NAIST-IS-DD861017

Doctoral Dissertation

Statistical Analysis of Optimization Algorithms

Hiroyuki Funaya

February 20, 2011

Department of Information Systems
Graduate School of Information Science
Nara Institute of Science and Technology

Hiroyuki Funaya

Thesis Committee:	
Professor Kazushi Ikeda	(Supervisor)
Professor Yuji Matsumoto	(Co-supervisor)
Associate Professor Takashi Takenouti	(Co-supervisor)

Statistical Analysis of Optimization Algorithms*

Hiroyuki Funaya

Abstract

Mathematical algorithms have been performing important roles in the modern society; optimization to find the best solution to mathematical problems, classification to find groups in observations, and statistical analysis to find an underlying regularities in various phenomena in the nature. Inspite of the great results that these algorithms have been yielding, their mathematical mechanisms have yet to be so well studied. In this dissertation, three subjects are studied: a genetic algorithm, an SVM with forgetting factor, and neural spike sequences. For genetic algorithms, we propose a new analytical method from a network point of view. First the genetic algorithm is formulated as a netowrk with nodes representing generations connected by two genetic operations: crossover and mutation. Then, its characterstic path length is derived mathematically. For support vector machines, a forgetting factor is proposed and the asymptotic generalization ability is derived for a simple linearly separable problem. The learning curve is mathematically derived for two types of forgetting factors: exponential and factorial forgetting factors. It is proved that the learning curve of factorial forgetting factors converges to zero in the asymptotic state. At last, for the analysis on neuron spike sequences, information geometrical method is employed to classify neurons in cortical areas. The interspike intervals of a spike sequence of a neuron is modeled as a gamma process with a time-variant spike rate, and a semiparametric estimation problem is forumulated for the parameters and is solved using an information geometrical method to derive the optimal estimators from a statistical viewpoint.

*Doctoral Dissertation, Department of Information Systems, Graduate School of Information Science, Nara Institute of Science and Technology, NAIST-IS-DD861017, February 20, 2011.

Keywords:

Genetic Algorithms, network, SVM, asymptotic analysis, spike

最適化アルゴリズムの統計的解析[*]

船谷 浩之

内容梗概

数理的アルゴリズムは現代社会に多大な貢献をしている。最適化アルゴリズムは、数理的に定式化された問題に対し最も良い解決法を求めるアルゴリズムである。また、クラス分けアルゴリズムは、ある観測に対し、どの点がどのグループに属するかを学習し、予測するアルゴリズムである。また自然界のデータに対し背後に潜む規則性を見つける為に、統計処理が盛んに行われて、有用な結果が多数報告されている。しかし、これらのアルゴリズムが、数理的になぜ、いい結果をだすか、という事に関してはあまり知られていない。本博士論文では、最適化アルゴリズムとして遺伝的アルゴリズム、クラス分けアルゴリズムとして SVM, そして自然現象として神経スパイク系列を取り上げ、それぞれ数理的、統計的に解析を行った。

キーワード

遺伝的アルゴリズム, SVM, 神経スパイク, 解析

[*]奈良先端科学技術大学院大学 情報科学研究科 情報システム学専攻 博士論文, NAIST-IS-DD861017, 2011 年 2 月 20 日.

Contents

List of Figures

1. Introduction

Mathematical algorithms have been performing important roles in the modern society; optimization to find the best solution to mathematical problems, classification to find groups in observations, and statistical analysis to find an underlying regularities in various phenomena in the nature. In spite of the great results that these algorithms have been yielding, their mathematical mechanisms have yet to be so well studied. In other words, little is known about how and why it works so flexibly in many applications. In this dissertation, three subjects are analyzed: a genetic algorithm, an SVM with forgetting factor, and neural spike sequences. For genetic algorithms, we propose a new analytical method from a network point of view. First the genetic algorithm is formulated as a network with nodes representing generations connected by two genetic operations: crossover and mutation. Then, its characteristic path length is derived mathematically. For support vector machines, a forgetting factor is proposed and the asymptotic generalization ability is derived. At last, for the analysis on spike sequences, information geometrical method is employed to classify neurons in cortical areas. Actually, GA is a good quasi-optimizer; it provides good solutions in realistic time in many situations, provided that the experimenter sets fair conditions in advance. In short, GA works through three stages: selection, reproduction, and termination. Of all these stages, what is mainly controlling the algorithm is the reproduction stage, in which results called "individuals" produce next generations with two operations: mutation and crossover. Although some studies have been carried out to reveal the mathematical meaning of these operators, Little is known so far. In particular, there has been no accounts for the functional relationship between these two.

GAs are analyzed from a network point of view. A GA is regarded as a network where each node and each edge respectively represent a population and the possibility of transition through mutation and crossover. The concept of characteristic path length (CPL) is employed, and it is one of the most popular criteria in the theory of small-world networks**??** defined as the average shortest path length (SPL) between a pair of populations. We analytically derive the CPL in the two following cases: (a) We simply count the minimum number of necessary operations from a population to another, and (b) we generalize the discussion by introducing the concept of weights to the edges which represent the transition probabilities of the corresponding operations. As a result, in the case of (a), the crossover operation shortens the CPL linearly with the length of individuals. In the case of (b), the minimum value of the CPL is found, but the crossover operation does not necessarily shorten the CPL depending on the probability of the mutation.

Genetic Algorithms are difficult to be analyzed because they are meta-algorithms that solve general problems in an indirect way. They map the original spaces into genetic spaces where normal metrics are no longer valid. However, more problem-specific algorithms such as support vector machines (SVMs) for classification are still difficult to analyze.

Second, support vector machines (SVMs), which are considered to be some of the most successful learning classifiers in the last decades [8–11], are analyzed. Since original SVM treats all its examples equally, it cannot be applied to time-varying environments. What makes the situation more difficult is that standard quadratic programming (QP) technique applied in SVM becomes infeasible for large data sets, and training an SVM requires solving a QP problem with a number of coefficients equal to the number of training examples.

One method to cope with these difficulties is an on-line approach, where only part of the training examples are selected [12] [13]. In this approach it is important to decide the number of examples (generally based on some heuristics) and there exists a limitation: it is not possible to set the weights of the examples freely. In other words, each example has only zero or one as its weight.

We can overcome this limitation by introducing forgetting factors [14] like the RLS algorithms for adaptive filters [15], which naturally puts non-integer weights on examples to select the filter coefficients. Although many variants of forgetting factors for an RLS algorithm have been proposed [16], those fundamentally use exponentially weighted forgetting factors such as λ^t as its weights, where λ denotes a forgetting factor and t denotes a time index. In this dissertation this is called exponential forgetting factor.

Although the idea of a forgetting factor is convenient for a time-varying problem, the cost function introduced in previous work [14] was just the sum of squares followed by the Tikhonov regularization, where the inequalities for margin maximization were replaced with the corresponding equations. Hence, the proposed algorithm has lost the convenient properties of SVMs such as sparse solutions. Therefore, the forgetting factor to SVMs is introduced in this study in a more natural way.

As the last subject of this dissertation, we study neural spike sequences using information geometry. The characteristics of neurons in cortical areas have been the subject of recent discussion in the literature. In particular, there has been discussions on the statistical properties of the inter-spike intervals(ISIs) of spike sequences such as the coefficient of variation, C_v; the skewness coefficient, S_k; the correlation coefficient of consecutive intervals, C_{OR}; and the local variation, L_v. In particular, Shinomoto et al. have shown that the local variation with a refractory period, L_{VR}, can almost classify the functions of neurons without any other information.

From the viewpoint of statistical modeling, ISIs can be modeled as gamma processes with a variable rate but a fixed shape parameter. Since gamma distributions form a two-dimensional e-flat manifold S from the information-geometrical viewpoint, the problem of estimating the shape parameter results in a semi-parametric estimation. These theoretical methods can be applied to estimate the refractory period in L_{VR}, which has heuristically been determined to date.

We classify neurons by first introducing new statistical measures of spike sequences and their properties, and formulating the problem of ISIs classification as a semi-parametric estimation, and at last deriving statistically optimal estimators for semi-parametric models.

The outline of this dissertation is as follows: In Sect.2, GAs are analyzed from the network viewpoint. In Sect.3, asymptotic generalization errors of SVMs with forgetting factor are analyzed. In Sect.4, an information-geometrical method is developed for classifying neurons in cortical areas. Sect.5 concludes this dissertation.

2. Network analysis of Genetic Algorithms

2.1 Genetic Algorithms

2.1.1 Basic Procedure

In this section, the basic procedures of genral GAs are presented. First, some technical terms are defined for the following sections.

An “Individuals” is a candidate of the optimal solution of a GA. Each individual is converted from the original problem space into what is called “genetic space” usually represented by series of bit-arrays. A set of individuals is called “population”, and throughout the process of a GA, basically one population is evolved to find a quasi-optimal solution.

Here is the basic procedure of a GA:

1. **Initialization** Many individuals are generated to form the initial population. In the most basic setting, they are randomly generated.

2. **Selection** A certain proportion of individuals are selected from the current population, on the basis of a fitness function defined in advance. The solutions with higher degree of fitness are more likely to be selected. According to Darwinism, the fitness function is also called an “environment”, and the power from the environment to terminate the weaker individuals is called “selection pressure.”

3. **Reproduction** In the next step, the new generation is spawned using the genetic operators; crossover and/or mutation.

   The genetic operators are employed to spawn the new generation in the following way; with the mutation operation, some bits of an individual are selected arbitrarily in both number and position and flipped, from 0 to 1 or 1 to zero. The crossover operation swaps a group of bits between two individuals, as shown Fig. 1. The main difference between these two is that the mutation operation works only on one individual, while the crossover operation works on two of them.

4. **Finalization** Step 2. and 3. are repeated until a termination condition is fulfilled. Common terminating conditions are, for example,

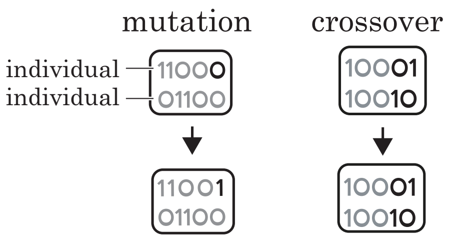


Figure 1. Genetic operators; mutation and crossover

- a solution is found that satisfies some minimum criteria, or
- a fixed number of generations are generated

2.1.2 Effectiveness of Crossover

The main purpose of this study is to clarify the difference between the crossover and the mutation, based on a mathematically measurable metric. In this subsection, the effectiveness of the crossover operation is discussed mainly by comparing it with the mutation operation. The method to asses the role of the crossover operation is to examine a network property of GAs. In the following discussion, "mutation/crossover operation" is just denoted as "mutation/crossover".

For any optimization problems, GAs with only mutations usually require an infinite time to find the optimal solution **??**. Shortly to say, this is because a mutation operation is virtually a random walk in the problem spaces.

On the other hand, a crossover operation swaps blocks of bits. That is, it is observed that crossovers seem to make "shortcuts" in the structure of the transition network of a GA. To mathematically formulate this observation and evaluate how much a crossover shortens the transition network, we introduce a network analysis in the next section.

Note that this network analysis is free from problem-specific properties different from other approaches. For example, the theory of fitness landscape [6] is well studied on many kinds of problems, but it does not describe the GA's quickness to find quasi-optimal solutions generally since fitness landscape, or the shapes of fitness functions varies depending on the problems and the parameter settings of GAs as well as coding of information in an individual.

2.2 Network Analysis

Network analysis has recently attracted much attention as a novel method to analyze complex phenomena. The most prominent property of this approach is the what is called “small world”, which initially motivated this study because it is expected that a GA which has the small-world property takes a shorter time to find a quasi-optimal solution. In the following the famous small-world experiment by Stanley Milgram is briefly introduced. Then, the small-world network model invented firstly by Watts and Strogatz [5] is introduced to show how two genetic operators correspond to the small-world property

2.2.1 Small-world experiment

The small-world experiment was conducted by Stanley Milgram examining the average path length for social networks of the people in the United States [7]. The research was surprising in that it revealed that human society is a small-world network characterized by the path lengths that is shorter than expected. The procedure of this experiment is briefly written as follows:

1. Milgram chose some individuals in the U.S. cities and set them the start and end points:

2. Information packets were initially sent to randomly selected individuals. They included letters, which detailed the stud’s purpose, and basic information about a target contact person:

3. Upon receiving the invitation to participate, the recipient was asked whether he or she *personally* knew the contact person described in the letter. If so, the person was to forward the letter directly to that person:

4. In the more likely case that the person did not personally know the target, then the person was to think of a friend or relative they know personally that is more likely to know the target:

5. When and if the package eventually reached the target person, the researchers could count the number of times it had been forwarded from person to person.

The result is, 64 of the letters eventually did reach the target contact. Among these chains, the average path length fell around 5.5 or six, very smaller than we think.

2.2.2 Characteristic Path Length

A CPL is defined for a network, as the shortest path length (SPL) between two nodes averaged over all possible pairs. In general, the networks with a same number of nodes are compared with this criterion because it is obvious that the networks with small number of nodes has relatively smaller CPLs compared to those with large number of nodes. So this criterion shows also how the nodes are connected, which is our interest in the analysis of GAs.

2.2.3 Small-world network model

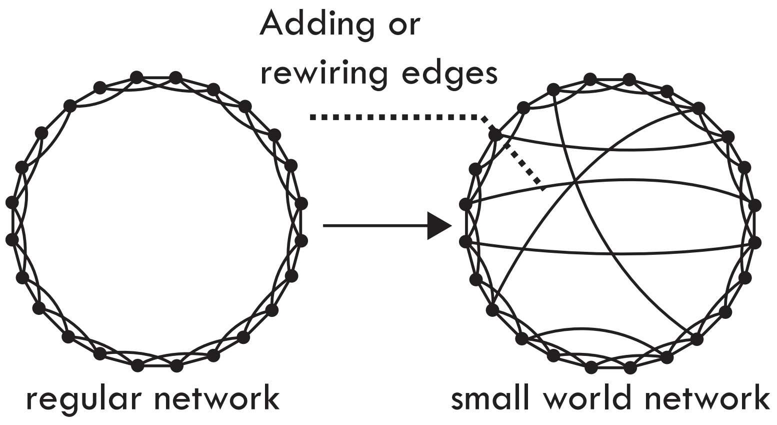


Figure 2. Small-world network model

To see the connectivity in a small-world network, the small-world model is introduced. This model begins with a regular lattice. Then we add some of random links. This operation reduces the diameter. It is surprising that the diameter reduces dynamically in spite of adding only a small number of links. (Fig. 2). This phenomenon implies that the both weak and strong connections are necessary for the small-world property. Now, let us consider the GAs. GAs have similar structures to the small-world network in that the crossover corresponds to the weak connection and the mutation corresponds to the strong connection. Hence, under the assumption that a GA with a smaller CPL takes a shorter time to find a solution, one can study how the two basic genetic operations in GAs, crossover and mutation, affect the CPL.

In the rest of this dissertation, a GA network is defined, and then its CPL is derived in purely mathematical way.

2.3 Networks of GAs

In this section, the networks of GAs are defined. The simplest case is considered so that the calculation is possible; each individual consists of a binary sequence of length L. That is, there are 2^L individuals, which form a population as a whole. It is assumed that each population has only two individuals at first and more general cases is discussed later. Then, the cardinality of the different populations becomes

$$\begin{aligned} N &\equiv \frac{1}{2}(2^{2L} + 2^L) && (1) \\ &= 2^{L-1}(2^L + 1), && (2) \end{aligned}$$

since the duplication of the same kinds of populations has to be excluded. When the generation evolves, individuals are modified only by one of the two basic genetic operations, one-point crossover or one-bit mutation. Note that any fitness function is not asssumed because the main purpose of this study is to focus on the basic networking structure of the GA.

In one GA, the lengths of all the indivisdual are assumed to be identical, and it is denoted by L. Then, the GA network consisted of those individuals is denoted by $\mathbf{G}_L$. The number of populations are denoted by N, which means GA network $\mathbf{G}_L$ has N nodes. Two nodes are linked by an edge if and only if one of the two nodes can be changed to each other only with a one-point crossover or a one-bit mutation operation (Fig. 3)

If a network consist of only the edges from mutation and we consider no combination of genes, it is a lattice of the $2L$-dimensional hypercube because an individual is an L-bit sequence and a population consists of two individuals. Therefore, the path-length of any two distinct nodes is the same as the Manhattan

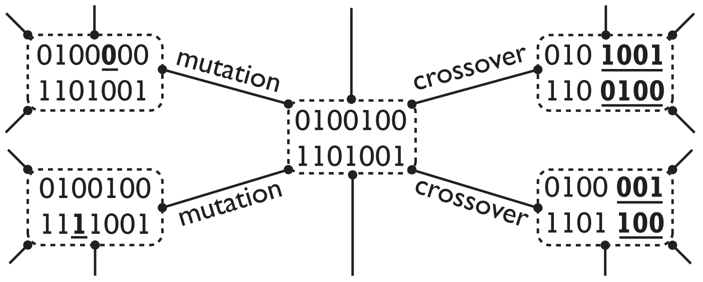


Figure 3. A part of $\mathbf{G}_L$; the network of a genetic algorithm

distance, that is, L on average. If we consider some combinations of genes, the path-length is expected to be a little smaller than L, as is described below.

In the GA network consisted of the crossover and mutation, an edge corresponding to the crossover is considered to be a shortcut in the network because multiple bits are changed at once. It is likely that these shortcuts enable a GA to find a quasi-optimal solution in a shorter time than it is expected, without the explosion of the number of possible transitions. The purpose of this study is to quantitatively evaluate how these shortcuts shorten the CPL.

2.4 Characteristic Path Length

In this section, we derive the CPL of a GA network. The derivation is achieved by counting the number of necessary operations from the initial population to the optimal population.

In many cases of the network analysis, the CPLs are calculated from the empirical data collected. Differently, the CPL of the GA network can be derived analytically due to its simple structure. First, the properties of the crossover and mutation are inspected, and then the idea to derive the shortest path length (SPL) between arbitrary two populations is explained. The CPLs are calculated under the following two different conditions; one network consists of only the edges of mutations and the other consists of also the edges of crossovers.

2.4.1 Properties of crossover and mutation

In the GA network $\mathbf{G}_L$, its genetic operations, the mutation and crossover, simplify the calculation process of the CPL. Before going into detail, additional terms are defined as shown in Fig. 4: "Population" denotes a set of individuals; there are only two individuals in this study. "Pair of populations" denotes literally a pair of populations, one of which is called the original population and the other is called the destination population. "Locus" means a position in a bit sequence, 1 to 7 in this example. "Pattern of loci" denotes four bits lined vertically in a pair of populations. The patterns of loci play an important role in calculating the CPL.

For example, Fig. 5 shows the patterns of loci. There are $2^4 = 16$ patterns and they are classified into four types, denoted as T_1, T_2, T_3, T_4, T'_4 respectively, based on how two populations can be matched by genetic operations:

Type 1 All four genes have the same alphabet.

Type 2 The two genes of a population are the same but the two genes of the other population are different.

Type 3 The two genes of each population are the same but the two populations have different alphabets.

Type 4 Each population has two different genes. That is, the genes at the locus are 0110, 1001, 0101 or 1010. The former two are termed Type 4-1 while the others Type 4-2.

We will call T_1, T_2 and T_3 the patterns of mutation and call T_4 and T'_4 the patterns of crossover.

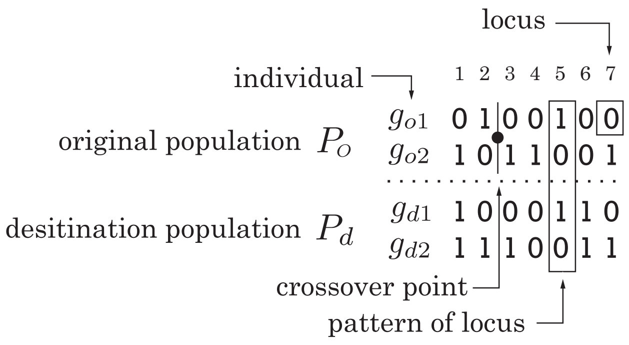


Figure 4. Pair of populations and terms

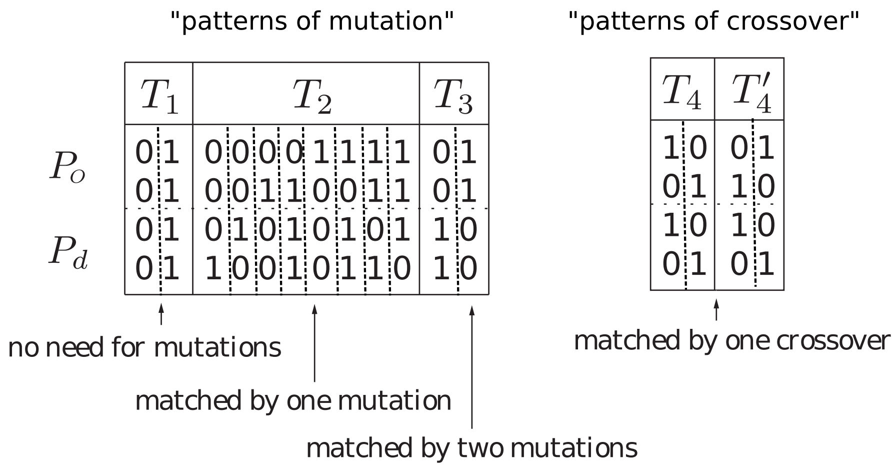


Figure 5. pattern of loci

Note that the crossover never changes the type of a locus only because they are defined so, and the mutation works bit-wise. Instead, the patterns belonging to Type 1, 2 and 3 respectively change zero, one and two bits. Moreover, each operation is independent from any other operations and never affects other part of the patterns. Hence, the SPL of two nodes is the sum of the above and the SPL of the two shorter nodes consisting of only the patterns of Type 4. Such properties are collected in the following theorems.

Theorem 1 Mutation operation do not affect other loci.

Proof Mutation operation works bit-wise.

Theorem 2 Crossover operation do not change the number of necessary mutations in the patterns of mutation.

Proof Crossover operation cannot change the type of a locus.

Theorem 3 Crossover operations always costs lower than mutation operations.

Proof One crossover operation can match two bits at a time.

2.4.2 SPL for only mutation

Now, the SPL of the GA network with only mutations is derived. To do this, let us consider a concrete example shown in Fig. 6 where $L = 3$. The SPL in this case is 0 if making an appropriate combination of pairs: $\{g_{o1}\ g_{d2}\}$ and $\{g_{o2}\ g_{d1}\}$. In this manner, optimal way of "pairing" genes can be found. In this example,

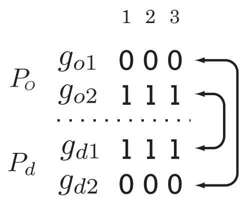


Figure 6. The SPL is zero if taking the good pair.

the SPL is zero if an appropriate combination of pairs is taken, but there are better combinations even when the SPL is more than one. Hence, there are two problems of deriving the SPL: First, finding an optimal pairing for arbitrary two populations, and second, calculating the SPL for appropriately paired set of populations.

The key to solving these problems is to focus on the patterns of loci, especially T_4 and T_4'. On the patterns of mutation, T_1, T_2 and T_3, how to make pairs does not affect the SPL since the necessary number of mutations is constant to any way of pairings, 0 for T_1 and so on. Now, in a pair of populations, let the numbers of T_1, T_2, T_3, T_4 and T_4' be denoted by l_1, l_2, l_3, l_4 and l_4' respectively, then we have

$$\nu_1 = 0 \cdot l_1 + l_2 + 2l_3 = l_2 + 2l_3. \tag{3}$$

where ν_1 is the necessary number of mutations for T_1, T_2 and T_3.

On the other hand, on the patterns of crossover, the number of necessary mutations differs to the way of pairings. Now, we denote the way of pairings such as $\{g_{o1}\ g_{d1}\}$ and $\{g_{o1}\ g_{d2}\}$ as $P1$, and denote the way of pairings such as $\{g_{o1}\ g_{d2}\}$ and $\{g_{o2}\ g_{d1}\}$ as $P2$. Then, the number of necessary mutations of T_4 is 0 when $P1$ but is 2 when $P2$. T_4' is vice versa. So the SPL for T_4 and T_4' is

$$s_2 = 2\min\{l_4, l_4'\} \tag{4}$$

That is, if the number of T_4 is greater than that of T'_4, the pairing $P1$ is better, and if not, the pairing $P2$ is better.

Consequently, the SPL of the network with only mutation is

$$\begin{aligned} \nu &= \nu_1 + \nu_2 \\ &= l_2 + 2l_3 + 2\min\{l_4, l'_4\}. \end{aligned} \tag{5}$$

2.4.3 CPL for only mutation

Next, we calculate the expectation of the SPL which is expressed as

$$\begin{aligned} \mathbb{E}[\nu] &= \mathbb{E}[\nu_1] + \mathbb{E}[\nu_2] \\ &= \mathbb{E}[l_2] + 2\mathbb{E}[l_3] + 2\mathbb{E}[\min\{l_4, l'_4\}], \end{aligned} \tag{6}$$

where $\mathbb{E}[\cdot]$ denote the expectation on all the pairs of the populations. The first two arguments in (6) are very simple because the expectation of the length of a pattern of locus in L loci is $L/16$; T_2 has 8 patterns and T_3 has 2 patterns, so

$$\begin{aligned} \mathbb{E}[\nu_1] &= \mathbb{E}[l_2] + 2\mathbb{E}[l_3] \\ &= \left(1 \cdot \frac{8}{16} + 2 \cdot \frac{2}{16}\right) L = \frac{3}{4}L. \end{aligned} \tag{7}$$

To calculate $\mathbb{E}[\min\{l_4, l'_4\}]$, let us think about a Bernoulli trial with L trials and success probability $p = 1/8$. Then we can assume l_4 and l'_4 as random variables of the success times, which follow the same binomial distribution $\mathcal{B}(L, p)$. Furthermore, we can approximate the binomial distribution $\mathcal{B}(L, p)$ with the normal distribution $\mathcal{N}(Lp, Lp(1-p))$ if L is rather large. So we can say that both l_4 and l'_4 follow the normal distribution

$$\mathcal{N}\left(\frac{L}{8}, \frac{7L}{64}\right) \tag{8}$$

approximately. Now we write $\mu = L/8$ and $\sigma^2 = 7L/64$, then the cumulative distribution function of the random variable $Z = \min\{l_4, l'_4\}$ is

$$\text{Prob}(Z \leq z) = 1 - \prod_{i=1}^{2} \text{Prob}(l_4 > z),$$

so the probability distribution function is

$$\begin{aligned}
\mathrm{Prob}(Z = z) &= \frac{\mathrm{d}}{\mathrm{d}z}\mathrm{Prob}(Z \geq z) \\
&= \frac{\mathrm{d}}{\mathrm{d}z}\prod_{i=1}^{2}\int_z^{\infty}\frac{1}{\sqrt{2\pi\sigma_y^2}}e^{-\frac{(y-\mu_y)^2}{2\sigma_y^2}}\mathrm{d}y \\
&= \frac{\mathrm{d}}{\mathrm{d}z}\left(\int_z^{\infty}\frac{1}{\sqrt{2\pi\sigma_y^2}}e^{-\frac{(y-\mu_y)^2}{2\sigma_y^2}}\mathrm{d}y\right)^2 \\
&= \frac{1}{\sqrt{2\pi\sigma_y^2}}e^{-\frac{(z-\mu_y)^2}{2\sigma_y^2}}\tilde{\Phi}\left(\frac{z-\mu_y}{\sqrt{2\sigma_y^2}}\right),
\end{aligned}$$

where $\tilde{\Phi}(z)$ it the complementary error function defined as

$$\tilde{\Phi}(z) = \frac{2}{\sqrt{\pi}}\int_z^{\infty}e^{-t^2}\mathrm{d}t.$$

Now we have

$$\mathbb{E}[\min\{l_4, l'_4\}] = \mathbb{E}[Z], \tag{9}$$

and from (6), (7) and (9), we have the CPL

$$\mathbb{E}[s] = \frac{3}{4}L + 2\mathbb{E}[Z]. \tag{10}$$

2.4.4 SPL for both operations

In this section, we think about the network which has both edges from mutation and crossover. The SPL $\tilde{\nu}$ is the sum of the minimum necessary number of mutations in patterns of mutation, denoted by $\tilde{\nu}_1$ and that of crossover, denoted by $\tilde{\nu}_2$. We have

$$\tilde{\nu}_1 = l_2 + 2l_3 \tag{11}$$

from (3). $\tilde{\nu}_2$ can be calculated using the following procedure, as shown in Fig. 7.

1. Extract all the patterns of crossover from L patterns and denote it by $\mathbf{T}_4$, $\mathbf{T}_4 = \{T_4T'_4T_4T_4T'_4\}$ for example. Now the length of $\mathbf{T}_4$ is $l_4 + l'_4$ and for convenience, let T_4 and T'_4 be denoted by $\hat{0}$ and $\hat{1}$ respectively.

2. Take $(l_4 + l'_4 - 1)$-bit XOR bit-wise between $\mathbf{T}_4$ and 1-bit shifted $\mathbf{T}_4$ and denote it by $\mathbf{X}_{od}$.

$$\boxed{T_4 \to \hat{0} \quad T_4' \to \hat{1}}$$

$$\begin{array}{rcl} \hat{0}\hat{0}\hat{1}\hat{0}\hat{1}\hat{1}\hat{0}\phantom{\hat{0}} & \leftarrow & \{T_4T_4T_4'T_4T_4'T_4'T_4\} \\ \text{xor} \quad \underline{\phantom{\hat{0}}\hat{0}\hat{0}\hat{1}\hat{0}\hat{1}\hat{1}\hat{0}} & \leftarrow & \text{1 bit shift} \\ \phantom{\hat{0}}\hat{0}\hat{1}\hat{1}\hat{1}\hat{0}\hat{1}\phantom{\hat{0}} & \leftarrow & \mathbf{X}_{od} \end{array}$$

Figure 7. How to calculate the SPL of two nodes consisting of the loci of Type 4

3. Count the number of 1s in $\mathbf{X}_{od}$.

For example, when $\mathbf{T}_4 = \{T_4T_4T_4'T_4T_4'T_4'T_4\}$, $l_4 = 7$ and $\tilde{\nu}_2 = 3$.

Finally, the SPL is expressed as follows,

$$\tilde{\nu} = \tilde{\nu}_1 + \tilde{\nu}_2. \tag{12}$$

2.4.5 CPL for both operations

The CPL $\tilde{\tilde{\nu}}$ is the expectation of (12) about all the pairs of populations.

$$\begin{aligned} \tilde{\tilde{\nu}} &= \mathbb{E}[\tilde{\nu}_1 + \tilde{\nu}_2] \qquad (13) \\ &= \mathbb{E}[\tilde{\nu}_1] + \mathbb{E}[\tilde{\nu}_2], \qquad (14) \end{aligned}$$

where $\mathbb{E}[\cdot]$ is the expectation.

$\mathbb{E}[\tilde{\nu}_1]$ is equal to $\mathbb{E}[\nu_1]$, so we have

$$\mathbb{E}[\tilde{\nu}_1] = \frac{3}{4}L \tag{15}$$

from (7).

We can also calculate $\mathbb{E}[\tilde{\nu}_2]$ by considering all the pairs. See Fig. 7 again. First, the expectation of l_4 is $L/4$ since the number of T_4 and T_4' is 4 out of 16 patterns of loci, as shown in Fig. 5. The length of $\mathbf{X}_{od}$, denoted l_x, cannot be defined when $l_4 = 1$ and is $l_4 - 1$ otherwise. So,

$$\mathbb{E}[l_x] = \frac{L}{4} - \left\{1 - \left(\frac{3}{4}\right)^L\right\}. \tag{16}$$

Then, when we sum up the number of $\hat{0}$ and that of $\hat{1}$ in $\mathbf{X}_{od}$ for all the pairs, they are exactly equal because the number of 0 and 1 are equal. That is, the half of the number of T_4 and T_4' is the number of crossovers. So, we have

$$\mathbb{E}[\tilde{\nu}_2] = \frac{1}{2}\mathbb{E}[l_x] = \frac{L}{8} - \frac{1}{2}\left\{1 - \left(\frac{3}{4}\right)^L\right\}. \tag{17}$$

Finally, from (14), (15) and (17), we have the CPL

$$\mathbb{E}[\tilde{\nu}] = \frac{7}{8}L - \frac{1}{2}\left\{1 - \left(\frac{3}{4}\right)^L\right\}. \tag{18}$$

2.4.6 Discussion

In Fig. 8, the CPLs calculated above are shown. The line "no combination" shows the expectation of the hamming distance between $2L$-bit binary strings, that is L. The CPL of the network with only mutation operations is approximately expressed as (10), because $\mathbb{E}[Z]$ and $\tilde{\nu}$ is intractable. Instead, the following can be used:

$$\frac{7}{8} < \frac{\tilde{\nu}}{\nu} < 1. \tag{19}$$

Since ν is smaller than L, the expectation of the hamming distance between $2L$-bits and $\tilde{\nu}$ is exactly expressed as (18).

This case is the same as the case in which all the weights are assumed to be 1. To improve this approach, probabilistic weights are introduced to the edges in the next section.

2.5 Edge Weights Expressing Transition Probability

2.5.1 Weight on an edge and transition probability

So far, we saw how crossover shortens the CPL by simply counting the necessary operations from the initial population to the other population. This means that each edges has the same weight value 1 if individuals on the edge are connected with either operations. However, each operator in GAs have probability which represents the possibility of the transition. So we introduce a weight expressing transition probability to each edges. If two populations are connected with

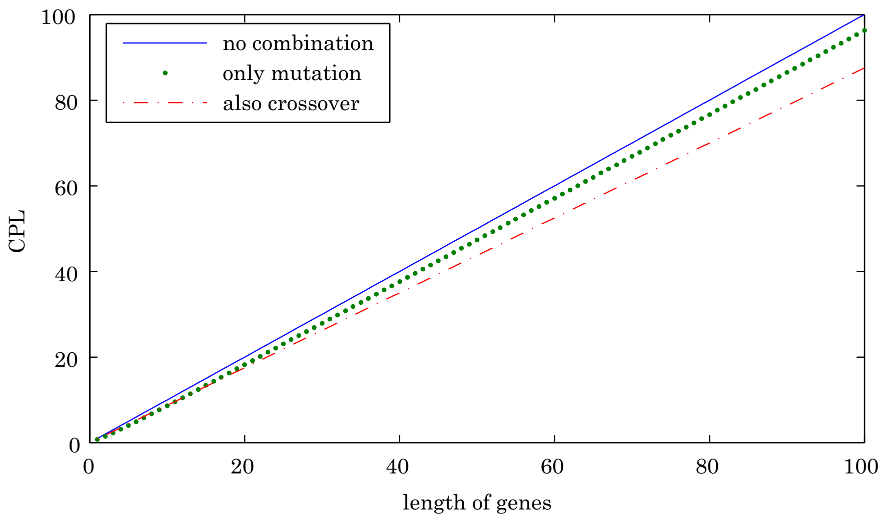


Figure 8. The CPLs when $n = 2$

probability p, it is natural in this study to define the weight of their edge as

$$w = -\log_2 p, \tag{20}$$

since the sum of the weights corresponds to the negative log of the probability of state-transition from an end-node to the other.

For convenience, we assume that a population changes by mutation with probability μ or by crossover with probability $1 - \mu$. This means that the weight of edges from of mutation and crossover are expressed as

$$w_m(\mu) = -\log_2 \frac{\mu}{2L} \tag{21}$$

$$w_c(\mu, k) = -\log_2 \frac{k(1-\mu)}{L-1} \tag{22}$$

respectively, since each population has $2L$ bits and has $L - 1$ crossover points, where k is the number of loci of type T_1, T_2 and T_3 which exists between loci of type T_4, T_4': Not one crossover points change a population to a certain population. Figure 9 show the case of $k = 7$.

2.5.2 Crossover is not always stronger than mutation

Introducing probabilistic weights to the edges in a GA network makes Theorem 3 invalid. That is, there exists such μ as crossover is not more efficient than

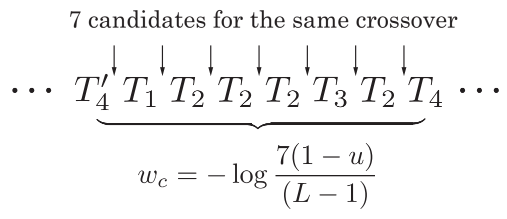

7 candidates for the same crossover
$\cdots\ T_4'\ T_1\ T_2\ T_2\ T_2\ T_3\ T_2\ T_4\ \cdots$
$w_c = -\log \frac{7(1-u)}{(L-1)}$

Figure 9. There are several "same" crossover points.

mutation. So, at what μ mutation beats crossover? In fact, when the probability of one mutation is same as the probability of one crossover, that is

$$\frac{\mu_1}{2L} = \frac{k(1-\mu_1)}{L-1} \quad \Leftrightarrow \quad \mu_1 = \frac{2L}{4L-1} \simeq \frac{2}{3}, \tag{23}$$

where $L \gg 1$, then the cost to match the two populations by mutation is the same as that by crossover in some cases such as $\{T_4 T_4' T_4\}$. and the condition (23) is satisfied, the cost to match the two populations are same whether we use crossover or mutation.

Let us consider the case of a larger μ, where two mutations becomes the same cost as one crossover. This leads a more general condition

$$\left(\frac{\tilde{\mu}}{2L}\right)^2 = \frac{k(1-\tilde{\mu})}{L-1} \quad \Leftrightarrow \quad \tilde{\mu} \simeq 1. \tag{24}$$

where k is the number of mutations which is equal to one crossover in terms of transition probability. Actually we can ignore the case of $k \geq 2$ when $L \gg 1$.

2.5.3 Derivation of the CPL

Firstly, we consider the case that crossover is always stronger than mutation, that is $0 < \mu \leq \mu_1$, then derive the mean SPL between them. The CPL with mutation can be derived in the same way as 7,

$$\mathbb{E}[\hat{a}_{od}] = -\frac{3}{4} L \log \frac{\mu}{2L}. \tag{25}$$

The CPL with crossover is also derived by considering CPL at each k. If we select a link of crossover randomly, the probabilities of $k = \kappa$ at that point is

$\frac{1}{4}\left(\frac{3}{4}\right)^{\kappa}$, so the expectation of weight of crossover links is calculated as follows,

$$\begin{aligned}\mathbb{E}[w_c(\mu,k)] &= -\frac{1}{4}\sum_{k=0}^{L}\left(\frac{3}{4}\right)^{k}\log\frac{k(1-\mu)}{L-1}\\ &= -\frac{1}{4}\log\frac{(1-\mu)}{L-1}\sum_{k=0}^{L}\left(\frac{3}{4}\right)^{k}-\frac{1}{4}\sum_{k=0}^{L}\left(\frac{3}{4}\right)^{k}\log k\\ &\simeq -\log\frac{(1-\mu)}{L-1}-\gamma(L), \qquad (26)\end{aligned}$$

where

$$\gamma(L)\equiv\frac{1}{4}\sum_{k=0}^{L}\left(\frac{3}{4}\right)^{k}\log k \qquad (27)$$

and γ is not dependant on μ. So by substituting weights, that is 1, in (18) with (26) and disregard the small arguments, we get

$$\mathbb{E}[\tilde{a}_{od}]\simeq\frac{L}{8}\left\{-\log\frac{(1-\mu)}{L-1}-\gamma(L)\right\}. \qquad (28)$$

Finally, with (25) and (28), We get

$$\begin{aligned}C_{c1}(\mu) &\simeq \mathbb{E}[\hat{a}_{od}]+\mathbb{E}[\tilde{a}_{od}]\\ &= L\left(-\frac{3}{4}\log\frac{\mu}{2L}-\frac{1}{8}\log\frac{1-u}{L-1}\right)-\frac{L}{8}\gamma(L), \qquad (29)\end{aligned}$$

where $C_{c1}(\mu)$ is the CPL on $0\le\mu\le\mu_1$.

Next, we examine $\mu_1<\mu\le 1$. When increasing μ from $\mu=\mu_1$, crossovers are taken place by mutations in ascending order of k. At the situation $\mu=\mu_1$, one crossover has same weight as one mutation; $\{T_4(T_3T_2)T_4'(T_3T_2)T_4\}$ is a good example for this case. In addition, we assume that all crossovers are taken place by mutations at $\mu_k>\mu_1$. This assumption do not change the CPL significantly since the number of crossovers as $k=\kappa$ decays exponentially as k increases.

Hence it comes the matter how many crossovers are successive in the populations. For example, loci $\{T_4(T_1T_0)T_4'(T_3T_2)T_4\}$ have two successive crossovers. We can calculate the number of such successive crossovers in a similar way to calculate the number of crossovers. That is, we get the number by taking $(l-2)$-bit XOR between X_{od} and 1-bit shifted X_{od} and count the number of 0 (Fig. 10). Note that such loci as $\{T_4T_4'T_4T_4'\}$ has three successive crossovers, but only two

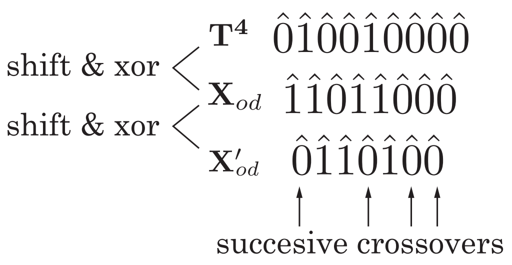

shift & xor
shift & xor
$\mathbf{T^4}$ $\hat{0}\hat{1}\hat{0}\hat{0}\hat{1}\hat{0}\hat{0}\hat{0}\hat{0}$
$\mathbf{X}_{od}$ $\hat{1}\hat{1}\hat{0}\hat{1}\hat{1}\hat{0}\hat{0}\hat{0}$
$\mathbf{X}'_{od}$ $\hat{0}\hat{1}\hat{1}\hat{0}\hat{1}\hat{0}\hat{0}$
succesive crossovers

Figure 10. How to calculate the number of successive crossovers

right or left crossovers can be taken place by four mutations because remaining part can't be matched by only one mutation. In other words, only even number of crossovers can be taken place by mutations. Additionally, four or more successive crossovers are rare compared to only two successive crossovers, so we may think about only two successive crossovers. Approximately, the number of only two successive crossovers are expected to be approximately half part of all crossovers since the number of T_4 and the number of T'_4 are the same. So the CPLs on $\mu_1 < \mu \leq 1$ are calculated as follows,

$$\begin{aligned} C_{c2}(\mu) &\simeq -L\left(\frac{3}{4}+\frac{1}{8}\cdot\frac{1}{2}\right)\log\frac{\mu}{2L} \\ &\quad -L\left(\frac{1}{8}\cdot\frac{1}{2}\right)\log\frac{1-u}{L-1}-\frac{L}{8}\gamma(L). \end{aligned} \tag{30}$$

Note that when $\mu = 1$, all crossovers are taken place by mutations and $C_m(1) = C_c(1)$ is satisfied. We can see apparently

$$C_c(1) = L\log 2L. \tag{31}$$

2.5.4 Discussion

Fig. 11 shows $C_c(\mu)$ and $C_m(\mu)$ vs. μ when $L = 50$. We can see that the CPL of the network of only mutation and that of the network of also crossover have the same value when $\mu = 1$ and $C_{c1}(\mu) = C_m$ when $\mu = 0.41$. $\tilde{\mu}$ is 27/28, which is given by differentiating (30) with μ and setting the answer to 0. The CPLs are monotone decreasing for almost all of μ, the probability of mutations. In other words, the crossover operation make the CPL longer in the network to which the weight introduced.

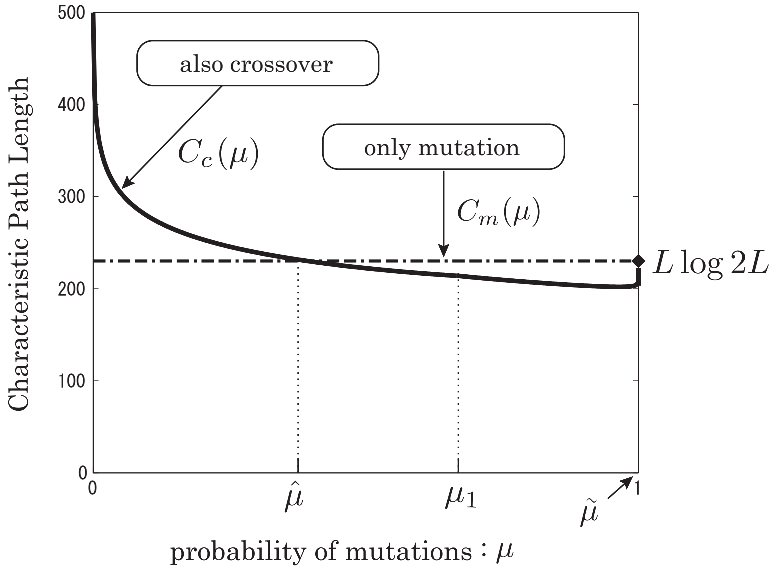


Figure 11. μ vs. $C(\mu)$; $L = 50$

2.6 Conclusions

Our main contribution of this study is the derivation of the CPLs of a GA network of $N = 2$ under some diffierent conditions. When all the weights of edges are equal to 1, the crossover operation surely shortens the CPL linearly with the length of the individuals. But when each edge has the weight expressing transition probability, the crossover operation does not necessarily shortens the CPL. Instead, the CPL are monotonically decreasing for $\mu \leq 27/28$.

It is obvious that the GA does not necessarily work best at $\mu = 27/28$. This means that the effectiveness of searching does not seem to directly correspond to the CPLs. In summary, we found out the simple mathematical relationship between the CPL and the genetic operations. In future work, we consider in case of $n > 2$. and find out better criteria for further analysis of GAs.

3. Statistical properties of support vector machines with forgetting factor

3.1 Introduction

A support vector machine (SVM) is one of the most successful learning classifiers in decades [8–11]. Since the original SVM treats all the examples equally, it cannot be applied to time-varying environments. Furthermore, since training an SVM requires a quadratic programming (QP) problem with a number of coefficients equal to the number of training examples, standard QP techniques become infeasible for large data sets.

One method to cope with these difficulties is an on-line approach, where some of the training examples are discarded [12] [13] [22] [23]. In this approach we can only decide the number of examples (by using heuristics) but cannot freely set the weight of each example. In other words, the weights of examples can only assume values of zero or one.

We can relax this regulation by introducing a forgetting factor [14] like in RLS algorithms for adaptive filters [15], which naturally assigns non-integer weights to examples, to select the filter coefficients. Although many variants of forgetting factors for an RLS algorithm have been proposed [16], they fundamentally use an exponentially decaying forgetting factor, that is, each example has λ as its weight, where λ $(0 < \lambda < 1)$ denotes a forgetting factor and $t \in \mathbb{N}$ denotes the time when they are generated. We named this an *exponential forgetting factor* (EFF) and analyzed it in a simple case in the previous work [21].

We first chose an exponential forgetting factor simply because it is often used in the signal-processing literature. It did work in terms of adaptation to time-varying problems in a simple case, but the asymptotic averaged generalization error (AGE) did not converge to zero. We can choose an EFF so that the averaged generalization error is arbitrarily small, but it is still better to consider a forgetting factor to assure convergence. Hence, we introduce a *factorial forgetting factor* (FFF) to the same problem and prove that its averaged generalization error goes to zero as the number of examples increases to infinity in the same simple case as was considered in the previous work [21]. The outline of this paper is the following:

In section **3.2**, we formulate the SVM with forgetting factor (SVM-FF) for both an EFF and FFF in detail. This algorithm, based on the ν-SVM [17], has a good geometrical property with which we can analyze a simple problem introduced in section **3.3**. We derive the generalization performance of the algorithm in the asymptotic state in section **3.4**. The validation with computer simulations is presented in section **3.6**. Finally, we present conclusions and discussions in section **4.7**.

3.2 SVM with Forgetting Factor

3.2.1 Exponential Forgetting Factor (EFF)

An SVM is originally formulated as a learning machine which maps input vectors to the corresponding feature vectors and classifies them with the optimal hyperplane in terms of margin [18]. However, we employ the linear kernel (which has the feature space identical to the input space) and the ν-SVM with homogeneous hyperplanes (which has a little different criterion from the margin-maximization). These make the analysis of the algorithm possible as was done in [19] for analyzing the soft-margin effect.

Given a set of N examples $(\boldsymbol{x}_i, y_i)$, $i = 1, 2, \ldots, N$, where $\boldsymbol{x}_i$ is an input and y_i is the corresponding label, the ν-SVM with homogeneous hyperplanes is formulated as

$$\begin{aligned} &\min_{\boldsymbol{w},\xi_i,\beta} \frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{i=1}^{N}\xi_i - \beta \\ &\text{s.t. } \boldsymbol{w}'\boldsymbol{f}_i \geq \beta - \xi_i, \quad \xi_i \geq 0, \end{aligned} \tag{32}$$

where $\boldsymbol{w}$ is the normal vector of a hyperplane $\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x} = 0$, ξ_i are slack variables for soft margins, C is a constant determining the softness and $\boldsymbol{f}_i \equiv \boldsymbol{x}_i y_i$ [17, 19].

In the RLS algorithm for adaptive filters, the squared error at each time decays exponentially, that is, the sum of the squared errors at time N is defined as

$$\sum_{i=0}^{N} \lambda^{N-i} e_i^2, \tag{33}$$

where e_i denotes the error of the example i and $0 < \lambda < 1$ is a constant called the forgetting factor. We introduce this idea to the ν-SVM and give an exponentially decaying weight to the slack variables. Then, the SVM with forgetting factor (SVM-FF) is formulated as

$$\begin{aligned} &\min_{\boldsymbol{w},\xi_i,\beta} \frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{i=1}^{N} \lambda^{N-i}\xi_i - \beta \\ &\text{s.t. } \boldsymbol{w}'\boldsymbol{f}_i \geq \beta - \xi_i, \quad \xi_i \geq 0. \end{aligned} \tag{34}$$

The dual problem of (34) is easily derived as

$$\begin{aligned} &\min_{\alpha_i} \frac{1}{2}\|\boldsymbol{w}\|^2 \\ &\text{s.t. } \boldsymbol{w} = \sum_{i=1}^{N} \alpha_i \boldsymbol{f}_i, 0 \leq \alpha_i \leq C\lambda^{N-i}, \sum_{i=1}^{N} \alpha_i = 1, \end{aligned} \tag{35}$$

where α_i are the Lagrange multipliers.

Note that the cost function of RLS-SVM algorithm [14] is

$$\min_{\boldsymbol{w},\xi_i,\beta} \frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{i=1}^{N} \lambda^{N-i}\xi_i^2, \tag{36}$$

which has the squared slack variables. This reduces the problem to a linear equation, not QP. Our FF can be introduced to the original SVM too in the same manner.

3.2.2 Factorial Forgetting Factor

The asymptotic convergence of AGE cannot be assured for even a linear separable problem. For that reason, we replace the EFF with a *factorial forgetting factor* (FFF), $1/(N-i+1)$, so that the square sum of the FF diverges and the simple sum of the FF converges. In a similar way as the EFF's case, we can derive the dual problem as

$$\begin{aligned} &\min_{\alpha_i} \frac{1}{2}\|\boldsymbol{w}\|^2 \\ &\text{s.t. } \boldsymbol{w} = \sum_{i=1}^{N} \alpha_i \boldsymbol{f}_i, 0 \leq \alpha_i \leq \frac{C}{(N-i+1)}, \sum_{i=1}^{N} \alpha_i = 1. \end{aligned} \tag{37}$$

3.3 Problem Statement

3.3.1 Geometry of SVM-FF

The cost function to be minimized in (35) is the norm of the weight vector $\boldsymbol{w}$. Hence, let us consider the constraints $\boldsymbol{w}$ should satisfy. The minimum convex set containing all the given points is called the convex hull of the points. The conditions that any α_i is nonnegative and that their sum is unity lead to the fact that the weight vector $\boldsymbol{w}$ is included in the convex hull of $\{\boldsymbol{f}_i\}_{i=1}^{N}$. The additional condition that each α_i has an upper bound $C/(N-i+1)$ reduces the area where $\boldsymbol{w}$ can exist. We call the area determined by (35) the reduced convex hull.

Bennett and Bredensteiner [20] first pointed out the relationship between the ν-SVM solution and the reduced convex hull (RCH), where two RCHs appear since they considered inhomogeneous separating hyperplanes, as simply shown in Fig. 13 where two convex hulls are just lines. Ikeda and Aoishi [19] added the assumption of homogeneous hyperplanes and showed that the ν-SVM solution is the point nearest to the origin in the RCH. Thanks to this simple geometrical picture, the effect of the soft margin parameter was quantitatively elucidated, where the upper bound of any α_i is C (Fig. 12).

In the case of SVM-FF, each α_i has different upper bounds $C\lambda^{N-i}$ for each i. However, the method for deriving the AGE in [19] can also be applied to this case as will be presented in the next subsection.

3.3.2 Problem Formulation

Now we analyze the SVM-FF by using a simple problem, which is almost the same one described in [19]. We assume that N inputs, $\boldsymbol{x}_i$, $i = 1, \ldots, N$, with class labels, $y_i \in \{-1, +1\}$, are independently generated from two classes (a positive one and a negative one) with equal probability, which are linearly separable by a hyperplane $\boldsymbol{x}^{\mathrm{T}}\boldsymbol{w} + b = 0$. We also assume that the distribution of each class is arbitrary except that two classes are uniformly and identically distributed around the separating hyperplane. Fig. 13 shows a one-dimensional problem linearly separable at the origin. Homogeneous re-notation, $\boldsymbol{x}_i = (\boldsymbol{x}_i^{\mathrm{T}}, 1)^{\mathrm{T}}, \boldsymbol{w}_i = (\boldsymbol{w}_i^{\mathrm{T}}, b)^{\mathrm{T}}$ is convenient from a geometrical and mathematical point of view, which is referred to as “lifting-up” (Fig. 14) [19].

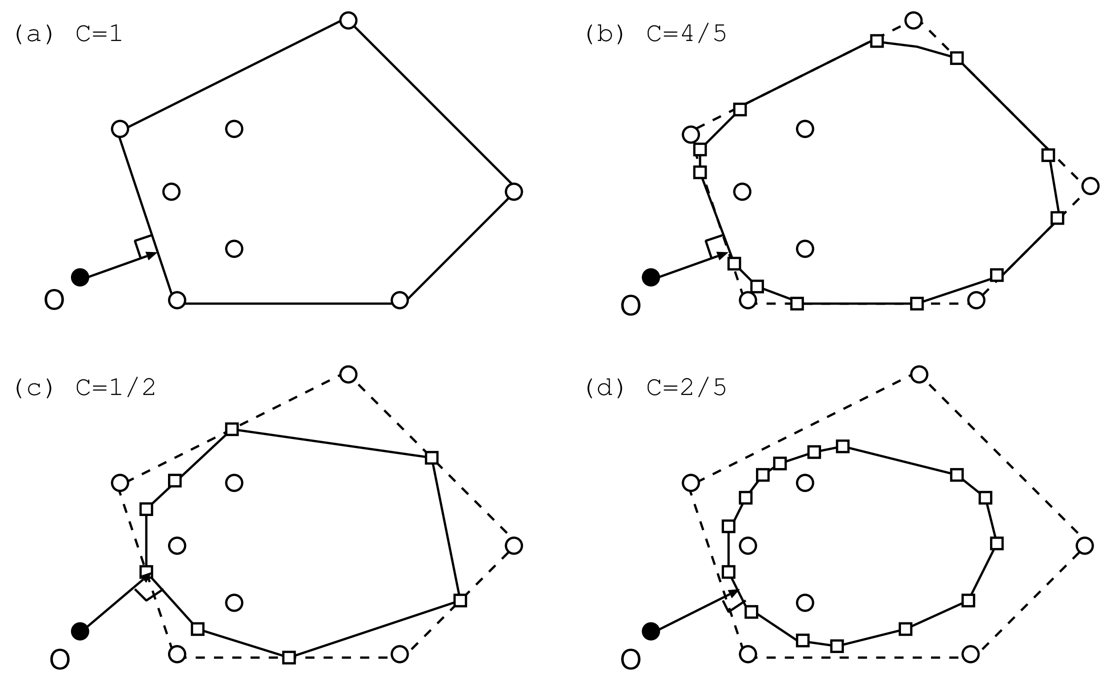


Figure 12. Points closest to the origin O of the convex hull (the dashed line) and the reduced convex hull (the solid line, (a)$C = 1$, (b)$C = 4/5$, (c) $C = 1/2$, (d)$C = 2/5$) of examples are shown as $\circ$.

Because $\boldsymbol{f}_i = \boldsymbol{x}_i y_i$, the $\boldsymbol{f}_i$ are all in the positive side (Fig. 15). The $\boldsymbol{f}_i$ form a convex hull in the case of hard margin, whereas they form a reduced convex hull in the case of soft margin and forgetting factor. Geometrically, it is obvious that the generalization error of the SVM appears $\|\theta_\epsilon\|$ in the right figure of the Fig. 3.3.2, assuming that the generalization error is small. Hence, we can calculate the generalization error by deriving the distribution of θ.

Furthermore, we modify the problem utilizing sparseness. Since the solution of an SVM is sparse, the examples far from the separating hyperplane do not correspond to solutions with high probability. Hence we map both distributions to a uniformly distributed semicircle (Fig. 3.3.2). Now the problem has been modified as follows: We assume that N inputs, $\boldsymbol{x}_i$, $i = 1, \ldots, N$, are independently uniformly chosen from S^m, where S^m is an m-dimensional unit sphere. Then, the

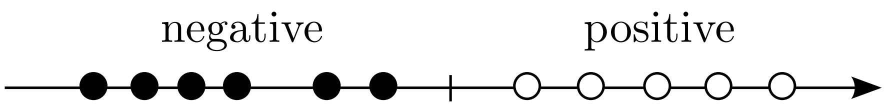


Figure 13. Simple one-dimensional problem which is linearly separable at the origin. There are two "convex hulls" (two lines) and the hyperplane is just a point on the origin.

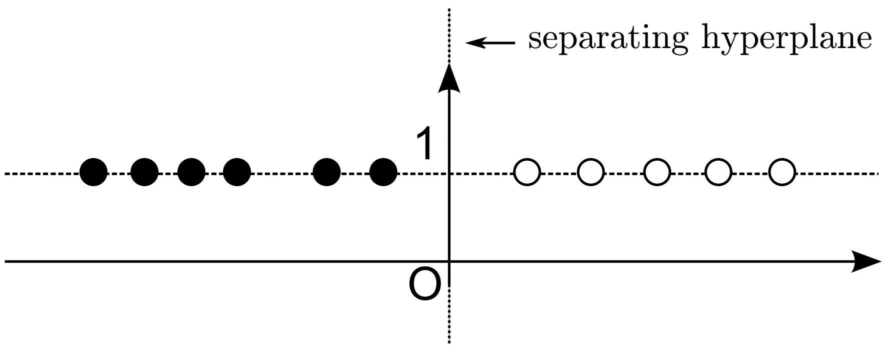


Figure 14. Lifting-up: Re-notated $\boldsymbol{x}_i$ are displayed homogeneously in the two-dimentional plane.

vectors $\boldsymbol{f}_i = y_i\boldsymbol{x}_i$, $i = 1, \ldots, N$, are uniformly distributed in S_+^m where:

$$y_i = \mathrm{sgn}(\boldsymbol{w}^{*'}, \boldsymbol{x}_i), \tag{38}$$

$$\mathrm{sgn}(s) = \begin{cases} +1, & \text{if } \ s \geq 0, \\ -1, & \text{otherwise}, \end{cases} \tag{39}$$

$$S_+^m = \{\boldsymbol{f} | \boldsymbol{w}^{*'}\boldsymbol{f} \geq 0, \boldsymbol{f} \in S^m\}. \tag{40}$$

In this case, the probability that an estimate $\hat{\boldsymbol{w}}$ mis-predicts the output of a new input $\mathbf{x}$, which is called the generalization error or the prediction error, is written as θ/π, where θ is the angle between $\hat{\boldsymbol{w}}$ and $\boldsymbol{w}^*$. The average generalization error is defined as the probability that an estimate $\hat{\boldsymbol{w}}$ mis-predicts the output of a new input, averaged over given examples. In the following sections, we derive the average generalization error of the SVM with FF for the one-dimensional case ($m = 1$) in the asymptotic limit of $N \to \infty$.

In the case of one-dimensional space, the nearest point to the origin in the

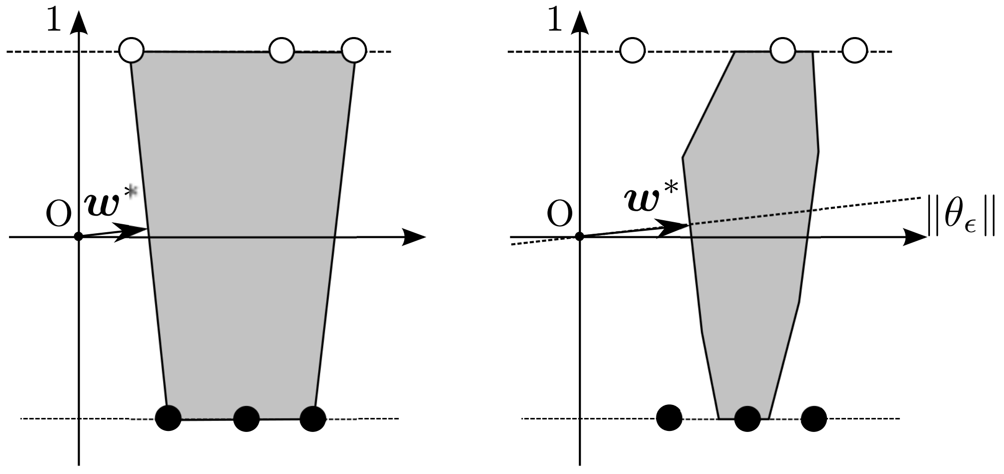


Figure 15. All $\boldsymbol{f}_i$, shown with $\circ$ or $\bullet$, are in the right-hand side of the plane and form a convex hull with $C = 1$ or a reduced convex hull with $C = 1/3$ (both colored in gray). The QP is equal to find the closest point $\boldsymbol{w}^*$ in the (reduced) convex hull from the origin.

reduced convex hull of the examples is the midpoint of the two points, $\boldsymbol{f}_L$and $\boldsymbol{f}_R$, each of which is the nearest to the horizontal axis. If we know the distributions of the angles θ_L and θ_R of $\boldsymbol{f}_L$and $\boldsymbol{f}_R$, we can derive the average generalization error since the SVM solution $\hat{\boldsymbol{w}}$ is written as $\hat{\boldsymbol{w}} = (\boldsymbol{f}_L + \boldsymbol{f}_R)/2$. Then, the average generalization error is written as the average of $\theta_\varepsilon/\pi = |\theta_L - \theta_R|/2\pi$. We can assume that the solutions, θ_L and θ_R, follow the same distribution because the examples are uniformly distributed. Hence, we consider only the left half of the semicircle (the second quadrant in Fig. 17). Here, let us denote by θ_i ($i = 1, \ldots, N$) the angle with the ith nearest example from the left edge named L (Fig. 3.3.2), denote the Lagrangian multipliers for θ_i by α_i and denote forgetting factors by $F_i \in \{\lambda^{N-t_i}, 1/(N-t_i+1)\}$ where $t_i(i = 1, \cdots, N)$ is the time when the i-th example is generated. Then, the solutions in the left half under constraints are obtained by solving the optimization problem:

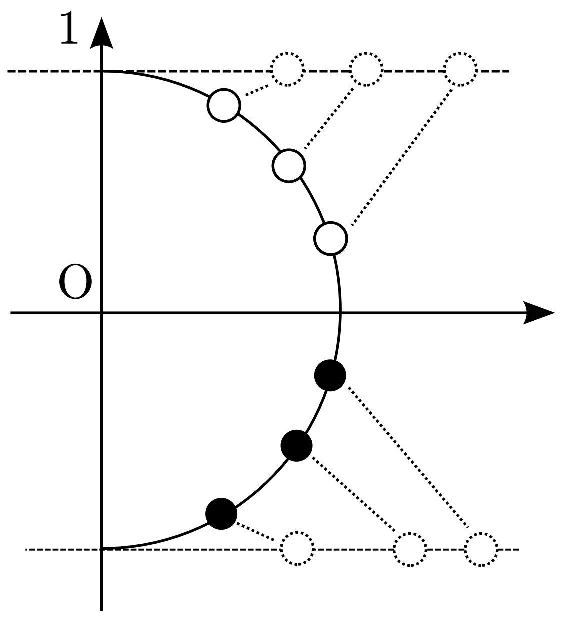


Figure 16. $\boldsymbol{f}_i$ are mapped to the uniform distribution on the semicircle. The semicircle will be rotated for the latter explanation (Fig. 17).

$$\begin{aligned} \min_{\boldsymbol{\alpha}} \quad & \theta_L = \sum_{i=1}^{N} \alpha_i \theta_i & (41) \\ \text{s.t.} \quad & 0 \le \alpha_i \le CF_i, & (42) \\ & \sum_{i=1}^{\ell} \alpha_i = 1, \quad \theta_i \le \theta_{i+1}. & (43) \end{aligned}$$

The solutions of the problem (41) can be written as:

$$\theta_L = \sum_{i=1}^{\ell-1} CF_i \theta_i + (1 - \sum_{i=1}^{\ell-1} CF_i)\theta_\ell, \quad (44)$$

where n is the number of support vectors with non-zero α_i. By solving (41)–(43), we have:

$$\alpha_i = \begin{cases} CF_i, & \text{if } \ 1 \le i \le n-1, \\ 1 - \displaystyle\sum_{i=1}^{n-1} CF_i, & \text{if } \ i = n, \\ 0, & \text{otherwise} \end{cases} \quad (45)$$

This solution is obtained easily because SVs can be taken from the smallest angle since the simple problem is one-dimensional. The examples which have positive α_i $(1 \leq i \leq n)$ as weights are called support vectors (SVs).

3.4 Statistical Properties of SVM-FF

3.4.1 Typical Learning Curve of Different SVMs

Before entering the analysis, we review the typical properties of learning curves of several SVMs. In the original SVM and the ν-SVM, as the number of examples increases, the generalization error decreases and converges to zero in the noise-free case. However, it is not trivial in the case of the SVM-FF. In the case of EFF, the θ_L goes to a certain steady state as a distribution due to the following two equivalent reasons: (1) The number of support vectors increases by the influence of EFF (suppose almost all of α_i are nearly 0 when N is large) and (2) each α_i exponentially decrease as N increases. Hence, the generalization error does not converge to zero. On the contrary, θ_L converges to zero in the FFF case, which means that its learning curve also converges to zero in probability.

In the Fig. 19, we can see that the learning curve of EFF converges to a constant value, depending on the magnitude of the EFF, whereas other learning curves seem to converge to zero. The learning curve of SVMs with FFF is slower than that of EFF, but the both AGEs decrease as the number of examples grows. This property is fairly good, because the learning curve converges to zero without losing the ability of adapting to time-varying problems.

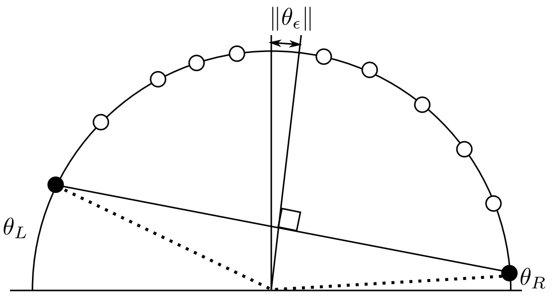
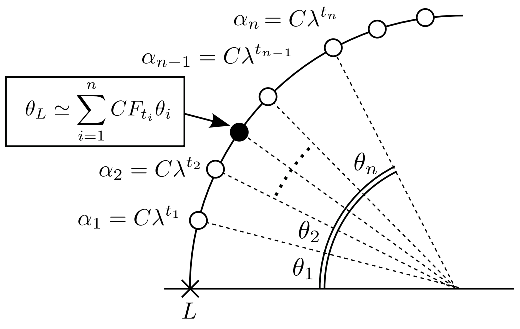


Figure 17. $\boldsymbol{f}_i$'s mapped to the semicircle.



$\alpha_n = C\lambda^{t_n}$
$\alpha_{n-1} = C\lambda^{t_{n-1}}$
$\theta_L \simeq \sum_{i=1}^{n} CF_{t_i}\theta_i$
$\alpha_2 = C\lambda^{t_2}$
$\alpha_1 = C\lambda^{t_1}$
θ_n
θ_2
θ_1
L

Figure 18. The left-hand side of the circle. The θ_i are placed starting from the example which is nearest to the point L. The center of the SVs is denoted as θ_L.

We revise the previous work on the analysis of the SVMs with exponential forgetting factor [21]. In that work, we assumed the independence among the angles of ordered examples, where the angles with higher indexes contained the angles with all smaller indexes. Instead, following the original work [19], we assume that no angles θ_i contain each other, as shown in figure 3.3.2, which makes the analysis more simple and yields better results especially for the case of factorial forgetting factor.

First we derive the probability distribution of the left center of SVs, θ_L, for any size of sample data N. Because the θ_L and θ_R follow the same distribution, then we can straightforwardly derive the AGE of θ. Since the derivation processes of averaged generalization errors for exponential and factorial forgetting factors are similar, we will derive the formula in a generalized manner by denoting both forgetting factors as $F_i \in \{\lambda^{t_i}, 1/t_i\}$ for simplicity. Note that F_i depend on t_i but independent from i. Since the ith smallest angle, denoted by θ_i, is independent of the forgetting factor λ^{t_i}, the mean of the product of the two variables can be written as

$$\mathbb{E}[F_i\theta_i] = \mathbb{E}[F_i]\mathbb{E}[\theta_i]. \tag{46}$$

Hence, the mean of θ_L is derived as

$$\mathbb{E}\left[\theta_L\right] = \mathbb{E}\left[\sum_{i=1}^{n-1} CF_i\right]\mathbb{E}\left[\theta_i\right] + \mathbb{E}\left[(1 - \sum_{i=1}^{n-1} CF_i)\right]\mathbb{E}\left[\theta_n\right]. \tag{47}$$

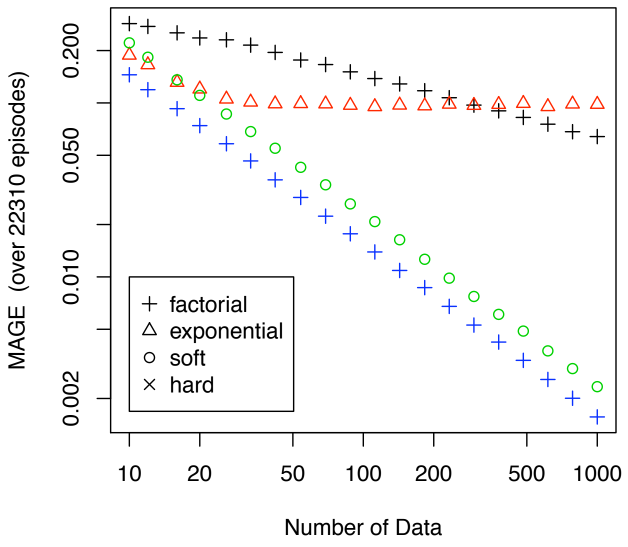


Figure 19. Typical learning curves of four types of SVMs: hard margin SVM, soft margin SVM ($C = 1$), SVM with exponential forgetting factor ($C = 1, \lambda = 0.9$), and SVM with factorial forgetting factor ($C = 1$). The horizontal axes shows the number of samples and the vertical axes shows averaged generalization error over 20000 episodes, each of which is log scaled.

Now let us replace the coefficients of the second term by just F_i since the second term of (47) itself is considered to be small compared to the whole θ_L, so we have:

$$\theta_L \simeq \sum_{i=1}^{n} C F_i \theta_i \tag{48}$$

Furthermore, since $\theta_i = \sum_{j=1}^{i} \eta_j$, the expectation of θ_L is

$$\theta_L = C \sum_{i=1}^{n} \sum_{j=1}^{\ell-i+1} F_j \eta_j, \tag{49}$$

and its variance is

$$
\begin{aligned}
& \mathbb{V}\left[\theta_L\right] && (50)\\
= \;& C^2 \mathbb{V}\left[\sum_{i=1}^{n}\sum_{j=1}^{\ell-i+1} F_j \eta_i\right] && (51)\\
= \;& C^2 \sum_{i=1}^{n}\left(\mathbb{V}\left[\sum_{j=1}^{\ell-i+1} F_j\right] + \mathbb{E}^2\left[\sum_{j=1}^{\ell-i+1} F_j\right]\right) \\
& \times \left(\mathbb{V}\left[\eta_i\right] + \mathbb{E}^2\left[\eta_i\right]\right) && (52)\\
& - \sum_{i=1}^{n}\sum_{j=1}^{\ell-i+1} (\ell-i+1)^2 \mathbb{E}^2\left[F_j\right]\mathbb{E}^2\left[\eta_i\right].
\end{aligned}
$$

For both exponential and factorial forgetting cases, all the SVs are very small compared to π, so that all the η_i follow the identical distribution to that of η_1. The distribution of η_1 can be obtained by considering the probability of all the other samples falling into the area $[\eta_1, \pi]$, which is

$$
\text{Prob}[\Theta \leq \eta_1] = 1 - \left(1 - \frac{\theta_L}{\pi}\right)^N. \quad (53)
$$

Differentiating (53) with respect to θ_L gives the distribution

$$
p(\eta_i) = \frac{N}{\pi}\left(1 - \frac{\theta_L}{\pi}\right)^N, \quad (54)
$$

which is asymptotically the same as the exponential distribution with parameter $\lambda = N/\pi$ for infinitely large N. Hence we have

$$
\begin{aligned}
\mathbb{E}\left[\eta_i\right] &= \frac{\pi}{N} && (55)\\
\mathbb{V}\left[\eta_i\right] &= \frac{\pi^2}{N^2}, && (56)
\end{aligned}
$$

which simplifies (52) to be

$$
\mathbb{V}\left[\theta_L\right] = \frac{C^2 \ell^3 \pi^2}{3N^2}\left(2\mathbb{V}\left[F_j\right] + \mathbb{E}^2\left[F_j\right]\right) \quad (57)
$$

The terms $\mathbb{V}\left[\sum_{j=1}^{\ell-i+1} F_j\right]$ and $\mathbb{E}^2\left[\sum_{j=1}^{\ell-i+1} F_j\right]$ have different values for exponential and factorial cases, so that we handle these differently.

3.4.2 AGE of exponential forgetting factor

We now turn to the calculations of averaged generalization error for each F_{t_i}. In the case of *exponential forgetting factor*, recalling that λ is some constant and t_i are a randomly permutated sequence of $\{1, 2, \cdots, n\}$, the mean and the variance of λ^{t_i} are

$$\begin{aligned}\mathbb{E}\left[F_j\right] = \mathbb{E}[\lambda^{t_i}] &= \frac{1}{N(1-\lambda)} & (58)\\ \mathbb{V}\left[F_j\right] = \mathbb{V}\left[\lambda^{t_i}\right] &= \frac{1}{N}\sum_{i=0}\left(\lambda^{t_i} - \frac{1}{N(1-\lambda)}\right)^2 \\ &\simeq \frac{1}{N}\sum_{i=0}^{N}\lambda^{2t_i} \quad \text{(for large } N\text{)} \\ &= \frac{1}{N(1-\lambda^2)}. & (59)\end{aligned}$$

By substituting (58) and (59) into (57) and considering that the number of SVs, ℓ, is approximated as $N(1-\lambda)$ (or $N(1-\lambda)/2$ for left- and right-hand sides), we have

$$\mathbb{V}\left[\theta_L\right] = \frac{C^2\pi^2(1-\lambda)^2}{24}\left(1 + \frac{2(1-\lambda)}{1-\lambda^2}\right). \tag{60}$$

Here, θ_L and θ_R asymptotically follow a Gaussian distribution because they are the sum of infinite identical random variables. Hence we need only the variances of θ_L and θ_R to obtain the normalized generalization error, which was already calculated in (60). That is:

$$\mathbb{E}[|\theta_e|] = \int_{-\infty}^{\infty} x\, \mathcal{N}(x|0, 2\mathbb{V}\left[\theta_L\right])\mathrm{d}x = \sqrt{\frac{2\mathbb{V}\left[\theta_L\right]}{\pi}}, \tag{61}$$

where θ_e denotes the generalization error and $\mathcal{N}(x|0, 2\mathbb{V}\left[\theta_L\right])$ denotes a Gaussian distribution.

We can derive a concrete form of the AGE for the EFF by substituting (60) into (61), but that is not necessary here. However, we should note that the AGE for the EFF does not converge to zero unless $\lambda = 1$ because (60) shows that the variance of both left and right-hand centers of the SVs do not converge to zero.

3.5 AGE on factorial forgetting factor

In a similar way to the case of the EFF, we can derive the AGE for the *factorial forgetting factor* (FFF) . In the equation (60),

$$\mathbb{E}\left[F_j\right] = \frac{1}{N}\sum_{k=1}^{N}\frac{1}{k} \simeq \frac{\log N}{N} \tag{62}$$

$$\begin{aligned}\mathbb{V}\left[F_j\right] &= \frac{1}{N}\sum_{k=1}^{N}\left(\frac{1}{k}-\frac{\log N}{N}\right)^2 \\ &\simeq \frac{1}{N}\left(1-\frac{(\log N)^2}{N}\right),\end{aligned} \tag{63}$$

with the approximation $\sum_{k=1}^{N} 1/k \simeq \log N$. Now we know the concrete form of the approximated AGE for the FFF, by using (61), (62) and (63),

$$\mathbb{E}\left[|\theta_e|\right] \simeq C\sqrt{\frac{\pi}{4(\log N)^3}\left(s_2 - \frac{(\log N)^2}{N}\right)}, \tag{64}$$

where $s_2 = \sum_{i=1}^{N} 1/i^2$, which converges to a constant value when $N \to \infty$. (64) shows that the order of the AGE follows $O(1/(\log N)^{3/2})$, which is much slower than both hard-margin SVMs and soft-margin SVMs, but it is a fair deficit considering the ability to adapt to time-varying problems.

3.5.1 Other Forgetting Factors

We can consider more generalized types of forgetting factors. As can be seen, the result (57) does not depend on the shapes of forgetting factors, so we can derive AGEs for arbitrary forms of forgetting factors. For example, we can think of $t_i^{-\alpha}$ $(\alpha \geq 1)$ as a generalizalized form of $1/t_i$, whose AGE can be instantly obtained by calculating its expectation and variance and using (61), in the same way as we did in this study. This will accelerate the convergence of the algorithm, but on the other hand it will be less sensitive to variations over time. Practically, one can trade off between the sensitivity to time and the convergence rate.

3.6 Computer Simulations

In order to confirm the validity of the theory derived above, we carried out computer simulations. We employed the one-dimentional simple problem described in section 2, and solved it by using SVMs with the FFF. We created 10842 episodes, in each of which QP problems were solved at 20 points of sample size at equal intervals along a log scale. At last we took the average of all the resulting generalization errors at each horizontal point over all episodes.

The code was written in R, a statistical programming language, with the "kernlab" package which provides the interior point method to solve QP problems.

The theoretical and experimental learning curves are compared in Fig. 20. The circles show the experimental learning curve and the solid line shows the theoretical result derived at (64). The two learning curves match better in the range of larger sample sizes (around 10^4) than in the range of smaller sample sizes (around 10^2). Simulations on over 10^4 are computationally infeasible because solving a QP problem requires polynomial computational time.

3.7 Conclusions and Discussions

An SVM with factorial forgetting factor has been revised so that it has zero generalization error in the asymptotic state. Our asymptotic theory on its averaged generalization error in the simple noiseless case shows that the generalization error of the proposed method goes to zero when the number of examples goes to infinity, differently from an SVM with exponential forgetting factor.

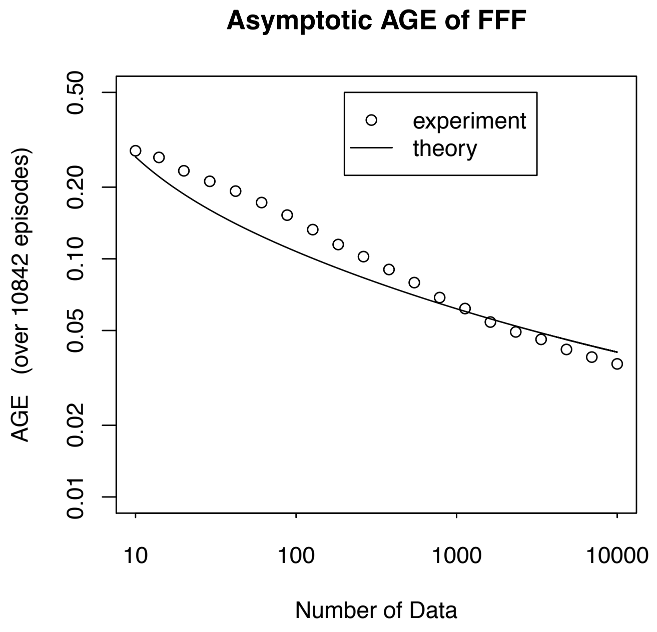


Figure 20. The learning curve of SVMs with factorial forgetting factor. The horizontal and vertical axes show the number of examples and the velocity of the generalization errors, both in log plot, respectively. The sample points are averaged over 10842 episodes generated in a parallel manner.

4. Information geometry in neural spike sequences

4.1 Inter sipike sequences of cortical neurons

The characteristics of neurons in cortical areas have been the subject of recent discussion in the literature. In particular, there has been discussion on the statistical properties of the interspike intervals (ISIs) of spike sequences such as the coefficient of variation, C_V, the skewness coefficient, S_K, the correlation coefficient of consecutive intervals, C_{OR}, and the local variation, L_V [24–28]. In particular, Shinomoto et al. [29] have shown that the local variation with a refractory period, L_{VR}, can almost classify the functions of neurons without any other information.

From the viewpoint of statistical modeling, ISIs can be modeled as a gamma process with a variable rate but a fixed shape parameter [28, 30]. Since gamma distributions form a two-dimensional e-flat manifold S from the information-geometrical viewpoint [31–33], the problem of estimating the shape parameter results in a semiparametric estimation [34, 35]. These theoretical methods also apply to estimate the refractory period in L_{VR}, which has heuristically been determined to date.

The rest of this paper is organized as follows: Some statistical measures of spike sequences and their properties are introduced in Section 2. Section 3 formulates the problem of ISIs as a semiparametric estimation and Section 4 briefly introduces the information geometry for semiparametric models. The solution of the problem is given in Section 5 and Section 6 confirms the results by computer simulations. Section 7 concludes with some discussion.

4.2 Statistical Measures for ISIs

When a spike sequence is given and its N ISIs are written as $T_1, T_2, \ldots, T_N$, the C_V and S_K measures are defined as the standard deviation of the ISIs divided by the mean of the ISIs and the skewness of the ISI distribution, respectively. That is,

$$C_V = \frac{1}{\bar{T}}\sqrt{\frac{1}{N-1}\sum_{n=1}^{N}(T_n - \bar{T})^2}, \tag{65}$$

$$S_K = \frac{\frac{1}{N-1}\sum_{n=1}^{N}(T_n - \bar{T})^3}{\left(\frac{1}{N-1}\sum_{n=1}^{N}(T_n - \bar{T})^2\right)^{3/2}}, \tag{66}$$

where

$$\bar{T} = \frac{1}{N}\sum_{n=1}^{N} T_n. \tag{67}$$

The C_V measure expresses the regularity: It takes a low value for a regular spike sequence, one for a sequence of infinite length generated by a fixed Poisson process, and a large value when the process is time-dependent. The S_K measure shows the asymmetry of a sequence: It can be either positive or negative, but it takes two for a sequence of infinite length generated by a stationary Poisson process. However, since they are based on the mean $\bar{t}$ of the ISIs, C_V or S_K will become large when the spike rate is globally modulated, even though the spike sequence is locally quasi-regular [24, 36]. As a result, they are not suitable for classifying particular neurons such as those found in cortical areas that change their spike rate drastically in a waiting-period task, for example.

To overcome this problem, Shinomoto et al. [28] proposed the L_V measure, defined as

$$L_V = \frac{1}{N-1}\sum_{n=1}^{N-1}\frac{3(T_n - T_{n+1})^2}{(T_n + T_{n+1})^2}, \tag{68}$$

where the factor 3 is taken so that the expectation of L_V becomes one when the sequence obeys a stationary Poisson process. Since the L_V measure reflects the stepwise variability of ISIs and does not compare ISIs with different spike rates, L_V can assume a small value, even for a sequence with a time-variant spike rate. They confirmed that C_V undergoes a large change but L_V does not for a sequence generated by a time-dependent Poisson process [36].

Recently, Shinomoto et al. [29] proposed a variant of L_V called L_{VR}, that explicitly includes an absolute refractory period R for each neuron, where the value is common to all neurons. That is,

$$L_{VR} = \frac{1}{N-1} \sum_{n=1}^{N-1} \frac{3(T_n - T_{n+1})^2}{(T_n + T_{n+1} - 2R)^2}, \tag{69}$$

where the refractory period R is determined heuristically from the given data. L_{VR} is shown to classify the functions of neurons in other words, the value of L_{VR}.

4.3 Statistical Model of Interspike Intervals

In the literature, ISIs are modeled as a gamma process with a variable rate and a fixed shape parameter. Although the rate can fluctuate continuously, it can be simplified by assuming that the rate parameter is fixed between two spikes. Then, an interspike interval T independently obeys a gamma distribution with a rate $\xi^{(l)}$ and shape parameter κ,

$$q(T; \xi^{(l)}, \kappa) = \frac{(\xi^{(l)}\kappa)^\kappa}{\Gamma(\kappa)} T^{\kappa-1} \exp\left[-\xi^{(l)}\kappa T\right], \tag{70}$$

where l runs from 1 to N.

As for L_{VR}, the absolute refractory period R slightly modifies (70) to

$$\begin{aligned} &q(T; \xi^{(l)}, \kappa, R) \\ &= \frac{(\xi^{(l)}\kappa)^\kappa}{\Gamma(\kappa)} (T-R)^{\kappa-1} \exp\left[-\xi^{(l)}\kappa(T-R)\right]. \end{aligned} \tag{71}$$

This is assumed in the existing models that R has a constant value of zero. So, we consider (71) in the following.

From (71), the probability distribution of a spike sequence is written as

$$p(\{T\}; \kappa, R, k(\xi)) = \int \prod_{l=1}^{N} q(T_l, \xi^{(l)}, \kappa, R) k(\xi^{(l)}) \mathrm{d}\xi^{(l)}. \tag{72}$$

This is called a semiparametric model, where two kinds of parameters appear: One is a finite number of parameters of interest, κ and R, and the other is a nuisance parameter, k, which has an infinite degrees of freedom. In other words, estimating the shape parameter κ and the refractory period R is formulated as a semiparametric estimation [34].

4.4 Information Geometry for Semiparametric Models

We generalize (72) by assuming that m observations, $\{T^{(l)}\} \equiv \{T_1^{(l)}, \ldots, T_m^{(l)}\}$, are given for each $\xi^{(l)}$, where $\xi^{(l)}$ is generated from an unknown probability density $k(\xi)$. That is, the distribution of the ISIs in the lth set is described as

$$p(\{T^{(l)}\}; \xi^{(l)}, \kappa, R) = \prod_{i=1}^{m} q(T_i^{(l)}; \xi^{(l)}, \kappa, R) \tag{73}$$

and that of the sequence is

$$p(\{T\}; \kappa, R, k(\xi)) = \int \prod_{l=1}^{N/m} q(\{T^{(l)}\}, \xi^{(l)}, \kappa, R) k(\xi^{(l)}) \mathrm{d}\xi^{(l)}. \tag{74}$$

A semiparametric estimation is known to be solvable using the estimating function method [37, 38], where the estimator is the solution of the estimating equation,

$$\sum_{l=1}^{N/m} \sum_{i=1}^{m} f(T_i^{(l)}, \kappa, R) = 0, \tag{75}$$

where $f(T; \kappa, R)$ is an estimating function that satisfies

$$\mathrm{E}_{\kappa,R,k}[f(T; \kappa, R)] = 0 \tag{76}$$

for any κ, R and k. Here $\mathrm{E}_{\kappa,R,k}$ denotes the expectation with respect to the distribution

$$p(T; \kappa, R, k) = \int p(T; \xi, \kappa, R) k(\xi) \mathrm{d}\xi. \tag{77}$$

How can we find good estimating functions? The information geometry [31, 32], which sheds light on a various fields of information science, gives a basic theory for estimating functions. Amari and Kawanabe [39] show the conditions under which estimating functions exist, how large the set of estimating functions is, and what estimating function is optimal.

We give an intuitive explanation for the theory of estimating functions (see Section 3 of Miura et al. [35] and its references for details). Note that in the present paper, the same notation is employed and the superscripts (l) have been omitted for $\xi^{(l)}$ and $T_i^{(l)}$ in the following analysis.

In short, an estimating function extracts the component orthogonal to the nuisance parameters, that includes the elements of the parameters of interest and has no other elements. See Fig. 1, for example, where F_i shows the space in which the information on the parameters of interest is included, F_n shows the space of the nuisance parameters, and F_a shows the orthogonal complement to $F_i \oplus F_n$. Since no assumptions are given for the nuisance parameters, the elements of the estimate in the direction of F_n are of no use. In other words, consideration needs to be made of the space orthogonal to F_n. Hence, we denote by F_e the space where the elements of F_n are removed from F_i. The estimating functions are included in $F_e \oplus F_a$ so that they are not affected by nuisance parameters. Obviously, F_a has no information on the parameters of interest. This means that F_e itself is the optimal space, where the optimal estimating function is included.

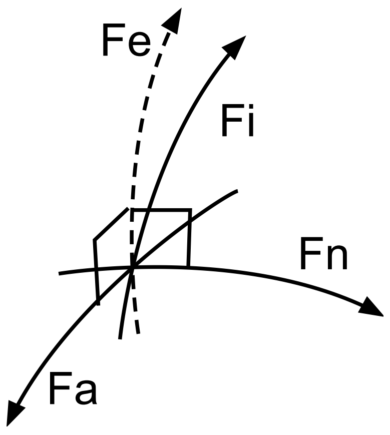


Figure 21. Illustration of estimating functions.

Using the Amari and Kawanabe method, Miura et al. [35] proved that there does not exist any estimating function when $m = 1$ and derived the optimal estimating function when $m > 1$, which leads to the optimal estimator of κ, irrespective of $\xi^{(l)}$. Their estimator has the minimum variance for estimating κ as long as the ISIs obey gamma distributions.

4.5 Estimating Functions for Interspike Intervals

Using the theory in the previous section, we derive the optimal estimating function for κ and R in a purely mathematical manner, where the refractory period R is common and constant but unknown, following the analysis in Miura et al. [35].

Substituting (71), (73) is written as

$$\prod_{i=1}^{m} q(T_i;\xi,\kappa,R) = \exp\left[\xi \cdot s(\{T\},\kappa) \right.$$
$$\left. + r(\{T\},\kappa,R) - \psi(\kappa,R,\xi)\right], \qquad (78)$$

where

$$s(\{T\},\kappa) = -\kappa \sum_{i=1}^{m} T_i, \qquad (79)$$

$$r(\{T\},\kappa,R) = (\kappa-1)\sum_{i=1}^{m} \log(T_i - R), \qquad (80)$$

$$\psi(\kappa,R,\xi) = -m\kappa\log(\xi\kappa) + m\log\Gamma(\kappa) - m\xi\kappa R. \qquad (81)$$

Amari and Kawanabe show that the optimal estimating functions for κ and R are given by

$$u_\kappa^I(\{T\},\kappa,R) = u_\kappa - \mathrm{E}[u_\kappa|s] \qquad (82)$$

$$= \sum_{i=1}^{m} \log(T_i - R) - m\mathrm{E}[\log(T_1 - R)|s], \qquad (83)$$

$$u_R^I(\{T\},\kappa,R) = u_R - \mathrm{E}[u_R|s] \qquad (84)$$

$$= (1-\kappa)\sum_{i=1}^{m} \frac{1}{T_i - R}$$
$$- m(1-\kappa)\mathrm{E}\left[\frac{1}{T_1 - R}|s\right], \qquad (85)$$

where u_κ^I and u_R^I are defined so that they are orthogonal to any function of s, as seen in (82) and (84). The marginal distribution of s and the conditional expectation in the last term of (83) are given in Appendix of Miura et al. [35] as being

$$
\begin{aligned}
& p(s) = \\
& \int \delta \left[s + \kappa \sum_{i=1}^{m} T_i \right] \prod_{i=1}^{m} q(T_i; \xi, \kappa, R) \mathrm{d}T_i k(\xi) \mathrm{d}\xi && (86) \\
& = \int \prod_{i=1}^{m-1} B(i\kappa, \kappa) \left(-\frac{s}{\kappa} - mR \right)^{m\kappa - 1} \frac{(\xi\kappa)^{m\kappa}}{\Gamma(\kappa)^m} \\
& \cdot \exp[\xi s + \xi\kappa m R] \frac{k(\xi)}{\kappa} \mathrm{d}\xi, && (87)
\end{aligned}
$$

$$
\begin{aligned}
\mathrm{E}[\log(T_1 - R)|s] = & \int \log(T_1 - R) \delta \left[s + \kappa \sum_{i=1}^{m} T_i \right] \\
& \prod_{i=1}^{m} q(T_i; \xi, \kappa, R) \mathrm{d}T_i k(\xi) \mathrm{d}\xi \frac{1}{p(s)} && (88) \\
& = \log \left[-\frac{s}{\kappa} - mR \right] - \phi(m\kappa) + \phi(\kappa), && (89)
\end{aligned}
$$

where δ is the Dirac delta function and $\phi(\kappa)$ is the digamma function defined as

$$
\phi(\kappa) = \frac{\Gamma'(\kappa)}{\Gamma(\kappa)}. \tag{90}
$$

Similarly, we can derive the conditional expectation in the last term of (85) as

$$
\begin{aligned}
\mathrm{E}\left[\frac{1}{T_1 - R} \Big| s \right] = & \int \frac{1}{T_1 - R} \delta \left[s + \kappa \sum_{i=1}^{m} T_i \right] \\
& \prod_{i=1}^{m} q(T_i; \xi, \kappa, R) \mathrm{d}T_i k(\xi) \mathrm{d}\xi \frac{1}{p(s)} && (91) \\
& = \frac{1}{-\frac{s}{\kappa} - mR} \left(1 - \frac{m\kappa - \kappa}{\kappa - 1} \right). && (92)
\end{aligned}
$$

In total, the optimal estimating functions are written as

$$u^I_\kappa(\{T\},\kappa,R) = \sum_{i=1}^{m}\log(T_i - R)$$

$$- m\log\sum_{i=1}^{m}(T_i - R) + m\phi(m\kappa) - m\phi(\kappa), \tag{93}$$

$$u^I_R(\{T\},\kappa,R) = (1-\kappa)\sum_{i=1}^{m}\frac{1}{T_i - R}$$

$$- \frac{m(1-\kappa)}{\sum_{i=1}^{m}(T_i - R)}\left(1 - \frac{m\kappa - \kappa}{\kappa - 1}\right). \tag{94}$$

Using the estimating functions above, the estimators of κ and R are the solution of the estimating equations,

$$\sum_{l=1}^{N} u^I_\kappa(\{T^{(l)}\},\kappa,R) = 0, \tag{95}$$

$$\sum_{l=1}^{N} u^I_R(\{T^{(l)}\},\kappa,R) = 0. \tag{96}$$

4.6 Computer Simulations

To confirm the validity of the theoretical analysis given above, computer simulations were carried out. In each experiment, κ, R, m and ξ are fixed at 0.5, 0.2, 3 and 1[1], respectively.

Fig. 2 shows the estimated κ and R determined by the optimal estimating functions derived above as a function of the number N of observations, where the solid and dashed lines denote the averages of estimated κ and R over 1000 trials and each error bar shows the standard deviation. We can see that the values of κ and R converge to the true values, 0.5 and 0.2, with decreasing deviations, as N increases.

Next, we compared the estimate error of κ of our method to those of the conventional method by Miura et al. with $R = 0.2$ (the true value) and with $R = 0$ (the conventional model). Obviously, the former gives the lower bound of the estimate error in this framework. In Fig. 3 the solid, dashed and dotted

[1]The estimator does not depend on ξ, and we can assume a constant value.

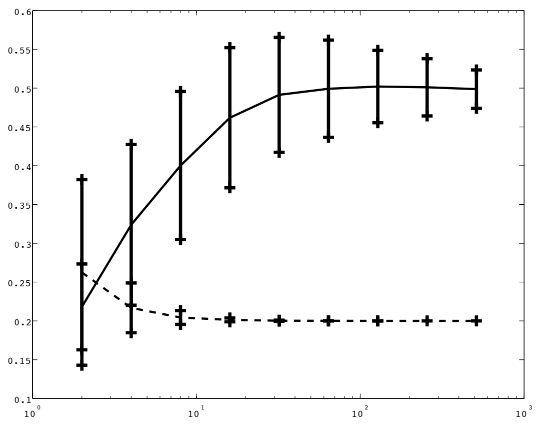


Figure 22. κ and R of the proposed method versus N.

lines describe the root-mean-square errors of our method, the lower bound and the conventional model, respectively. We can see that the proposed method has a comparable error to the lower bound and smaller than the conventional model even when examples are few.

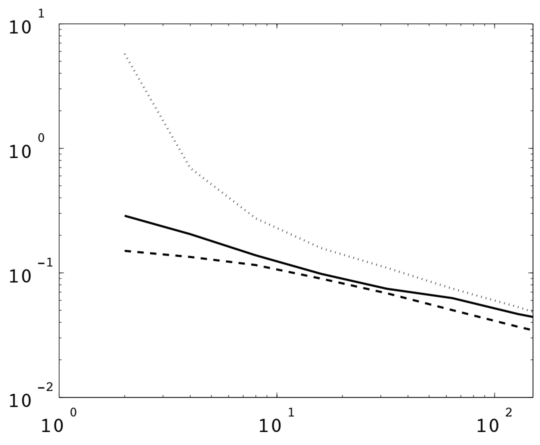


Figure 23. Comparison of the estimated errors of κ.

4.7 Conclusions

In this section, we discuss the statistical models of ISIs. When a spike sequence of a neuron obeys a gamma process with a time-variant spike rate, the information geometry gives the optimal estimating function from the statistical viewpoint. The method is also applicable to the modified model which has an absolute refractory period, and successfully estimate not only the shape parameter but also the refractory period.

As shown in [34], L_V is an approximated estimator of the regularity, and so must be L_{VR}. Since L_{VR} is a strong tool for neuron classification [29], our method will replace L_{VR} in the field in the near future.

5. Conclusion

We studied three subjects: genetic algorithms, SVMs with forgetting factors, and neural spike sequences.

In Sect.2, GAs are analyzed from a network point of view based on the assumption that the shorter the CPL is, the more effectively a GA can find desirable solutions. The main contribution of this study is the derivation of the CPLs of a genetic network of $L = 2$ under both conditions of equal weights and different weights, where weights represent transition probabilities. When all the weights of edges are set to 1, the crossover operation surely shortens the CPL linearly with the length of the individuals. In the case in which each edge might have a different weight, the crossover operation does not necessarily shorten the CPL. Instead, the CPL is monotonically decreasing where $\mu \leq 27/28$. It is obvious that a GA does not necessarily work best at $\mu = 27/28$. This means that the effectiveness of the process of searching does not seem to directly correspond to its CPL.

In Sect.3, SVMs with forgetting factors have been proposed and analyzed in an asymptotic manner. In the case of an exponential forgetting factor, the proposed asymptotic theory on its generalization error in the simple noiseless case shows that the proposed method has a non-zero lower bound in the average generalization error, differently from the conventional SVMs or the RLS algorithm for adaptive filters, where errors decrease in the order of $O(1/N)$. The main problem of SVMs with exponential forgetting factors is that this approach never allows generalization errors to converge to zero even for the proposed simple problem. This difficulty is overcome by introducing a factorial forgetting factor.

In the case of the factorial forgetting factor, the generalization errors converges to zero at the cost of slower convergence rate. That is because the sum of the forgetting factors diverges to infinity when the number of examples goes to infinity. The convergence rate is calculated mathematically and it is $O(1/(\log N)^{(3/2)})$, which is slower than the rate in the of an exponential forgetting factor, $O(1/N)$, but it is a fair cost to pay for the quality of convergence.

In Sect.4, the statistical models of interspike intervals are developed. When a spike sequence of a neuron obeys a gamma process with a time-variant spike rate, the information geometry gives the optimal estimating function from the statistical viewpoint. The method is also applicable to the modified model which has an absolute refractory period, and successfully estimates not only the shape parameter but also the refractory period.

Acknowledgements

I am heartily thankful to my supervisor, Dr. Kazushi Ikeda, whose delightfull encouragement, advice and support from six years ago have been saving me uncountable times. I also would like to make a special reference to Dr. Yuji Matsumoto who understood my ideas and helped me finish writing this dissertation. Dr. Takashi Takenouchi's precious ideas and advice helped me find solutions to many obstacles. Finally, I would like to say thank you to my father and mother.

References

[1] J. Holland, "*Adaptation in Natural and Artificial Systems*," MIT Press Cambridge, MA, USA, 1992.

[2] D. Goldberg *et al.*, "*Genetic Algorithm in Search, Optimization and Machine Learning*," Addison-Westey, Readomg, Mass, 1989.

[3] J. Suzuki, "A Markov Chain Analysis on Simple Genetic Algorithms," *IEEE Transactions on Systems, Man and Cybernetics*, vol.25, no.4, pp.655–659, 1995.

[4] D. Goldberg *et al.*, "Finite Markov Chain Analysis of Genetic Algorithms," *Proceedings of the Second International Conference on Genetic Algorithms*, pp. 1–8 1987.

[5] D. Watts and S. Strogatz, "Collective Dynamics of Small-world Networks," *Nature*, vol.393, no.6684, pp.409–10, 1998.

[6] P. Stadler and C. Stephens, "Landscapes and Effective Fitness," *Theoretical Biology*, vol.8, no.4, pp.389–431, 2003.

[7] A.L. Barabasi, and RE. Crandall, "*Linked: The New Science of Networks*," American Journal of Physicsp. 2003.

[8] Vapnik, V.N.: The Nature of Statistical Learning Theory. Springer-Verlag, New York, NY (1995)

[9] Schölkopf, B., Burges, C., Smola, A.J.: Advances in Kernel Methods: Support Vector Learning. Cambridge Univ. Press, Cambridge, UK (1998)

[10] Cristianini, N., Shawe-Taylor, J.: An Introduction to Support Vector Machines. Cambridge Univ. Press, Cambridge, UK (2000)

[11] Smola, A.J., Bartlett, P.L., Schölkopf, B., Schuurmans, D., eds.: Advances in Large Margin Classifiers. MIT Press, Cambridge, MA (2000)

[12] Klinkenberg, R., Joachims, T.: Detecting concept drift with support vector machines. Proc. ICML, (2000) 487–494

[13] Cauwenberghs, G. and Poggio, T.: Incremental and Decremental Support Vector Machine Learning ADVANCES IN NEURAL INFORMATION PROCESSING SYSTEMS (2001) 409–415

[14] Liu, F., Zhang, T., Zhang, R.: Modified kernel RLS-SVM based multiuser detection over multipath channels. IEICE Trans. Fundamentals **E86-A**(8) (2003) 1979–1984

[15] Haykin, S.: Adaptive Filter Theory. 3rd edn. Prentice-Hall (1996)

[16] Montazeri, M. and Duhamel, P.: A set of algorithms linking NLMS and block RLS algorithms Signal Processing, IEEE Transactions on **43**(2) (1995) 444–453

[17] Schölkopf, B., Smola, A.J., Williamson, R.C., Bartlett, P.L.: New support vector algorithms. Neural Computation **12**(5) (2000) 1207–1245

[18] Cortes, C., Vapnik, V.: Support vector networks. Machine Learning **20** (1995) 273–297

[19] Ikeda, K., Aoishi, T.: An asymptotic statistical analysis of support vector machines with soft margins. Neural Networks **18**(3) (2005) 251–259

[20] Bennett, K.P., Bredensteiner, E.J.: Duality and geometry in SVM classifiers. Proc. ICML (2000) 57–64

[21] Hunaya, H., Nomura, Y., Ikeda, K.: A support vector machine with forgetting factor and its statistical properties. Advances in Neuro-Information Processing (2009) 929–936

[22] Ikeda, K., Yamasaki, T.: Incremental Support Vector Machines and Their Geometrical Analyses. Neurocomputing, 70/13-15 (2007), 2528-2533.

[23] Takemasa, Y., Ikeda, K.: Statistical Properties of Incremental SVMs Storing Support Vectors. SICE Journal of Control, Measurement, and System Integration, 3, in press.

[24] Holt G R, Softky W R, Koch C, Douglas R J. Comparison of discharge variability in vitro and in vivo in cat visual cortex neurons. Journal of Neurophysiology, 1996, 75: 1806–1814

[25] Shinomoto S, Sakai Y, Funahashi S. The Ornstein-Uhlenbeck process does not reproduce spiking statistics of neurons in prefrontal cortex. Neural Computation, 1999, 11: 935–951

[26] Sakai Y, Funahashi S, Shinomoto S. Temporally correlated inputs to leaky integrate-and-fire models can reproduce spiking statistics of cortical neurons. Neural Networks, 1999, 12: 1181–1190

[27] Shinomoto S, Shima K, Tanji J. New classification scheme of cortical sites with the neuronal spiking characteristics. Neural Networks, 2002, 15(10): 1165–1169

[28] Shinomoto S, Shima K, Tanji J. Differences in spiking patterns among cortical neurons. Neural Computation, 2003, 15(12): 2823–2842

[29] Shinomoto S, Kim H, Shimokawa T, Matsuno N, Toyama K. Relating neuronal firing patterns to functional differentiation of cerebral cortex. PLoS Computational Biology, 2009, 5:e1000433

[30] Tiesinga P H E, Fellous J M, Sejnowski T J. Attractor reliability reveals deterministic structure in neuronal spike trains. Neural Computation, 2002, 14:1629–1650

[31] Amari S. Differential-Geometrical Methods in Statistics. Lecture Notes in Statistics. Berlin: Springer-Verlag, 1985

[32] Amari S, Nagaoka H. Methods of Information Geometry. New York: Oxford University Press, 2000

[33] Miura K, Shinomoto S, Okada M. Search for optimal measure to discriminate random and regular spike trains. IEICE Technical Report, 2004, NC2004-52

[34] Ikeda K. Information geometry of interspike intervals in spiking neurons. Neural Computation, 2005, 17(12): 2719–2735

[35] Miura K, Okada M, Amari S. Estimating spiking irregularities under changing environments. Neural Computation, 2006, 18(10): 2359–2386

[36] Shinomoto S, Tsubo Y. Modeling spiking behavior of neurons with time-dependent Poisson processes. Physical Review E, 2001, 64:041910

[37] Godambe V P. Conditional likelihood and unconditional optimum estimating equations. Biometrika, 1976, 63: 277–284

[38] Godambe V P, Ed. Estimating Functions, Oxford, UK: Oxford Univ. Press, 1991

[39] Amari S, Kawanabe M. Information geometry of estimating functions in semiparametric statistical models. Bernoulli, 1996, 2(3)