THE ADABOOST FLOW

A. LYKOV, S.MUZYCHKA AND K. VANINSKY

Abstract. We introduce a dynamical system which we call the AdaBoost flow. The flow is defined by a system of ODEs with control. We show that three algorithms of the AdaBoost family (i) the AdaBoost algorithm of Schapire and Freund (ii) the arc-gv algorithm of Breiman (iii) the confidence rated prediction of Schapire and Singer can be can be embedded in the AdaBoost flow.

The nontrivial part of the AdaBoost flow equations coincides with the equations of dynamics of nonperiodic Toda system written in terms of spectral variables. We provide a novel invariant geometrical description of the AdaBoost algorithm as a gradient flow on a foliation defined by level sets of the potential function.

We propose a new approach for constructing boosting algorithms as a continuous time gradient flow on measures defined by various metrics and potential functions. Finally we explain similarity of the AdaBoost algorithm with the Perelman's construction for the Ricci flow.

Contents

1. Introduction.

The AdaBoost algorithm does not need an advertisement in data mining community. It was discovered by Robert Schapire and Yoav Freund in their seminal paper in 1997, [6]. Nowadays, together with the PageRank algorithm, the AdaBoost is considered among the top ten algorithm in data mining, [16]. It is worth mentioning that for their AdaBoost paper, Schapire and Freund won the Gödel Prize, which is one of the most prestigious awards in theoretical computer science, in the year 2003.

The AdaBoost algorithm appeared from an abstract problem. In 1988, Kearns and Valiant asked a question whether a learning algorithm that performs just slightly better than random guess could be boosted into an arbitrarily accurate learning algorithm. Schapire in 1990, [13], found that the answer is yes, and the proof he gave is a construction of the first boosting algorithm. The AdaBoost proposed by Freund and Schapire in 1997, [6], initiated an extensive research on theoretical aspects of ensemble methods, which can be easily found in machine learning and statistical literature. From a practical viewpoint the AdaBoost is used to construct spam filtering systems, search engines, face recognition and recommender systems to name a few possibilities. A Mathematician can forget about all that and treat the AdaBoost as an algorithm which solves some special optimization problem.

In the present paper we introduce a dynamical system which we call the AdaBoost flow. The flow is defined by a system of ODEs with control. We show that, by a suitable choice of control, three algorithms of the AdaBoost family (i) the AdaBoost algorithm of Schapire and Freund, [6], (ii) the arc-gv algorithm of Breiman , [4], (iii) the confidence rated prediction of Schapire and Singer, [14], can be can be embedded in the AdaBoost flow.

The AdaBoost flow is nothing but the well known dynamics of the classical non-periodic Toda system of particles on isospectral manifold. Introduced in 1967, see [15], the Toda lattice is a basic example of a system of classical mechanics integrable in the Liouville sense. Its complete integrability was proved by Moser in 1974, [10]. As an outcome of this connection we provide a novel geometrical description of the AdaBoost as a gradient flow on a foliation defined by level sets of a potential function.

We propose a general approach for constructing AdaBoost like algorithms using gradient flows defined by various metrics and potential functions. We also introduce a new continuous time algorithm which we call SuperBoost. Finally we discuss similarity of the AdaBoost algorithm with the Perelman's approach to control the Ricci flow. We show that all parts of the Schapire and Freund construction have their counterparts in the Perelman's construction. We present a dictionary between two problems. It turns out, the AdaBoost and the Perelman's construction are different realization of the same idea.

The idea can be described vaguely as follows. Suppose we have a space X which is mapped into a space Y by a family of maps f_t, where $t \in [0, +\infty)$. The space of all such maps f_t is denoted by $\mathcal{F}$. We are interested in the behavior of maps when t becomes large. For example, X can be a manifold and Y can be a space of symmetric nonnegative matrices. In this case f_t define a metric on X for each t. The family f_t can be produced by the Ricci flow.

Suppose that there are points $x_1, ..., x_N$; where the maps f_t behave badly when t becomes large, but we still want to control the maps around these points. In order to do this an extension of the original dynamics on the space $\mathcal{F}$ is introduced. Let $\mathcal{W}$ be a space of all probability measures on X. At each moment t the probability measure w_t is chosen such that some (integral) functional I of two variables increases with time

$$\frac{d}{dt} I(f_t, w_t) \geq 0.$$

In other words the functional I plays the role of a Lyapunov function for the dynamics on the extended phase space $\mathcal{F} \times \mathcal{W}$. Of course, there are different variations of the described construction. For example w_t can be replaced by w_0 as in the case of the AdaBoost. The point of this construction is that monotonicity of the functional I allows to control the maps f_t around "bad" points.

The paper consists of two sections. In the first section we present all necessary facts about classical discrete AdaBoost algorithm. For many of these facts we establish continuous time counterparts. In the second section we introduce continuous time AdaBoost flow and consider embeddings of three algorithms of the AdaBoost family into continuous time flow. We demonstrate gradient character of the AdaBoost flow and introduce a general approach to boosting based on continuous time gradient flows. Finally we show that the AdaBoost and the Perelman's construction for the Ricci flow are closely related.

We would like to thank Igor Krichever, Sasha Veselov and Vadim Malyshev for stimulating discussions. Our special thanks to Henry McKean for numerous remarks.

2. Discrete time AdaBoost algorithm.

2.1. **Basic AdaBoost algorithm.** We start with simple description of the AdaBoost algorithm. The reader can consult the book [7] for an additional information.

Given a set of points

$$TS = \{(x_1, y_1), .., (x_m, y_m)\},$$

where $x_i \in X$ and $y_i \in \{-1, +1\}$. Usually $X = R^d$ and y_i are called labels. Such set TS is called a training set and it will be used to construct a decision rule. Also given a finite set of functions $\mathcal{H}_0 = \{h_\gamma,\ \gamma \in \Gamma\}$; where each $h_\gamma \in \mathcal{H}_0$ is such that $h_\gamma : X \to \{-1, +1\}$. These functions are called weak classifiers. Pick some probability distribution $w(i),\ i = 1, ..., m$; on the training set. The classification error of any weak classifier h_γ with respect to the measure w is defined as

$$W^-(h_\gamma, w) = w\{i : h_\gamma(x_i)y_i = -1\}.$$

In a practical situation this error can be quite big, *i.e.* $W^-(h_\gamma, w) >> 0$. That is the reason to call h_γ weak classifier. To remedy large classification error one can try to combine weak classifiers.

Let $\mathcal{H}$ be a positive cone over a set of basic classifiers

$$\mathcal{H} = \{H : H = \sum_{\gamma\in\Gamma} \alpha_\gamma h_\gamma;\ h_\gamma \in \mathcal{H}_0,\ \alpha_\gamma \geq 0\}.$$

From any H, the combined classifier $\mathbf{H} : X \to \{-1, 0, +1\}$ can be constructed. Namely, if $H(x) \neq 0$, then $\mathbf{H}(x) = \operatorname{sign} H(x)$; if $H(x) = 0$, then no decision can be made and $\mathbf{H}(x) = 0$. Let us define

$$W^-(H, w) = w\{i : \mathbf{H}(x_i)y_i = -1\} \qquad W^0(H, w) = w\{i : \mathbf{H}(x_i)y_i = 0\}.$$

The problem is to minimize the error $W^- + W^0$ of the combined classifier $\mathbf{H}$ by choosing appropriate values of α_γ. The difficulty of this constrained minimization problem is that the error is almost everywhere constant on $\mathcal{H}$ and gradient methods can not be applied directly.

The AdaBoost algorithm offers a candidate for the solution of this problem in a series of $N + 1$ rounds, where N is some integer number. For any $n = 0, 1, ..., N$; the combined classifier $\mathbf{H}_n(x) = \mathbf{H}_n : X \to \{-1, 0, +1\}$ is defined as

$$H_n = t_0 h_{\gamma_0} + ... + t_n h_{\gamma_n};$$

where the sequence of positive numbers $t_0, t_1, ..., t_n$, is constructed simultaneously with h's. The final classifier $\mathbf{H} = \mathbf{H}_N$ is a candidate for the solution of the minimization problem. In practical applications one simply chooses N large enough. Theoretical bound for the misclassification error will be given below.

Specifically, the AdaBoost recursively constructs a family of classifiers by means of probability measures $w_0, w_1, ..., w_N$. It starts with the fixed distribution w:

$$w_0 : \quad w_0(i) = w(i), \qquad i = 1, ..., m.$$

Given a distribution $w_n,\ n = 0, ..., N$; the AdaBoost algorithm picks arbitrary h_{γ_n} from $\mathcal{H}_0$ such that

$$W_n^- = W^-(h_{\gamma_n}, w_n) < 1/2. \tag{2.1}$$

If at some step it is not possible to do this, *i.e.* if

$$\min_{h_\gamma \in \mathcal{H}_0} W^-(h_\gamma, w_n) \geq 1/2;$$

then the procedure stops unfinished. The reason for this will be explained later. Note that on each step the algorithm does not have to go through the whole set $\mathcal{H}_0$, one has just to find one h_γ that satisfies 2.1. If this is the case and $W_n^- < 1/2$ for all $n = 0, 1, ..., N$; then the measure is constructed recurrently

$$w_{n+1}(i) = \frac{e^{-t_n y_i h_{\gamma_n}(x_i)} w_n(i)}{Z_n},$$

where t_n is some positive number and

$$Z_n = \sum_{i=1}^{m} e^{-t_n y_i h_{\gamma_n}(x_i)} w_n(i).$$

The whole procedure can be represented by a diagram

$$\begin{array}{ccccccc} H_0 & & H_1 & & ... & & H_N \\ \uparrow & \searrow & \uparrow & \searrow & ... & \searrow & \uparrow \\ w_0 & & w_1 & & ... & & w_N \end{array}$$

The function $H(x) = H_N(x)$ takes values in the segment $[-T, +T]$, where $T = \sum_{n=0}^{N} t_n$.

At each step of the AdaBoost procedure the set of training points TS falls into two categories G_n (good) and B_n (bad). Points of G_n are classified correctly by h_{γ_n}

$$G_n = \{(x_i, y_i) : h_{\gamma_n}(x_i) y_i = +1\}.$$

The measure $W_n^+ = w_n\{G_n\}$ of these points decreases upon the next step

$$w_n(i) \to w_{n+1}(i) = \frac{e^{-t_n}}{Z_n} w_n(i), \qquad (x_i, y_i) \in G_n.$$

The points of B_n are misclassified by h_{γ_n}:

$$B_n = \{(x_i, y_i) : h_{\gamma_n}(x_i) y_i = -1\}.$$

The measure $W_n^- = w_n\{B_n\}$ of these points increases upon the next step:

$$w_n(i) \to w_{n+1}(i) = \frac{e^{t_n}}{Z_n} w_n(i), \qquad (x_i, y_i) \in B_n.$$

Apparently, $W_n^+ + W_n^- = 1$ and $W_n^- < \frac{1}{2},\ n = 0, 1, ..., N$. The values of t_n at each step are chosen to minimize probability of error of the final combined classifier. Remark 2.1 shows that with an optimal choice of t_n

$$w_{n+1}\{G_n\} = w_{n+1}\{B_n\} = \frac{1}{2}.$$

2.2. **AdaBoost map on the extended phase space.** Boosting can be viewed as a discrete time dynamical system on the extended phase space $\mathcal{H} \times \mathcal{W}$ which is the direct product of the positive cone $\mathcal{H}$ and the simplex of probability measures $\mathcal{W} = \{w : \sum_{i=1}^m w(i) = 1,\ w(i) \geq 0\}$. The vector field $v(H, w)$ on $\mathcal{H} \times \mathcal{W}$ is a constant $h = h(w)$ on the "fibers" $\mathcal{H}_w = \{(H, w') :\ H \in \mathcal{H},\ w' = w\}$, *i.e.*

$$v(H, w) = h, \qquad \text{for any } H \in \mathcal{H}_w.$$

The AdaBoost dynamics maps (H_n, w_n) into (H_{n+1}, w_{n+1}) by the rule

$$w_{n+1}(i) = \frac{e^{-t_n y_i v(H_n, w_n)(x_i)} w_n(i)}{Z_n}, \qquad n = 0, 1, ...;\ i = 1, 2, ..., m; \tag{2.2}$$

$$H_{n+1} = H_n + t_{n+1} v(H_n, w_{n+1}), \qquad n = -1, 0, 1, ... \tag{2.3}$$

Along the trajectory $\{(H_n, w_n),\ n = 0, 1, ..., N\}$ the Adaboost dynamics drives the error

$$W^-(H_n, w_0) + W^0(H_n, w_0) = \sum_{i=1}^m w_0(i) \chi_{[y_i H_n(x_i) \leq 0]}(x_i)$$

to zero. To prove this one overestimates the function $W^-(\cdot, w_0) + W^0(\cdot, w_0)$ which has no gradient by some smooth function $\mathcal{E}(\cdot, w_0)$ defined as

$$\mathcal{E}(H, w) = \sum_{i=1}^m w(i) e^{-y_i H(x_i)}.$$

Clearly,

$$W^-(H, w_0) + W^0(H, w_0) = \sum_{i=1}^m w(i) \chi_{[y_i H(x_i) \leq 0]}(x_i) \leq \mathcal{E}(H, w) = \sum_{i=1}^m w(i) e^{-y_i H(x_i)}.$$

The function $\mathcal{E}(\cdot, w_0)$ plays the role of Lyapunov function for the AdaBoost dynamics. It is strictly convex in the first argument

$$\mathcal{E}(\lambda H' + \mu H'', w) < \lambda\, \mathcal{E}(H', w) + \mu\, \mathcal{E}(H'', w),$$

and linear in the second.

Equations of the Adaboost dynamics imply that the values of $\mathcal{E}(H_n, w_0)$ along the trajectory satisfy two equivalent identities. The first connects two consecutive values

$$\mathcal{E}(H_{n+1}, w_0) = Z_{n+1} \mathcal{E}(H_n, w_0). \tag{2.4}$$

The second identity reads

$$\mathcal{E}(H_n, w_0) = \prod_{p=0}^{n} Z_p. \tag{2.5}$$

The identities can be easily proved by means of the relation

$$\mathcal{E}(H_n, w_k) = Z_k \times \cdots \times Z_n \, \mathcal{E}(H_{k-1}, w_{n+1}), \qquad k < n; \tag{2.6}$$

and boundary condition $\mathcal{E}(H_{-1}, w_n) = 1$, due to $H_{-1} = 0$. Both identities have their counterparts in the continuous time case.

The constant $t_n > 0$ is chosen to minimize Z_n on each step. In detail,

$$Z_n(t) = e^{-t} W_n^+ + e^t W_n^-,$$

and from the condition of critical point $\frac{dZ_n}{dt} = 0$ we get an explicit formula

$$t_n = \frac{1}{2} \log \frac{W_n^+}{W_n^-}.$$

The constant t_n is positive if and only if $W_n^- < 1/2$. The formula for Z_n

$$Z_n = 2\sqrt{W_n^+ W_n^-};$$

with optimal t_n follows easily. If $W_n^- = \frac{1}{2} - \beta_n$, then

$$Z_n = \sqrt{1 - 4\beta_n^2} \le e^{-2\beta_n^2},$$

and

$$W^-(H_N, w_0) + W^0(H_N, w_0) \le \mathcal{E}(H_N, w_0) \le e^{-2\sum_{p=0}^{N} \beta_p^2}.$$

Therefore the training error decays exponentially with N, if β_n are uniformly bounded from zero. Moreover, if N is such that

$$\min_i w_0(i) > e^{-2\sum_{p=0}^{N} \beta_p^2},$$

then $W^-(H_N, w_0) + W^0(H_N, w_0) = 0$.

Remark 2.1. *Note that formulas for Z_n and t_n imply*

$$w_{n+1}\{B_n\} = \frac{e^{t_n}}{Z_n} W_n^- = \frac{e^{t_n} W_n^-}{2\sqrt{W_n^- W_n^+}} = \frac{1}{2}.$$

2.3. **AdaBoost as entropy projection.** The fact that updates of the measure w_n on each step are solutions of an extremal problem is due to Kivinen and Warmouth, [9]. It can be checked directly that w_{n+1} is a solution of the following minimization problem.

Given w_n find such w that it minimizes Kulback-Leibler divergence

$$KL(w||w_n) = \sum_{p=1}^{m} w(p) \log \frac{w(p)}{w_n(p)},$$

subject to constraints

$$\sum_i y_i h_n(x_i) w(i) = 0, \tag{2.7}$$

and

$$\sum_i w(i) = 1.$$

The solution of this problem w_{n+1} is given by a projection of w_n into a hyperplane defined by condition 2.7. Usually such a projection is given by a geodesic curve which is a solution of the second order equation.

3. Continuous time AdaBoost flow.

The first observation of our paper can be described as follows. We will write a gradient flow which is a system of the first order equations. The solution of these equations is a continuous curve which connects w_n and w_{n+1}. The time it takes for the trajectory of the gradient flow to travel from w_n to w_{n+1} is exactly the weight t_n. In order to write such a gradient flow we use the metrics which arises from the KL divergence and a potential function associated with h_n.

3.1. **Differential equations for the AdaBoost flow.** In this section we introduce a continuous time AdaBoost flow on the $\mathcal{H}\times W$. Namely we construct a family of combined classificators H_t and measures w_t, for all $t,\ 0 \le t \le T$. Naturally,

$$H(x_1), H(x_2), ..., H(x_m), \qquad w(1), w(2), ..., w(m);$$

are coordinates on $\mathcal{H} \times W$.

Let $\gamma_t : [0, \infty) \to \Gamma$ be a function with finite number of values on any finite interval and let it be continuous from the right with respect to the discrete topology on Γ. We choose a vector field constant on $\mathcal{H}_{w_t}$ as in discrete case, *i.e.*

$$v(H_t, w_t) = h_{\gamma_t}.$$

In this language the AdaBoost flow differential equations are the following

$$\frac{d}{dt} H(x_k) = \quad v(H_t, w_t)(x_k), \qquad k = 1, 2, ..., m; \tag{3.1}$$

$$\frac{d}{dt} w_t(k) = -y_k v(H_t, w_t)(x_k) w_t(k) + \sigma_t w_t(k), \qquad k = 1, 2, ..., m; \tag{3.2}$$

where $\sigma_t = \sigma_{w_t} = \sum_{p=1}^m y_p v(H_t, w_t)(x_p) w_t(p)$. It can be checked easily that the quantity

$$w(1) + w(2) + ... + w(m)$$

is an integral of motion. Therefore, the orbits of the AdaBoost flow remain on the simplex W for all times.

Differential equation allows us to define the AdaBoost flow when the weak classifiers take arbitrary real values, *i.e.* we assume that $h_\gamma : X \to R^1$; for any $h_\gamma \in \mathcal{H}_0$. The solution of the differential equations with a fixed γ_t is a straight line motion

$$H_t = H_0 + t \times v(H_0, w_t);$$

and the measure is

$$w_t(k) = \frac{w_0(k) e^{-t y_k v(H_0, w_0)(x_k)}}{\sum_{p=1}^m w_0(p) e^{-t y_p v(H_0, w_0)(x_p)}}, \qquad k = 1, 2, ..., m.$$

The equations for $w_t(k),\ k = 1, ..., m$; coincide with the equations for spectral weights in [10] and [19]. In the case of Toda lattice all the numbers $y_k v(H_t, w_t)(x_k)$ are distinct for different k. They are the simple spectrum of the Jacobi matrix. Here the situation is different. In the case of weak classifiers which take only two

values $+1$ and -1, the components $y_k v(H_t, w_t)(x_k)$ of the vector field also take only these two possible values.

Now we want to write the equations 3.2 in a different form. This new form will be used in section 3.7. The measure w can be defined in terms of the potential function f as $w_t(k) = e^{-f_t(k)}$, $k = 1, ..., m$. Then we can write 3.2 as

$$\frac{d}{dt} f_t(k) = y_k v(H_t, w_t)(x_k) - \sigma_t, \qquad k = 1, 2, ..., m; \tag{3.3}$$

What are the orbits of the AdaBoost flow on the simplex $\mathcal{W}$? Let weak classifiers take values $\{+1, -1\}$. Define

$$W^+ = w_0\{i : y_i h(x_i) = 1\},$$
$$W^- = w_0\{i : y_i h(x_i) = -1\} > 0;$$

and

$$U(t) = \frac{1}{W^+ + e^{2t} W^-}, \qquad t \geqslant 0.$$

Lemma 3.1. *Assume that the AdaBoost flow runs for all $t \geq 0$ with the fixed $v(H_t, w_t)$ i.e. it comes from a fixed single classifier h. Then for*

$$w_t = \mathcal{L}[w_0] + \mathcal{D}[w_0] U(t),$$

where the vectors $\mathcal{L}[w_0]$ and $\mathcal{D}[w_0]$ are defined by the following formulas:

$$\mathcal{L}[w_0](i) = \begin{cases} 0 & y_i h(x_i) = 1 \\ \frac{w_0(i)}{W^-} & y_i h(x_i) = -1 \end{cases} \qquad i = 1, \ldots, m;$$

$$\mathcal{D}[w_0](i) = \begin{cases} w_0(i) & y_i h(x_i) = 1 \\ -\frac{W^+}{W^-} w_0(i) & y_i h(x_i) = -1 \end{cases} \qquad i = 1, \ldots, m;$$

Proof. Substitute explicit expression for the flow into the formulas. □

Since for $t \geqslant 0$, $U(t) \in (0, 1]$, the orbit of the point w_0 under the AdaBoost flow is a semi-interval between the points w_0 and $\mathcal{L}(w_0)$. Moreover, as $t \to +\infty$, $w_t \to \mathcal{L}[w_0]$, *i.e.* the AdaBoost flow transports the measure towards the points where the classificator makes an error.

Now let weak classifiers take values in $\{-1, 0, +1\}$. Define:

$$W^0 = w_0\{i : h(x_i) = 0\}.$$

Let us assume $0 < W^0 < 1$. Define:

$$Z(t) = W^+ e^{-t} + W^- e^t + W^0$$
$$\alpha(t) = \frac{e^{-t}}{Z(t)},$$
$$\beta(t) = \frac{1}{Z(t)}.$$

Lemma 3.2. *For any $t \geqslant 0$,*

$$w_t = \mathcal{L}[w_0] + \mathcal{D}^+[w_0]\alpha(t) + \mathcal{D}^0[w_0]\beta(t),$$

where the vectors $\mathcal{D}^+[w_0]$ and $\mathcal{D}^0[w_0]$ are defined as:

$$\mathcal{L}[w_0](i) = \begin{cases} \frac{w_0(i)}{W^-} & y_i h(x_i) = -1 \\ 0 & \textit{otherwise} \end{cases} \qquad i = 1, \ldots, m;$$

$$\mathcal{D}^+[w_0](i) = \begin{cases} w_0(i) & y_i h(x_i) = 1 \\ 0 & h(x_i) = 0 \\ -\frac{W^+}{W^-} w_0(i) & y_i h(x_i) = -1 \end{cases} \qquad i = 1, \ldots, m;$$

$$\mathcal{D}^0[w_0](i) = \begin{cases} 0 & y_i h(x_i) = 1 \\ w_0(i) & h(x_i) = 0 \\ -\frac{W^0}{W^-} w_0(i) & y_i h(x_i) = -1 \end{cases} \qquad i = 1, \ldots, m.$$

Moreover, functions α and β satisfy the equation

$$a\alpha^2 + d\alpha\beta + b\beta^2 - \alpha = 0,$$

where $a = W^+,\ b = W^-$ and $d = W^0$.

Proof. The equations follow from the obvious relations

$$\frac{\alpha}{\beta^2} = Ze^{-t},$$

$$\alpha - \frac{1}{a} = -\frac{b}{a}\frac{1}{Ze^{-t}} - \frac{d}{a}\frac{1}{Z}.$$

□

As in the first case when classifier takes only two values, $w_t \to \mathcal{L}[w_0]$ when $t \to +\infty$. In the present case with three values, the orbit of w_0 lies in a two dimensional plane on a second degree algebraic curve.

Lemma 3.3. *Assume that the AdaBoost flow runs on the infinite time interval with the same $v(H_t, w_t)$ i.e. it comes from the same classificator. Let*

$$V_{\max} = \max_k y_k v(H_t, w_t)(x_k) \qquad \textit{and} \qquad V_{\min} = \min_k y_k v(H_t, w_t)(x_k).$$

Then, $\frac{d}{dt}\sigma(t) < 0$ and $\lim_{t\to-\infty} \sigma(t) = V_{\max}, \quad \lim_{t\to+\infty} \sigma(t) = V_{\min}$.

Proof. By Jensen's inequality and the strict convexity of the function x^2, we get

$$\begin{aligned} \frac{d}{dt}\sigma(t) &= \sum_{p=1}^{m} y_p v(H_t, w_t)(x_p) \left[-y_p v(H_t, w_t)(x_p) w_t(p) + \sigma_t w_t(p)\right] = \\ &= -\sum_{p=1}^{m} \left[y_p v(H_t, w_t)(x_p)\right]^2 w_t(p) + \sigma_t^2 < 0; \end{aligned}$$

The rest can be proved easily. □

The next result for the Lyapunov function is central for our discussion. This identity is a continuum analog of 2.6.

Theorem 3.4. *If $p < t$, then*

$$\log \mathcal{E}(H_t, w_p) - \log \mathcal{E}(H_p, w_t) = -\int_p^t \sigma_s \, ds.$$

Proof. Assume that the AdaBoost flow runs on the time interval $[p, t]$ with the same $v(H_s, w_s)$ *i.e.* it comes from the same classificator h_γ. In general, the interval $[p, t]$ can be split into subintervals with this property. Differential equations imply two identities

$$H_t = H_p + \int_p^t v(H_s, w_s) \, ds;$$

on such subintervals, and for any $k = 1, ..., m$;

$$w_t(k) = w_p(k) \, e^{-\int_p^t y_k v(H_s, w_s)(x_k) ds} \, e^{\int_p^t \sigma_s ds}.$$

Therefore,

$$\mathcal{E}(H_t, w_p) = \sum_k w_p(k) e^{-y_k H_t(x_k)} = \sum_k w_p(k) e^{-\int_p^t y_k v(H_s, w_s)(x_k) ds} e^{-y_k H_p(x_k)} =$$

$$= \sum_k w_t(k) e^{-\int_p^t \sigma_s ds} e^{-y_k H_p(x_k)} = \mathcal{E}(H_p, w_t) \, e^{-\int_p^t \sigma_s \, ds}.$$

□

Using the fact that, $\mathcal{E}(H_0, w_p) = 1$, we obtain the analog of 2.4

$$\frac{d}{dt} \log \mathcal{E}(H_t, w_0) = -\sigma_t, \tag{3.4}$$

and the analog of 2.5

$$\mathcal{E}(H_T, w_0) = e^{-\int_0^T \sigma_s \, ds}. \tag{3.5}$$

The last identity implies that one should try to choose γ_t so that σ_t is maximal along the path. In fact, there are a few choices. As it will be shown below, they correspond to the discrete AdaBoost algorithms, the arc-gv algorithm and the AdaBoost with varying confidence level.

3.2. **Embedding of the discrete AdaBoost into the AdaBoost flow.** In this section we assume that all weak classifiers h_γ, $\gamma \in \Gamma$; take only two values $-c$ and $+c$, $c > 0$. The formulas obtained in this section will be generalized for the case of classificators with varying confidence level. For each classificator, we define

$$W^- = w_0\{i : y_i h_\gamma(x_i) = -c\} \qquad W^+ = w_0\{i : y_i h_\gamma(x_i) = +c\}.$$

We also assume that $1/2 < W^+ < 1$.

Theorem 3.5. *Let the AdaBoost flow runs up to time Δ with fixed $v(H_t, w_t)$ i.e. it comes from a fixed classificator h_γ. Then,*

(i) For any $\Delta > 0$,

$$e^{-\int_0^\Delta \sigma_s\, ds} \geq 2\sqrt{W^+W^-}. \tag{3.6}$$

(ii) The equality in 3.6 holds if and only if

$$\Delta = \frac{1}{2c}\log\frac{W^+}{W^-}. \tag{3.7}$$

(iii) The equality in 3.6 holds for some $\Delta > 0$ if and only if $\sigma_\Delta = 0$.

Proof. (i.) By 3.5,

$$e^{-\int_0^\Delta \sigma_s\, ds} = \sum_{k=1}^m w_0(k)e^{-\Delta y_k v(x_k)} = W^+e^{-\Delta c} + W^-e^{\Delta c}.$$

Inequality 3.6 follows from the inequality between the arithmetic and geometric means.

(ii.) Write,

$$Z(\Delta, c) = e^{-\int_0^\Delta \sigma_s\, ds}.$$

If left hand side of 3.6 attains its minimum and equality holds, then

$$\frac{\partial Z}{\partial \Delta} = 0.$$

This implies the formula 3.7. The converse statement can be checked directly.

(iii.) The condition $\sigma_\Delta = 0$ means that

$$\sigma_\Delta = W^-e^{\Delta c} - W^+e^{-\Delta c} = 0,$$

This is equivalent to 3.7 and so to 3.6 as well. □

To explain the connection between continuous time system and the AdaBoost algorithm we assume that $c = 1$. One can define the values of the control $\gamma_t : [0, +\infty) \to \Gamma$, recurrently by the following procedure. Given w_0 and for $t_{-1} = 0$, define

$$\gamma_0 = \gamma_{t_{-1}} = \arg\min_{\gamma\in\Gamma} W^-(h_\gamma, w_0). \tag{3.8}$$

Supposed that $W^- = W^-(h_{\gamma_0}, w_0) < 1/2$. Then $\sigma_0 = W^+ - W^- > 0$ decays with time. The AdaBoost flow runs with this γ_0 until the time $t_0 = \Delta$; that can be computed from 3.7. Note that $\sigma_\Delta = 0$. Therefore, we define $\gamma_t = \gamma_0$ for $t \in [0, t_0)$. It is interesting to check that

$$w_{t_0}\{i : y_i h_{\gamma_0}(x_i) < 0\} = \frac{1}{2}.$$

For the next step one should look for a new solution of the minimization problem 3.8 with w_0 replaced by w_{t_0}, *etc.*

In general, put $t_n = \sum_{p=0}^{n} \Delta_p$, for $n \geq 1$. The corresponding control and errors are

$$\gamma_{t_{n-1}} = \arg\min_{\gamma \in \Gamma} W^-(h_\gamma, w_{t_{n-1}}),$$

and $W_n^- = W^-(h_{\gamma_{t_{n-1}}}, w_{t_{n-1}}) < 1/2$. The intervals Δ_n are determined from 3.7:

$$\Delta_n = \frac{1}{2c} \log \frac{W_n^+}{W_n^-} = \frac{1}{2} \log \frac{1 + 2\beta_{t_{n-1}}}{1 - 2\beta_{t_{n-1}}},$$

where

$$W_n^- = W^-(h_{\gamma_{t_{n-1}}}, w_{t_{n-1}}) = \frac{1}{2} - \beta_{t_{n-1}}.$$

It is easy to see that

$$w_{t_n}\{i : h_{\gamma_{t_{n-1}}}(x_i) y_i = -1\} = \frac{1}{2}.$$

The sequence of $(H_n, w_n) = (H_{t_n}, w_{t_n})$ is also a trajectory of the discrete AdaBoost algorithm.

3.3. **Embedding of the arc-gv algorithm into the AdaBoost flow.** First we formulate a version of the discrete algorithm, see [4].

Assume that we have a classifier

$$H = \sum_{k=0}^{n} t_k h_k,$$

where $t_k > 0$ and $h_k \in \mathcal{H}_0$, for $k = 0, \ldots, n$. Introduce the norm,

$$\|H\| = \sum_{k=0}^{n} t_k;$$

together with the normalized margin of the function H at the point $(x, y) \in X \times \{-1, +1\}$

$$m(x, y; H) = y \frac{H(x)}{\|H\|},$$

and the minimal margin

$$\mu(H) = \min_{(x,y) \in TS} \{m(x, y; H)\}.$$

Let $\mu(0) = -1$ and note the obvious properties of $\mu(H)$. First, $-1 \leqslant \mu(H) \leqslant 1$. Second, $\mu(H) = -1$ if and only if there exists $(x, y) \in TS$ such that $h_k(x) \neq y$ for all $k = 0, \ldots, n$, *i.e.* there is a point that all weak classifiers constituting H, make on error or $H = 0$. Third, assume, that $\mu(H) = 1$, then for all $(x, y) \in TS$ one has $h_k(x) = y$, where $k = 0, \ldots, n$; in other words all weak classifiers are able to separate points without error. We assume that there are no such classifiers at all and $\mu(H) < 1$. Fourth, $\mu(H) > 0$ if and only if for all $(x, y) \in TS$ one has $yH(x) > 0$, *i.e.* all points are classified correctly by the function H.

Now we describe the arc-gv algorithm itself.

Initialization,

- $H_{-1} = 0$,
- $w_0(i) = \frac{1}{m},\ i = 1, \dots, m$,
- $\widetilde{t} \gg 1$ - regularization parameter (large positive number).
- For $n = 0, \dots$

For $n = 0, 1, ...,$

- Choose a weak classifier $h_{\gamma_n} \in \mathcal{H}_0$: $W^-(h_{\gamma_n}, w_n) < \frac{1}{2}$;
- $\beta_n = \frac{1}{2} - W^-(h_{\gamma_n}, w_n)$;
- $\mu_{n-1} = \mu(H_{n-1})$;
- Determine the weight: $t_n = \min\{\tilde{t}, \frac{1}{2}\ln(\frac{1+2\beta_n}{1-2\beta_n}) - \frac{1}{2}\ln(\frac{1+\mu_{n-1}}{1-\mu_{n-1}})\}$;
- If $t_n \leqslant 0$, then the algorithm stops;
- Update the measure: $w_{n+1}(i) = \frac{1}{Z_n}\exp(-t_n y_i h_{\gamma_n}(x_i))w_n(i)$;
- $H_n = H_{n-1} + t_n h_{\gamma_n}$.

The formula for the weight t_n appears, see [4], from the following optimization problem: minimize in $t \in [0; \tilde{t}]$ the function

$$\Theta(t) = \sum_{i=1}^{m} w_n(i) e^{t(-y_i h_{\gamma_n}(x_i) + \mu_{n-1})}.$$

As for the AdaBoost, one finds an exact formula for the optimal t by differentiation. Moreover, for $\mu_{n-1} \neq \pm 1$:

$$Z_n = \sqrt{W_n^- W_n^+}\left(\sqrt{\frac{1-\mu_{n-1}}{1+\mu_{n-1}}} + \sqrt{\frac{1+\mu_{n-1}}{1-\mu_{n-1}}}\right),$$

$$w_{n+1}\{i : h_{\gamma_n}(x_i) \neq y_i\} = \frac{1-\mu_{n-1}}{2}.$$

The embedding of arc-gv into AdaBoost flow is the same as for the discrete AdaBoost algorithm. Note that:

$$\mu_t = \mu(H_t) = \frac{1}{t}\min_{(x,y)\in TS}\{yH_t(x)\}.$$

The formulas for embedding are similar:

$$H_0 = 0;$$

$$\Delta'_n = \min\{\overline{\Delta}, \frac{1}{2}\ln\left(\frac{1+2\beta_{t_{n-1}}}{1-2\beta_{t_{n-1}}}\right) - \frac{1}{2}\ln\left(\frac{1+\mu_{t_{n-1}}}{1-\mu_{t_{n-1}}}\right)\}, \quad n \geqslant 0;$$

where $\overline{\Delta}$ is a large fixed number. If at some moment $\Delta'_n \leqslant 0$, then the algorithm stops.

The general picture is as follows: At the beginning, when $\mu_{t_n} = -1$, we switch classifiers at the equal intervals $\overline{\Delta}$. Then $\mu_t > -1 + \epsilon$ and the algorithm starts to

switch at smaller intervals than $\overline{\Delta}$, but bigger then prescribed by AdaBoost. That happens until $\mu_t \leqslant 0$. At some moment $\mu_t = 0$ which is such that constructed classifier H_t learned how to separate points without error. Finally, as a protection from overfitting, the algorithms stops when $\mu_{t_n} > 2\beta_{t_n}$.

3.4. **Embedding of the CRP algorithm into the AdaBoost flow.** In this section we will show how confidence rated prediction (CRP) of Schapire and Singer, [14], can be embedded into the AdaBoost flow. Let the set of all values of h_γ be $c_j,\ j = 1, ..., p$; and take

$$W^{+,j} = w_0\{i :\ h_\gamma(x_i) = c_j,\ y_i = +1\},$$

and

$$W^{-,j} = w_0\{i :\ h_\gamma(x_i) = c_j,\ y_i = -1\}.$$

Theorem 3.6. *Fix some $\Delta > 0$. Let the AdaBoost flow runs up to time $t = \Delta$ with fixed $v(H_s, w_s)$ i.e. it comes from the fixed classifier h_γ.*

(i) Let $W^{+,j}\, W^{-,j} > 0$ for all $j = 1, ..., p$; then for any c_j

$$e^{-\int_0^\Delta \sigma_s\, ds} \geq \sum_{j=1}^{p} 2\sqrt{W^{+,j}\ W^{-,j}}. \tag{3.9}$$

(ii) Let $W^{+,j}\, W^{-,j} > 0$ for all $j = 1, ..., p$; then the equality holds in 3.9 if and only if

$$c_j = \frac{1}{2\Delta} \log \frac{W^{+,j}}{W^{-,j}}, \qquad\qquad j = 1, ..., p;$$

in which case $\sigma_\Delta = 0$.

(iii) Let $W^{+,j}\, W^{-,j} > 0$ for all $j = 1, ..., p'$; and $W^{+,j}\, W^{-,j} = 0$ for all $j = p' + 1, ..., p$; and if

$$c_j = \frac{1}{2\Delta} \log \frac{W^{+,j}}{W^{-,j}}, \qquad\qquad j = 1, ..., p';$$

then for any $\epsilon > 0$

$$e^{-\int_0^\Delta \sigma_s\, ds} \leq \sum_{j=1}^{p'} 2\sqrt{W^{+,j}\ W^{-,j}} + \epsilon,$$

by an appropriate choice of c_j for all $j = p' + 1, ..., p$.

Proof. (i) It can be verified directly that

$$\int_0^\Delta \sigma_s\, ds = -\log \left[\sum_{k=1}^{m} w_0(k) e^{-\Delta y_k v(x_k)} \right].$$

Therefore,

$$e^{-\int_0^\Delta \sigma_s\,ds} = \sum_{k=1}^{m} w_0(k) e^{-\Delta y_k v(x_k)} = \sum_{j=1}^{p} W^{+,j} e^{-\Delta c_j} + W^{-,j} e^{\Delta c_j}.$$

Inequality 3.9 follows from the inequality between arithmetic and geometric means.

Parts *(ii)* and *(iii)* follow from this formula similar to the proof of Theorem 3.5. □

The theorem suggests the following procedure. We put $\Delta_p = 1$ for all $p = 0, 1, 2, ...$ On each round of the boosting procedure, we pick $h_\gamma,\ \gamma \in \Gamma$; such that the corresponding sum

$$Z = \sum_{j=1}^{p} 2\sqrt{W^{+,j}\ W^{-,j}},$$

is minimal over the set of all weak classifiers. By adjusting the values of h_γ according to formulas of Theorem 3.7 we minimize the penalty function $\mathcal{E}$ on this round in an optimal way.

Let us comment on the square roots which appear in the formula for Z. The set of all values of h_γ is a finite set $c_j,\ j = 1, ..., p$. Therefore, we have two special points of $p-1$ dimensional simplex of probability measures

$$p^+ = \frac{1}{W^+}(W^{+,1}, ..., W^{+,p}),$$

and

$$p^- = \frac{1}{W^-}(W^{-,1}, ..., W^{-,p}).$$

It is apparent that

$$Z = \sum_{j=1}^{p} 2\sqrt{W^{+,j}\ W^{-,j}} = 2\sqrt{W^+ W^-}\, BC(p^+, p^-).$$

where $BC(p, q)$ is a Bhattacharyya coefficient, [5], the standard measure a separability of classes in classification.

3.5. **The AdaBoost Flow as a Gradient System.** In this section we write equations 3.1–3.2 in the gradient form and obtain geometrical description of the algorithm.

A function $D(v \,||\, w)$ where $v, w \in \mathcal{W}$ defined on a space $\mathcal{W}$, is called a divergence function, [1], when it satisfies the following conditions:

1) $D(v \,||\, w) \geq 0$.

2) $D(v \,||\, w) = 0$, when and only when $v = w$.

3) For small dw, Taylor expansion gives

$$D(w + dw \,||\, w) \approx \frac{1}{2} \sum g_{ij} dw(i) dw(j),$$

where $g_{ij} = g_{ij}(w)$ is a positive-definite matrix. A divergence is not a distance because it is not necessarily symmetric with respect to v and w, and it does not satisfy the triangular inequality.

The Kulback-Leibler divergence

$$KL(v \,||\, w) = \sum_{k=1}^{m} v(k) \log \frac{v(k)}{w(k)}$$

expanded near diagonal, see [1], produces the metric

$$g_{ij} = \frac{\delta_i^j}{w(i)}.$$

Using substitution $w_t(k) = r_t^2(k)$ and $\lambda_t(k) = y_k v(H_t, w_t)(x_k)$ equations 3.2 can be reduced to the form

$$\frac{d}{dt}\, r_t(k) = -\frac{1}{2}\,(l_t(k) - \sigma_t)\, r_t(k), \qquad\qquad k = 1, 2, ..., m.$$

Moser, [10], noted that on the surface of the sphere

$$\sum_{p=1}^{m} r_t^2(p) = 1, \qquad\qquad r_t(p) > 0;$$

these equations can be put in the gradient form

$$\frac{d}{dt}\, r_t(k) = -\frac{\partial V}{\partial r_t(k)}, \qquad\qquad k = 1, 2, ..., m;$$

with

$$V = \frac{\sum_{p=1}^{m} l_t(p) r_t^2(p)}{4 \sum_{p=1}^{m} r_t^2(p)}.$$

From this fact we have immediately

Lemma 3.7. *Equations 3.2 can be written in the gradient form*

$$\frac{d}{dt}\, w_t(k) = -\nabla^k V = -\sum_{j=1}^{m} g^{kj} \frac{\partial V}{\partial w(j)}, \qquad\qquad k = 1, 2, ..., m;$$

with the metric g^{kj} defined as

$$g^{kj} = \delta_k^j w_t(k),$$

and the potential function

$$V_h(w) = \frac{\sum_{p=1}^{m} l_t(p) w_t(p)}{\sum_{p=1}^{m} w_t(p)}.$$

In other words the Ada Boost flow on measures 3.2 is a gradient flow produced by the function V_h with respect to the Kulback-Leibler metric. Now we can give an invariant geometric description of the AdaBoost algorithm.

Consider a foliation of R^m_+, $\mathcal{W} \in R^m_+$, by level sets of the function

$$V_h(w) = \frac{\sum_{p=1}^m y_p h(x_p) w(p)}{\sum_{p=1}^m w(p)}, \qquad\qquad w \in R^m_+.$$

Namely, each leaf of the foliation is defined as

$$\{w \in R^m_+ : V_h(w) = c\},$$

where c is an arbitrary constant.

Initial step of the algorithm. Pick w_0 and chose h_{γ_0} such that

$$V_{h_{\gamma_0}}(w_0) > 0$$

takes maximal value. Namely,

$$V_{h_{\gamma_0}}(w_0) = \max_{h_0 \in \mathcal{H}_0} V_{h_0}(w_0).$$

Run the AdaBoost flow 3.1–3.2 with this h_{γ_0} until the moment t_0 when the flow reaches the leaf $\{w : V_{h_{\gamma_0}}(w) = 0\}$. Take $H_{t_0} = t_0 h_{\gamma_0}$ as a result of the initial step.

Repeat the procedure with w_0 replaced by w_{t_0}, *etc.*.

3.6. **General geometric approach to boosting.** The idea to define updates of the measure through solutions of a differential equation seems to be very general. To define boosting algorithm one needs three ingredients (i) the metrics; (ii) the potential function; (iii) the rule for changing a potential function.

Let us illustrate how this approach produces Logitboost algorithm. We modify in the original AdaBoost algorithm the metrics and the potential function. Define binary Kulback-Leibler divergence

$$BKL(v \,||\, w) = \sum_{k=1}^m \left[v(k) \log \frac{v(k)}{w(k)} + (1 - v(k)) \log \frac{1 - v(k)}{1 - w(k)} \right].$$

It produces the metrics

$$g_{ij} = \delta^j_i \left[\frac{1}{w(i)} + \frac{1}{1 - w(i)} \right].$$

Taking,

$$V_h(w) = \sum_{p=1}^m y_p h(x_p) w(p)$$

we obtain logistic equations

$$\frac{d}{dt}\, w_t(k) = -\nabla^k V_h = -y_k h(x_k)(1 - w_t(k)) w_t(k).$$

The explicit formulas for a solution

$$w_t(k) = \frac{w_0(k)}{(1 - w_0(k))e^{y_k h(x_k)t} + w_0(k)}$$

produce standard formulas for updates of the measure on each step, see [7].

As we see well known algorithms correspond the cases when differential equations have an explicit formulas for solution. In the case of general metrics obtained from the Bregman divergence an explicit formulas for updates of the measure may not exist. In these case one can solve differential equations by some numerical procedure.

Now we want to present the SuperBoost, a different modification of the AdaBoost algorithm corresponding to a new choice of control. The metric and the potential function are the same as for the AdaBoost. It is a greedy algorithm which for each moment of time $t \geq 0$ chooses a weak classifier h with the largest $\sigma_t(h)$.

Initialization,

- $H_{-1} = 0$.
- $w_0(i) = \frac{1}{m},\ i = 1, \ldots, m$.
- Choose weak classificator $h_{\gamma_0} \in \mathcal{H}_0$ such that : $\sigma_{w_0}(h_{\gamma_0}) = \max_{h \in \mathcal{H}_0} \sigma_{w_0}(h)$.

Updates are occurring on each infinitesimal step $t \to t + dt$

- Change classifier $h_{\gamma_t} = h$ on a new $h_{\gamma_{t+dt}} = h'$ if

$$\sigma_t(h) = \sigma_t(h') \quad \text{and} \quad \frac{d}{dt}\,\sigma_t(h) < \frac{d}{dt}\,\sigma_t(h').$$

- Update the measure: $w_{t+dt}(i) = \frac{1}{Z}\exp(-dty_i h_{\gamma_t}(x_i))w_t(i)$.
- Update the resulting classifier: $H_{t+dt} = H_t + dt \times h_{\gamma_t}$.

It is obvious that SuperBoost algorithm for each finite time interval $[0, T]$ updates the weak classifier only a finite number of times.

Gradient dynamics on measures defined above is different from the restricted gradient minimization procedure for the cost function proposed in [3, 8]. Their procedure is connected to the choice of new weak classifier on each step of boosting or in our language to the specification of control.

3.7. **Boosting and Perelman's ideas for the Ricci flow.** In this section we present the most unexpected consequence of our approach. We describe a striking similarity between the AdaBoost flow and Perelman's ideas, [11], to control the Ricci flow

$$\frac{d}{dt}\,g_t = -2Ric_{g_t} \tag{3.10}$$

where Ric_g is the Ricci tensor of the metric g and $g \in \mathcal{M}$ space of metrics on a Riemannian manifold M. In our notations we follow [2] and [17]. The equation

describes some optimization procedure in the space $\mathcal{M}$. Perelman extends that phase space. Namely he defines Gibbsian type measure dw on M as

$$dw = e^{-f} dV_g,$$

where dV_g is a volume element constructed from the metric g. Apparently for given g the measure dw can be identified with the potential function f. Now the flow is defined on the extended phase space $\mathcal{M} \times C^\infty$. In order to control singularities of the Ricci flow $g_t,\ t \geq 0$; Perelman chooses the potential function f in a special way determined by dynamics of the metric. The system of coupled equations for the metric g and potential function f

$$\frac{d}{dt} g_t = -2(Ric_{g_t} + Hess_{g_t} f_t), \tag{3.11}$$

$$\frac{d}{dt} f_t = -R_{g_t} - \Delta f_t, \tag{3.12}$$

where R_g is the scalar curvature of the metric g, leads to

$$\frac{d}{dt}(dw) = \frac{d}{dt}\left(e^{-f_t} dV_{g_t}\right) = 0.$$

The flow defined by 3.11 is the original Ricci flow 3.10 up to a time dependent diffeomorphism.

These equations are analog of the AdaBoost flow equations 3.1 and 3.2. To be precise equation 3.12 is an analog of 3.3 which is an equivalent form of 3.2. Moreover, these equations are similar termwise. The scalar curvature R_{g_t} in 3.12 plays the role similar to that of the margin $y_k v(H_t, w_t)(x_k)$ in 3.3. The Laplacian Δf_t in 3.12 is similar to the term σ_t in 3.3.

On the extended phase space $\mathcal{M} \times C^\infty$ Perelman defines the following functional

$$\mathcal{F}(g, f) = \int_M \left(R_g + |\bigtriangledown f|^2\right) e^{-f} dV_g.$$

Perelman calls the functional $\mathcal{F}(g, f)$ entropy for the Ricci flow. The functional $\mathcal{F}(g, f)$ increases along trajectories of the Ricci flow. Indeed, the formula

$$\frac{d}{dt}\mathcal{F}(g_t, f_t) = \int_M \langle -Ric_{g_t} - Hess_{g_t} f_t, \frac{dg_t}{dt}\rangle e^{-f_t} dV_{g_t},$$

together with equations 3.11 and 3.12 leads to

$$\frac{d}{dt}\mathcal{F}(g_t, f_t) = 2\int_M |Ric_{g_t} + Hess_{g_t} f_t|^2 e^{-f_t} dV_{g_t} \geq 0. \tag{3.13}$$

The functional $\mathcal{F}(g, f)$ is an analog of the Lyapunov function $\mathcal{E}(H, w)$. As we saw in section 3.2 the functional $\mathcal{E}(H, w)$ for the Ada Boost flow is closely connected to the ordinary Kullback-Leibler entropy. It steadily decreases along trajectories of the AdaBoost flow.

It is time to take stock of these similarities. The dictionary between two problems is below

TS training set	M Riemannian manifold
$\mathcal{H}$ cone over the set of classifiers	$\mathcal{M}$ space of Riemannian metrics
$\mathcal{H} \times W$ phase space of the AdaBoost flow	$\mathcal{M} \times C^\infty$ phase space of the controlled Ricci flow
$\frac{d}{dt}\ \lambda_t^k = v^k(H_t, w_t)$	$\frac{d}{dt}\, g_t = -2(Ric_{g_t} + Hess_{g_t} f_t)$
$\frac{d}{dt}\ f_t(k) = y_k v(H_t, w_t)(x_k) - \sigma_t$	$\frac{d}{dt}\, f_t = -R_{g_t} - \Delta f_t$
$\mathcal{E}(H, w)$	$\mathcal{F}(g, f)$
$\frac{d}{dt} \log \mathcal{E}(H_t, w_0) = -\sigma_t$	$\frac{d}{dt}\mathcal{F}(g_t, f_t) \geq 0$

Such coincidence is not accidental. We refer to the introduction where it is explained that the AdaBoost and the Perelman's construction are different realizations of the same idea.

References

[1] S. Amari and H. Nagaoka. Methods of information geometry. American Mathematical Soc., 2000.

[2] Anderson, M. Geometrisation of 3-manifolds via the Ricci Flow. Notices of AMS, 51(2): 184–193, 2004.

[3] Mason, L., Baxter, J., Bartlett, P. and Frean, M. Functional gradient techniques for combining hypotheses. In Advances in Large Margin Classifiers (A. Smola, P. Bartlett, B. Scholkopf and D. Schuurmans, eds.). MIT Press, Cambridge. (2000).

[4] Breiman, L. Prediction games and arcing algorithms. Neural Computation, 11(7):1493–1517, 1999.

[5] Bhattacharyya, A. On a measure of divergence between two statistical populations defined by their probability distributions. Bulletin of the Calcutta Mathematical Society 35: 99-109. (1943).

[6] Freund Y. and Schapire R., A decision-theoretic generalization of on-line learning and an application to boosting. J. Comput. System Sci., 55(1):119–139, August 1997.

[7] Freund Y. and Schapire R., Boosting: Foundations and Algorithms. MIT press 2012.

[8] Friedman J., Greedy function approximation: A gradient boosting machine. Annals of Statistics, 29: 1189–1232, 2000.

[9] Kivinen J., and Warmuth M., Boosting as entropy projection. In Proc. 12th Annual Conference on Computational Learning Theory, 134–144, ACM, New York, July 1999.

[10] Moser, J. Finitely many mass points on the line under the influence of an exponential potentialan integrable system. Dynamical Systems, Theory and Applications. (Rencontres, BattelleRes. Inst., Seattle, Wash., 1974), pp. 467497. Lecture Notes in Phys., Vol. 38, Springer, Berlin, 1975.

[11] G. Perelman The entropy formula for the Ricci flow and its geometric interpretation. preprint 2002, DG/0211159

[12] Pontryagin, L. S. The mathematical theory of optimal processes and differential games. (Russian) Topology, Ordinary Differential Equations, Dynamical Systems. Trudy Mat. Inst. Steklov. 169 (1985), 119-158, 254-255.

[13] Schapire R. The strength of weak learnability. 1990, Mach Learn 5(2):197-227.

[14] Schapire R., Singer Y., Improved boosting algorithms using confidence-rated predictions. Machine Learning 37(3): 297–336 (1999)

[15] Toda, M. Theory of Nonlinear Lattices. Second edition. Springer Series in Solid-State Sciences, 20. Springer-Verlag, Berlin, 1989. x+225 pp.

[16] Top 10 algorithms in data mining. Knowledge and Information Systems, vol. 14, 2008, 1-37.

[17] Topping P.M. Lectures on the Ricci flow. L.M.S. Lecture note series 325 C.U.P, 2006.

[18] Varadhan, S.R.S. Large Deviations and Applications. 75 pages, 1984, SIAM.

[19] Vaninsky, K. The Atiyah-Hitchin bracket and the open Toda lattice. J. Geom. Phys. 46 (2003), no. 3-4, 283307.

A.L and S.M
Faculty of Mathematics and Mechanics
Moscow State University
Vorobjevy Gory
Moscow
Russia

stepan.muzychka@gmail.com
alekslyk@yandex.ru

K.V.
Department of Mathematics
Michigan State University
East Lansing, MI 48824
USA

vaninsky@math.msu.edu