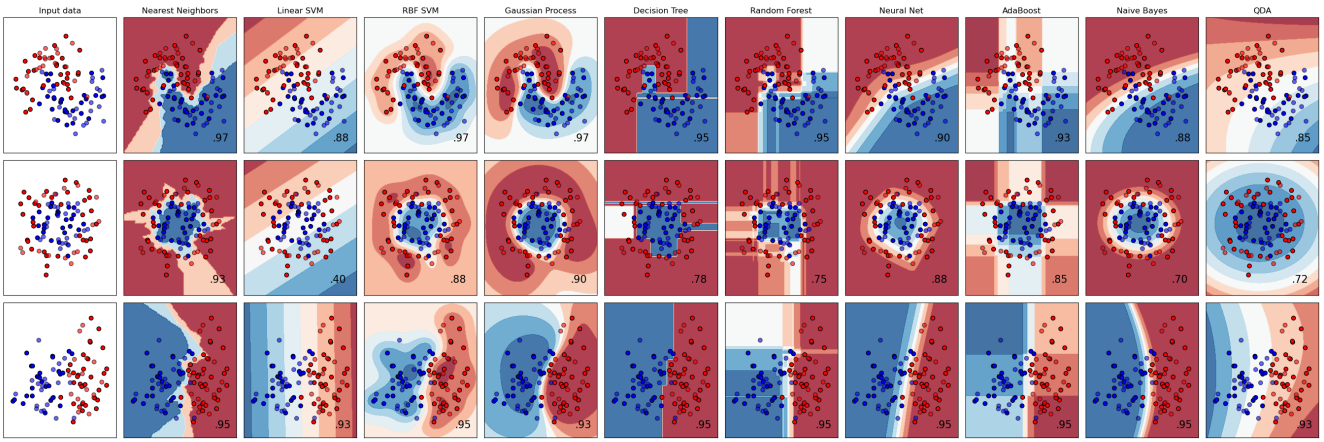

Figure 3.7: Link to the picture: Comparison of different classifiers by Scikit-learn

Inprovement: The improvement is to introduce Fourier features into neural networks, which can improve accuracy. This is the inspiration I got from positional encoding. Simply put, neural networks are universal function approximators, and Fourier series also approximate functions, so we can consider combining them. But I cannot disclose too many details because some other related patent application of mine is still ongoing.

you may find the code for the plots here List of Publications.