M-Power Regularized Least Squares Regression

Julien Audiffren, Hachem Kadri

M-Power Regularized Least Squares Regression

Julien Audiffren[1] and Hachem Kadri[1]

[1]QARMA, Aix-Marseille Université, CNRS, LIF, 13000, Marseille, France

Résumé

Regularization is used to find a solution that both fits the data and is sufficiently smooth, and thereby is very effective for designing and refining learning algorithms. But the influence of its exponent remains poorly understood. In particular, it is unclear how the exponent of the reproducing kernel Hilbert space (RKHS) regularization term affects the accuracy and the efficiency of kernel-based learning algorithms. Here we consider regularized least squares regression (RLSR) with an RKHS regularization raised to the power of m, where m is a variable real exponent. We design an efficient algorithm for solving the associated minimization problem, we provide a theoretical analysis of its stability, and we compare it with respect to computational complexity, speed of convergence and prediction accuracy to the classical kernel ridge regression algorithm where the regularization exponent m is fixed at 2. Our results show that the m-power RLSR problem can be solved efficiently, and support the suggestion that one can use a regularization term that grows significantly slower than the standard quadratic growth in the RKHS norm.

1 Introduction

Regularization is extensively used in learning algorithms. It provides a principled way of addressing the well-known overfitting problem by learning a function that balances fit and smoothness. The idea of regularization is hardly new. It goes back at least to [Tik63], where it is used for solving ill-posed inverse problems. Recently, there has been substantial work put forth to develop regularized learning models and significant progress has been made. Various regularization terms have been suggested, and different regularization strategies have been proposed to derive efficient learning algorithms. Among these algorithms one can cite regularized kernel methods which are based on a regularization over reproducing kernel Hilbert spaces (RKHSs) [SS02, STC04].

A considerable amount of flexibility for fitting data is gained with kernel-based learning, as linear methods are replaced with nonlinear ones by representing the data points in high dimensional spaces of features, specifically RKHSs. Many learning algorithms based on kernel methods and RKHS regularization [SS02], including support vector machines (SVM) and regularized least squares (RLS), have been used with considerable success in a wide range of supervised learning tasks, such as regression and classification. However, these algorithms are, for the most part, restricted to a RKHS regularization term with an exponent equal to two. The influence of this exponent on the performance of kernel machines remains poorly understood. Studying the effects of varying the exponent of the RKHS regularization on the regularization process and the underlying learning algorithm is the main goal of this research.

To the best of our knowledge, the most directly related work to this paper is that of Mendelson and Neeman [MN10] and Steinwart et al. [SHS$^+$09], who studied the impact of the regularization exponent on RLS regression (RLSR) methods from a theoretical point of view. In [MN10] the sharpest known learning rates of the RLSR algorithm was established in the case where the exponent of the regularization term m is less than or equal to one, showing that one can use a regularization term that grows slower than the standard quadratic growth in the RKHS norm. In [SHS$^+$09] an oracle inequality that holds for all $m \geq 1$ was provided, arguing that the exponent m may be chosen on the basis of algorithmic considerations. In this spirit we have asked whether, by additionally focusing attention on the algorithmic problem involved in the optimization, one could develop

an efficient algorithm for RLSR with a variable RKHS regularization exponent. The remainder of the paper is devoted to presenting an approach to answering this question.

In this work we demonstrate that the m-power RLS regression problem can also be solved efficiently and that there is no reason for ignoring this possibility. Specifically, we make the following contributions :

- we derive a semi-analytic expression of the solution of the regularized least squares regression problem when the RKHS regularization is raised to the power of m, where m is a variable real exponent, and we design an efficient algorithm for computing the solution (Section 3),
- we show that the proposed algorithm, called M-RLSR (m-power RLS regression), and the kernel ridge regression (KRR) algorithm, although they give the same solution under particular conditions, are not equivalent, and thus have not the same theoretical and practical properties (Section 4),
- we establish a theoretical result indicating that the M-RLSR algorithm is β-stable when $m \geq 2$ (Section 5),
- we experimentally evaluate the proposed algorithm and compare it to KRR with respect to speed of convergence and prediction accuracy (Section 6).

2 Problem Overview

This section presents the notation we use throughout the paper and introduces the problem of m-power regularized least squares regression.

Notation. Let $m > 0$ be a real number, $\mathcal{X}$ a polish space, $\mathcal{H} \subset \mathbb{R}^{\mathcal{X}}$ a separable reproducing kernel Hilbert space (RKHS), and $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ its positive definite kernel. For all set of n elements of $\mathcal{X} \times \mathbb{R}$, we denote by $Z = \{(x_1, y_1), .., (x_n, y_n)\}$ the training set, and by K the Gram matrix associated to k for Z with $(K_Z)_{i,j} = k(x_i, x_j)$. Finally, let $Y = (y_1, ..., y_n)^\top$ be the output vector.

M-power RLS regression. The algorithm we investigate here combines a least squares regression with an RKHS regularization term raised to the power of m. Formally, we would like to solve the following optimization problem :

$$f_Z = \underset{f \in \mathcal{H}}{\arg\min} \frac{1}{n} \sum_{i=1}^{n} (y_i - f(x_i))^2 + \lambda \|f\|_{\mathcal{H}}^m, \tag{1}$$

where m is a suitable chosen exponent. Note that the classical kernel ridge regression (KRR) algorithm [SGV98] is recovered for $m = 2$.

The problem (1) is well posed for $m > 1$. Indeed, the function to minimize is strictly convex, coercive and continuous, hence it has a unique minimum. For $m = 1$ the problem is not strictly convex and the solution is then not necessarily unique, while for $m < 1$ the problem is no longer convex, so the function to minimize may not have a global minimum and the existence of a solution is not guaranteed.

To solve the problem (1), one can perform a gradient descent algorithm to minimize the objective function in the primal, e.g. using a Newton method as in [Cha07]. However, because $m \neq 2$, this requires the storage and the inversion of a Hessian of size $O(n^2)$ at each step [1] and becomes computationally infeasible. In the next section, we introduce a novel fast m-power RLS regression algorithm, generalizing the kernel ridge regression algorithm to an arbitrary regularization exponent.

In addition, it is crucial to note that even though there is a certain equivalence between the M-RLSR and KRR optimization problems in the case of $m > 1$, the underlying algorithms are not equivalent, and thus have not the same theoretical and practical properties. Section 4 is devoted to this purpose. This does not occur when $m < 1$. Indeed, the objective function of the M-RLSR problem defined in (1) is not convex for $m < 1$, and the two optimization problems then become different.

1. For the particular case of $m = 2$, the Hessian does not depend on the function f to minimize and therefore needs to be computed only once, see [Cha07] for more details.

Algorithm 1 M-Power RLS Regression Algorithm (M-RLSR)

Input : training data $Z = \{(x_1, y_1), .., (x_n, y_n)\}$, parameter $\lambda \in \mathbb{R}_+^*$, exponent $m \in \mathbb{R}_+^*$

1. **Kernel matrix :** Compute the Gram matrix K from the training set Z
$$K = \big(k(x_i, x_j)\big)_{1 \leq i,j \leq n}$$
2. **Matrix diagonalization :** Diagonalize K in an orthonormal basis
$$K = QDQ^\top \quad ; \quad d_i = D_{ii} \ , \ \forall 1 \leq i \leq n$$
3. **Change of basis :** Perform a basis transformation
$$Y = Q^\top Y \quad ; \quad y_i = Y_i \ , \ \forall 1 \leq i \leq n$$
4. **Root-finding :** Find the root C_0 of the function F defined in (5)
We employ a Newton method
5. **Solution :** Compute α from (4) and reconstruct the weights
$$(\alpha_i)_{1 \leq i \leq n} = \frac{2y_i}{2d_i + \lambda mnC_0} \quad \text{and} \quad \alpha = Q\alpha$$

3 M-Power Regularized Least Squares Regression Algorithm

We now provide an efficient learning algorithm solving the m-power regularized least squares problem. It is worth recalling that the minimization problem (1) with $m = 2$ becomes a standard kernel ridge regression, which has an explicit analytic solution. In the same spirit, the main idea of our algorithm is to derive analytically from (1) a reduced one-dimensional problem on which we apply a root-finding algorithm.

First notice that, the objective function to minimize is Gâteaux differentiable in every direction. Thus, since f_Z is a minimum, we have :

$$0 = \sum_{i=1}^{n} -2k(., x_i)(y_i - f_Z(x_i)) + \lambda mn \|f_Z\|_{\mathcal{H}}^{m-2} f_Z,$$

i.e.,

$$f_Z = \sum_{i=1}^{n} 2k(., x_i) \frac{y_i - f_Z(x_i)}{\lambda mn \|f_Z\|_{\mathcal{H}}^{m-2}}.$$

That is to say, f_Z can be written in the following form :

$$f_Z = \sum_{i=1}^{n} \alpha_i k(., x_i), \tag{2}$$

with $\alpha_i \in \mathbb{R}$. Notice, that we have recovered exactly the form of the representer theorem, which can also be derived from a result due to Dinuzzo and Schölkop [DS12]. Now by combining (1) and (2), the initial problem becomes

$$\alpha = \underset{a \in \mathbb{R}^n}{\arg\min} (Y - Ka)^\top (Y - Ka) + n\lambda (a^\top Ka)^{m/2}, \tag{3}$$

where $\alpha = (\alpha_i)_{1 \leq i \leq n}$ is the vector to determine. The following theorem gives an explicit formula for α that solves the optimization problem (3).

Theorem 1 *Let Q an orthonormal matrix and D a diagonal matrix such that $K = QDQ^\top$. Let y_i' be the coordinates of $Q^T Y$, $(d_i)_{1 \leq i \leq n}$ the elements of the diagonal of D, $C_0 \in \mathbb{R}_+$ and $m > 1$. Then the vector $\alpha = Q\alpha'$ with*

$$\alpha_i' = \frac{2y_i'}{2d_i + \lambda mnC_0} \ , \ \forall 1 \leq i \leq n \ , \tag{4}$$

is the solution of (3) *if and only if* C_0 *is the root of the function* $F : \mathbb{R}_+ \to \mathbb{R}$ *defined by*

$$F(C) = \big(\sum_{i=1}^{n} \frac{4d_i y_i'^2}{(2d_i + \lambda mnC)^2}\big)^{m/2-1} - C. \tag{5}$$

Proof : By computing the Gâteaux derivative of the objective function to minimize in (3), we obtain that α must verify

$$Y = K\alpha + \lambda \frac{mn}{2}(\alpha^\top K\alpha)^{m/2-1}\alpha.$$

Then, since K is symmetric and positive semidefinite, $\exists Q$ an orthonormal matrix (the matrix of the eigenvectors) and D a diagonal matrix with eigenvalues $(d_i)_{1\leq i\leq n} \geq 0$ such that $K = QDQ^\top$. Hence,

$$\begin{aligned} Y &= QDQ^\top\alpha + \lambda\frac{mn}{2}((Q^\top\alpha)^\top D(Q^\top\alpha))^{m/2-1}\alpha \\ \Rightarrow Q^\top Y &= DQ^\top\alpha + \lambda\frac{mn}{2}((Q^\top\alpha)^\top D(Q^\top\alpha))^{m/2-1}Q^\top\alpha. \end{aligned}$$

Given this, one can define a new representation by changing the basis such that $Y' = Q^\top Y$ and $\alpha' = Q^\top\alpha$. We obtain

$$Y' = D\alpha' + \lambda\frac{mn}{2}(\alpha'^\top D\alpha')^{m/2-1}\alpha'.$$

Now if we write the previous equation for every coefficient of the vectors, we obtain that

$$\left\{ y_i' = d_i\alpha_i' + \lambda\frac{mn}{2}(\sum_{j=1}^{n} d_j\alpha_j'^2)^{m/2-1}\alpha_i' \quad , \quad \forall 1 \leq i \leq n. \right.$$

Note that $(\sum_{j=1}^{n} d_j\alpha_j'^2)^{m/2-1}$ is the same for every equation (i.e. it does not depend on i), so we can rewrite the system as follows, where $C \in \mathbb{R}$

$$\left\{ \begin{array}{l} C = \big(\sum_{j=1}^{n} d_j\alpha_j'^2\big)^{m/2-1} \\ \quad \text{and} \\ \quad \alpha_i' = \dfrac{2y_i'}{2d_i + \lambda mnC} \quad , \quad \forall 1 \leq i \leq n. \end{array} \right. \tag{6}$$

which is well defined if $d_i + \lambda mnC \neq 0$, which is the case when $C > 0$. Since $C \geq 0$ by definition, the only possibly problematic case is $C = 0$, but this implies that $Y = 0$, which is a degenerated case. Now we just need to calculate C, which verifies :

$$C = \big(\sum_{i=1}^{n} d_i\alpha_i'^2\big)^{m/2-1} = \big(\sum_{i=1}^{n} \frac{4d_i y_i'^2}{(2d_i + \lambda mnC)^2}\big)^{m/2-1}.$$

Thus to obtain an explicit value for α', we need only to find a root of the function F defined as follows :

$$F(C) = \big(\sum_{i=1}^{n} \frac{4d_i y_i'^2}{(2d_i + \lambda mnC)^2}\big)^{m/2-1} - C.$$

We have proven that any solution of (3) can be written as a function of C_0, a root of F. But for $m > 1$, F is strictly concave, and $F(0) > 0$, hence it has at most one root in $\mathbb{R}_+$. Thus since $\lim_{C\to+\infty} F(C) = -\infty$, F has exactly one root, which proves Theorem 1. □

In Theorem 1, we have shown that F has a unique root C_0 and that the solution of the optimization problem (1) is expressed analytically as a function of C_0. It is important to note that F is a function from $\mathbb{R}$ to $\mathbb{R}$, and

then computing C_0 using a root-finding algorithm, e.g. Newton's method, is a fast and accurate procedure. Our algorithm, which we call *m-power RLSR*, uses these results to provide an efficient solution to regularized least squares regression with a variable regularization exponent m (see Algorithm 1).

The case $m \leq 1$. It is important to note that for $m \leq 1$, although the m-power RLS minimization problem (1) is no longer strictly convex, Algorithm 1 can still be applied. Indeed, when $m \leq 1$, the objective function to minimize in (1) can have local extrema and the function F in (5) associated to the solution of (1) may have multiple roots. But, it is easy to see that each extrema (local or global) of the minimization problem (1), corresponds to a root of F. Hence, for $m \leq 1$, the root-finding step of the M-RLSR algorithm is modified as follows to search for the global minima :

1. iterate the newton method ten times starting from ten different initialization values, equally spaced on a logarithmic scale between 1 and 10^4,
2. for each root, calculate the corresponding α using (6),
3. for each α, compute the corresponding error using (1) and (2),
4. choose α with the lowest error.

While this procedure does not guarantee to find the solution of the M-RLSR problem (1), which may not exist when $m \leq 1$, it yields good results in practice (see Section 6 for more details).

Complexity analysis. Here we consider a naive implementation of the m-power RLSR algorithm. Obtaining the Gram matrix has complexity $O(n^2)$, while diagonalizing has complexity $O(n^3)$. The cost of change of basis is $O(n^2)$. The complexity of the root-finding algorithm depends on two parameters : the maximum number of iteration and the precision. In our case, we fix the maximum number of iteration to 500 and the precision to 10^{-20}, and we used the Newton algorithm. Finally, computing α and $Q\alpha$ has complexity $O(n^2)$. Then, the total complexity of a naive implementation of Algorithm 1 is $O(n^3)$.

4 M-Power RLSR Versus KRR

In this section, we examine the relation between m-power regularized least squares regression and kernel ridge regression algorithms. In particular, we show that while the two algorithms may under particular conditions give the same solution, they are not equivalent.

M-RLSR and KRR : are they equivalent ? One crucial issue regarding the interpretation of the M-RLSR algorithm is whether by rescaling the regularization parameter λ, M-RLSR gives the same solution as KRR. Indeed, when $m > 1$, the objective function of the M-RLSR optimization problem (1) is strictly convex, and then by Lagrangian duality it is equivalent to its unconstrained version. In this case, it is possible to find a value of the regularization parameter such that the solution of the M-RLSR minimization corresponds to that of the KRR optimization problem. However, this is not the case when $m \leq 1$. Moreover, even though there is an equivalence between M-RLSR and KRR optimization problems, the underlying algorithms are not necessarily equivalent. In order to explain this claim, the notion of equivalent algorithms need to be clearly defined. In the following we provide two definitions of such equivalence. The first definition, called Z-equivalence, corresponds exactly to the equivalence between the associated minimization problems. By Z-equivalence, we would like to emphasize here that this equivalence holds only for a fixed training set Z. This matches the equivalence between the optimization problems since the objective function is minimized for the set Z of examples $\{(x_i, y_i)\}_{i=1}^n$. The second definition is more general, in the sense that two learning algorithms are equivalent if they always provide the same predictive and optimality guarantees. We show below that, even for $m > 1$, M-RLSR and KRR algorithms are not equivalent but only Z-equivalent. Hence, they do not have the same theoretical and practical performances. To formalize these ideas, let $\mathcal{Z} = \bigcup_{n \geq 1} (\mathcal{X} \times \mathcal{Y})^n$ denotes all possible training set where $X \in \mathcal{X}$ and $Y \in \mathcal{Y}$, and $\mathcal{H} \subset \mathcal{Y}^{\mathcal{X}}$ be a normed vector space. We use here the same definition of a learning algorithm as given by Bousquet and Elisseeff [BE02], but for simplicity we restrict ourselves to learning algorithms associated to strictly convex optimization problems.

Definition 4.1 *A learning algorithm $\mathcal{A}$ is a function $\mathcal{A} : \mathcal{Z} \to \mathcal{H}$ which maps a learning set Z onto a function $A(Z)$, such that*

$$A(Z) = \arg\min_{g \in \mathcal{H}} R(Z, g),$$

where $R(Z, \cdot)$ is a strictly convex objective function.

For simplicity, we use in the following the same notation for a minimization problem and its objective function. In our case, since we consider only strictly convex objective functions, the learning algorithm tries to find the unique solution of the minimization problem. Based on this, an equivalence between two algorithms can be defined as follows.

Definition 4.2 *Let $Z \in \mathcal{Z}$. Two algorithms $\mathcal{A}$ and $\mathcal{B}$ are Z-equivalent if and only if $\mathcal{A}(Z) = \mathcal{B}(Z)$.*

Note that this first definition (Z-equivalence) is directly related to the equivalence between the optimization problems. In other words, let $\mathcal{A}$ (resp. $\mathcal{B}$) be a learning algorithm associated to the optimization problem R (resp. S), then $\mathcal{A}$ and $\mathcal{B}$ are Z-equivalent if and only if S and R are equivalent on Z, that is, the optimal solution of R is the optimal solution of S. It is important to point out that the optimal solutions of R and S are computed for a set Z, and even though they are equal on Z, there is no guarantee that this remains true if Z varies. This means that the two algorithms $\mathcal{A}$ and $\mathcal{B}$ provide the same output with the set Z, but this may not be necessarily the case with another set Z'.

We can now consider a stronger definition of equivalence between algorithms.

Definition 4.3 *Two algorithms $\mathcal{A}$ and $\mathcal{B}$ are equivalent if and only if $\mathcal{A} \equiv \mathcal{B}$ (equality between the functions $\mathcal{A}$ and $\mathcal{B}$ in $\mathcal{H}^{\mathcal{Z}}$).*

With this definition, two algorithms $\mathcal{A}$ and $\mathcal{B}$ are equivalent if they provide identical solutions for any training set Z. That means, applying $\mathcal{A}$ or $\mathcal{B}$ for any theoretical or practical learning purpose is exactly the same. This is in contrast to the Z-equivalence where the two algorithms give the same solution only with identical and fixed training data Z, which does not imply that they have the same theoretical and practical performance, nor the same optimality guarantee. In other words, if two algorithms are equivalent, they define the same function while if they are only Z-equivalent, their respective functions coincide only on a given training set Z. Also, if two algorithms $\mathcal{A}$ and $\mathcal{B}$ are Z-equivalent then $\mathcal{A}(Z) = \mathcal{B}(Z)$ but $\forall Z' \neq Z$ element of $\mathcal{Z}$, nothing can be said regarding $\mathcal{A}(Z')$ and $\mathcal{B}(Z')$. As a consequence, it is easy to see that any property of $\mathcal{A}$ involving varying the training data, such as stability, generalization, or cross-validation, does not bring any information about $\mathcal{B}$. In the following, we show that M-RLSR and KRR algorithms are Z-equivalent, but not equivalent. It is important to note that with Definition 4.1, each value of the regularization parameter, defines a different KRR and M-RLSR algorithms.

Lemma 4.4 *$\forall m > 1$, $\forall Z \in \mathcal{Z}, \exists F_{Z,m} : \mathbb{R}^+ \to \mathbb{R}^+$, bijective, such that $\forall \lambda > 0$, M-RLSR with regularization parameter λ and KRR with regularization parameter $\lambda_2 = F_{Z,m}(\lambda)$ are Z-equivalent. Moreover,*

$$F_{Z,m}(\lambda) = \frac{m}{2} C_0(Z, m, \lambda)\lambda,$$

where $C_0(Z, m, \lambda)$ is the unique root of the function F defined in (5).

Proof : For $m > 1$, the equivalence between constrained and unconstrained strictly convex optimization problems [KBSZ11, Appendix A] implies that $\exists \Gamma_{m,Z,\lambda} > 0$ such that the minimization problem defined by (1) on Z it is equivalent to the following constrained problem :

$$\arg\min_{f \in \mathcal{H}} \frac{1}{|Z|} \sum_{(x,y) \in Z} (y - f(x))^2, \quad \text{s.t. } \|f\|_{\mathcal{H}}^m \leq \Gamma_{m,Z,\lambda}$$

The constrain is equivalent to $\|f\|_{\mathcal{H}}^2 \leq \Gamma_{m,Z,\lambda}^{2/m}$, thus we deduce that $\exists \lambda_2(m, Z, \lambda) > 0$ such that (1) with regularization parameter λ is equivalent to

$$\arg\min_{f \in \mathcal{H}} \frac{1}{|Z|} \sum_{(x,y) \in Z} (y - f(x))^2 + \lambda_2(m, Z, \lambda)\|f\|_{\mathcal{H}}^2,$$

i.e., the KRR minimization problem with a regularization parameter $\lambda_2(m, Z, \lambda)$. Hence M-RLSR with λ is Z-equivalent to KRR with $\lambda_2(m, Z, \lambda)$. It is easy to see from (6) that the function $F_{Z,m}$ that maps λ to the corresponding λ_2 has the form $F_{Z,m}(\lambda) := \frac{m}{2} C_0(Z, m, \lambda)\lambda$. □

Remark 4.5 *It is important to note that since the value of $\lambda_2 = F_{Z,m}(\lambda)$ does depend on lambda, m and Z, the algorithms M-RLSR and KRR are only Z-equivalent and not equivalent. Indeed, let $m > 1$, $m \neq 2$, $\lambda > 0$ and $Z, Z' \in \mathcal{Z}$ such that $C_0(Z, m, \lambda) \neq C_0(Z', m, \lambda)$. Assume that M-RLSR with λ and KRR with λ_2 are Z-equivalent. Then, from Lemma 4.4, we have $\lambda_2 = \lambda m C_0(Z, m, \lambda) \neq \lambda m C_0(Z', m, \lambda)$, and hence the two algorithms are not Z'-equivalent*[2]*. This fact is illustrated by the experiments presented in Subsection 6.1.*

Since M-RLSR and KRR algorithms are only Z-equivalent and not equivalent, theoretical and practical properties involving varying the training set, such as stability, generalization, or cross-validation selection procedure, cannot be retrieved from KRR. It is also worth noting that Lemma 4.4 and the Z-equivalence is only valid for $m > 1$, and when $m <= 1$, M-RLSR and KRR gives rise to different solutions even with the same training set.

5 Stability Analysis

The notion of algorithmic stability, which is the behavior of a learning algorithm following a change of the training data, was used successfully by Bousquet and Elisseeff [BE02] to derive bounds on the generalization error of kernel-based learning algorithms. In this section, we extend the stability results of [BE02] to cover the m-power RLSR algorithm. As mentioned in the previous section, the stability properties of KRR does not imply the stability of M-RLSR, since the two algorithms are only Z-equivalent and not equivalent. We show here that the algorithm is stable for $m \geq 2$.

In this section we denote by $\underline{X}$ and $\underline{Y}$ a pair of random variables following the unknown distribution D of the data, $\underline{X}$ representing the input and $\underline{Y}$ the output, by $Z^i = Z \setminus (x_i, y_i)$ the training set from which was removed the element i. Let $c(y, f, x) = (y - f(x))^2$ denotes the cost function used in the algorithm. For all $f \in \mathcal{H}$, let $R_e(f, Z) = 1/n \sum_{1 \leq i \leq n} c(y_i, f, x_i)$ be the empirical error and $R_r(f, Z) = R_e(f, Z) + \lambda \|f\|^m_{\mathcal{H}}$ be the regularized error. Let us recall the definition of uniform stability.

Definition 5.1 *An algorithm $Z \to f_Z$ is said β uniformly stable if and only if $\forall n \geq 1$, $\forall 1 \leq i \leq n$, $\forall Z$ a realization of n i.i.d. copies of $(\underline{X}, \underline{Y})$,$\forall (x, y) \in \mathcal{X} \times \mathcal{Y}$ a Z independent realization of $(\underline{X}, \underline{Y})$, we have $|c(y, f_Z, x) - c(y, f_{Z^i}, x)| \leq \beta$.*

To prove the stability of a learning algorithm, it is common to make the following assumptions.

Hypothesis 1 *$\exists C_y > 0$ such that $|\underline{Y}| < C_y$ a.s.*

Hypothesis 2 *$\exists \kappa > 0$ such that $\sup_{x \in \mathcal{X}} k(x, x) < \kappa^2$*

Lemma 5.2 *If Hypotheses 1 and 2 hold, then $\forall n \geq 1$, $\forall 1 \leq i \leq n$, $\forall Z$ a realization of n i.i.d. copies of $(\underline{X}, \underline{Y})$,$\forall (x, y) \in \mathcal{X} \times \mathcal{Y}$ a Z independent realization of $(\underline{X}, \underline{Y})$,*

$$|c(y, f_Z, x) - c(y, f_{Z^i}, x)| \leq C |f_Z(x) - f_{Z^i}(x)|,$$

with $C = 2\left(C_y + \kappa\left(\frac{C_y^2}{\lambda}\right)^{\frac{1}{m}}\right)$.

Proof : Since $\mathcal{H}$ is a vector space, $0 \in \mathcal{H}$, and

2. The new value of C_0 with Z' may not change, but we observed experimentally and in simulations that the case when C_0 remains unchanged by varying Z is extremely rare and a degenerate case.

$$\lambda\|f_Z\|^m \leq \frac{1}{n}\sum_{i=1}^{n}(y_i - f_Z(x_i))^2 + \lambda\|f_Z\|_{\mathcal{H}}^m$$
$$\leq \frac{1}{n}\sum_{k=1}^{n}\|y_k - 0\|^2 + \lambda\|0\|_{\mathcal{H}}^m \leq C_y^2,$$

where we used the definition of f_Z as the minimum of (1) and Hypothesis 1. Using the reproducing property and Hypothesis 2, we deduce that

$$|f_Z(x)| \leq \sqrt{k(x,x)}\|f_Z\|_{\mathcal{H}} \leq \kappa\|f_Z\|_{\mathcal{H}} \leq \kappa\left(\frac{C_y^2}{\lambda}\right)^{\frac{1}{m}}.$$

The same reasoning holds for f_{Z^i}. Finally,

$$|c(y, f_Z, x) - c(y, f_{Z^i}, x)| = |(y - f_Z(x))^2 - (y - f_{Z^i}(x))^2|$$
$$\leq 2\left(C_y + \kappa\left(\frac{C_y^2}{\lambda}\right)^{\frac{1}{m}}\right)|f_Z(x) - f_{Z^i}(x)|.$$

□

The stability of our algorithm when $m \geq 2$ is established in the following theorem, whose proof is an extension of Theorem 22 in [BE02]. The original proof concerns the KRR case when $m = 2$. The beginning of our proof is similar to the original one ; but starting from (10), the proof is modified to hold for $m \geq 2$, since the equalities used in [BE02] no longer holds when $m > 2$,. We use inequalities involving generalized Newton binomial theorem instead.

Theorem 2 *Under the assumptions 1 and 2, algorithm $Z \to f_Z$ defined in* (1) *is β stable $\forall m >= 2$ with*

$$\beta = C\kappa\left(2^{m-2}\frac{C\kappa}{\lambda n}\right)^{\frac{1}{m-1}}.$$

PROOF : Since c is convex with respect to f, we have $\forall 0 \leq t \leq 1$

$$c(y, f_Z + t(f_{Z^i} - f_Z), x) - c(y, f_Z, x)t\left(c(y, f_{Z^i}, x) - c(y, f_Z, x)\right).$$

Then, by summing over all couples (x_k, y_k) in Z^i,

$$R_e(f_Z + t(f_{Z^i} - f_Z), Z^i) - R_e(f_Z, Z^i) \quad \leq t\left(R_e(f_{Z^i}, Z^i) - R_e(f_Z, Z^i)\right). \tag{7}$$

By symmetry, (7) holds if Z and Z_i are permuted. By summing this symmetric equation and (7), we obtain

$$R_e(f_Z + t(f_{Z^i} - f_Z), Z^i) - R_e(f_Z, Z^i) + R_e(f_{Z^i} + t(f_Z - f_{Z^i}), Z^i) - R_e(f_{Z^i}, Z^i) \leq 0. \tag{8}$$

Now, by definition of f_Z and f_{Z^i},

$$R_r(f_Z, Z) - R_r(f_Z + t(f_{Z^i} - f_Z), Z) + R_r(f_{Z^i}, Z^i) - R_r(f_{Z^i} + t(f_Z - f_{Z^i}), Z^i) \leq 0. \tag{9}$$

By using (8) and (9) we get

$$c(y_i, f_Z, x_i) - c(y_i, f_Z + t(f_{Z^i} - f_Z), x_i) + \lambda n\left(\|f_Z\|_{\mathcal{H}}^m - \|f_Z + t(f_{Z^i} - f_Z)\|_{\mathcal{H}}^m + \|f_{Z^i}\|_{\mathcal{H}}^m - \|f_{Z^i} + t(f_Z - f_{Z^i})\|_{\mathcal{H}}^m\right) \leq 0, \tag{10}$$

This inequality holds $\forall t \in [0, 1]$. By choosing $t = 1/2$ in (10), we obtain that

$$|c(y_i, f_Z, x_i) - c(y_i, f_Z + \frac{1}{2}(f_{Z^i} - f_Z), x_i)| \geq n\lambda\left(\|f_Z\|_{\mathcal{H}}^m - 2\left\|\frac{f_{Z^i} + f_Z}{2}\right\|_{\mathcal{H}}^m + \|f_{Z^i}\|_{\mathcal{H}}^m\right), \tag{11}$$

Let $u = (f_Z + f_{Z^i})/2$ and $v = (f_Z - f_{Z^i})/2$. Then,

$$\begin{aligned}
\|u+v\|_{\mathcal{H}}^m + \|u-v\|_{\mathcal{H}}^m - 2\|u\|_{\mathcal{H}}^m - 2\|v\|_{\mathcal{H}}^m &= \|f_Z\|_{\mathcal{H}}^m + \|f_{Z^i}\|_{\mathcal{H}}^m - 2\left\|\frac{f_{Z^i}+f_Z}{2}\right\|_{\mathcal{H}}^m - 2\left\|\frac{f_{Z^i}-f_Z}{2}\right\|_{\mathcal{H}}^m \\
&= \left(\|u\|_{\mathcal{H}}^2 + \|v\|_{\mathcal{H}}^2 + 2\langle u, v\rangle_{\mathcal{H}}\right)^{m/2} - 2\left(\|u\|_{\mathcal{H}}^2\right)^{m/2} \\
&\quad + \left(\|u\|_{\mathcal{H}}^2 + \|v\|_{\mathcal{H}}^2 - 2\langle u, v\rangle_{\mathcal{H}}\right)^{m/2} - 2\left(\|v\|_{\mathcal{H}}^2\right)^{m/2} \\
&\geq 2\left(\|u\|_{\mathcal{H}}^2 + \|v\|_{\mathcal{H}}^2\right)^{m/2} - 2\left(\|u\|_{\mathcal{H}}^2\right)^{m/2} - 2\left(\|v\|_{\mathcal{H}}^2\right)^{m/2} \\
&\geq 0,
\end{aligned}$$

where in the last transition we used both Newton's generalized binomial theorem for the first inequality and the fact that $m/2 > 1$ for the second one. Hence, we have shown that

$$\|f_Z\|_{\mathcal{H}}^m - 2\left\|\frac{f_{Z^i}+f_Z}{2}\right\|_{\mathcal{H}}^m + \|f_{Z^i}\|_{\mathcal{H}}^m \geq 2\left\|\frac{f_{Z^i}-f_Z}{2}\right\|_{\mathcal{H}}^m. \tag{12}$$

Now, by combining (11) and (12), we obtain by using Lemma 5.2,

$$\begin{aligned}
\|f_Z - f_{Z^i}\|_{\mathcal{H}}^m &\leq \frac{2^{m-1}}{\lambda n}\left(c(y_i, f_Z + \frac{1}{2}(f_{Z^i} - f_Z), x_i) - c(y_i, f_Z, x_i)\right) \\
&\leq 2^{m-2}\frac{C}{\lambda n}\|f_{Z^i}(x_i) - f_Z(x_i)\|_{\mathcal{Y}} \\
&\leq 2^{m-2}\frac{C\kappa}{\lambda n}\|f_{Z^i} - f_Z\|_{\mathcal{H}},
\end{aligned}$$

which gives that

$$\|f_Z - f_{Z^i}\|_{\mathcal{H}} \leq \left(2^{m-2}\frac{C\kappa}{\lambda n}\right)^{\frac{1}{m-1}}.$$

This implies that, $\forall (x, y)$ a realization of (X, Y),

$$\begin{aligned}
|c(y, f_Z, x) - c(y, f_{Z^i}, x)| &\leq C\|f_Z(x) - f_{Z^i}(x)\|_{\mathcal{Y}} \\
&\leq C\kappa\left(2^{m-2}\frac{C\kappa}{\lambda n}\right)^{\frac{1}{m-1}}.
\end{aligned}$$

For $1 < m < 2$, the problem (1) is well posed but the question whether the algorithm is stable or not in this case remains open. Future studies need to be conducted to further address this issue explicitly. □

6 Experiments

In this section, we conduct experiments on synthetic and real-world datasets to evaluate the efficiency of the proposed algorithm. We use the following real-world datasets extracted from the UCI repository [3] : Concrete Compressive Strength (1030 instances, 9 attributes), Concrete Slump Test (103 instances, 10 attributes), Yacht Hydrodynamics (308 instances, 7 attributes), Wine Quality (4898 instances, 12 attributes), Energy Efficiency (768 instances, 8 attributes), Housing (506 instances, 14 attributes) and Parkinsons Telemonitoring (5875 instances, 26 attributes). Additionally, we also use a synthetic dataset (2000 instances, 10 attributes) described in [TKL05]. In this dataset, inputs $(x_1, ..., x_{10})$ are generated independently and uniformly over $[0, 1]$ and outputs are computed from $y = 10\sin(\pi x_1 x_2) + 20(x_3 - 0.5)^2 + 10x_4 + 5x_5 + \mathcal{N}(0, 1)$. In all our experiments, we use a Gaussian kernel $k_\mu(x, x') = \exp(-\|x - x'\|_2^2/\mu)$ with $\mu = \frac{1}{n^2}\sum_{i,j}\|x_i - x_j\|_2^2$, and the scaled root mean square error (RMSE), defined by $\frac{1}{\max y_i}\sqrt{\frac{1}{n}\sum_i (y_i - f(x_i))^2}$, as evaluation measure.

3. `http://archive.ics.uci.edu/ml/datasets`.

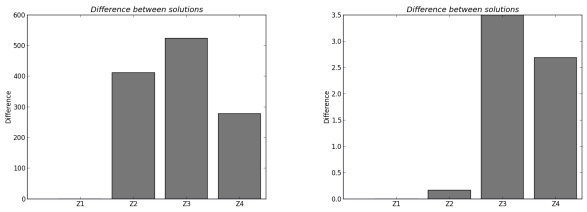

FIGURE 1 – The norm of the difference between the optimal solutions given by M-RLSR and KRR on two datasets randomly split into 4 parts $Z_1, \ldots, Z_4$. While the difference on Z_1 is zero for the two algorithms since they are Z_1-equivalent, they give different solution for Z_1, Z_2 and Z_3. (left) Concrete compressive strength dataset : $m = 1.5, \lambda = 1e-2$ (obtained by 10-fold cross validation) and $\lambda_2 = 5.6e-4$(computed from the Z-equivalence lemma). (right) Synthetic dataset : $m = 1.2, \lambda = 1e-5$ (obtained by 10-fold cross validation) and $\lambda_2 = 6.5e-7$ (computed from the Z-equivalence lemma).

6.1 Z-Equivalence

We start by illustrating experimentally the fact that M-RLSR and KRR algorithms are Z-equivalent but not equivalent. We use here Synthetic and Concrete Compressive Strength datasets. We randomly split these datasets into 4 parts of equal size $Z_1, \ldots, Z_4$. Using Z_1, m is fixed and the regularization parameter λ is chosen by a 10-fold cross-validation for M-RLSR. Then the equivalent λ_2 for KRR is computed using Lemma 4.4. For each part $Z_i, 1 \leq i \leq 4$, we calculate the norm of the difference between the optimal solutions given by M-RLSR and KRR. The results are presented in Figure 1. The difference between the solutions of the two algorithms is equal to 0 on Z_1, since both algorithms are Z_1-equivalent, but is strictly positive on Z_2, Z_3, Z_4, showing that the algorithms are not equivalent.

6.2 Prediction Accuracy

We evaluate the prediction accuracy of the M-RLSR algorithm using the datasets described above and compare it to KRR. For each dataset we proceed as follows : the dataset is split randomly into two parts (70% for training and 30% for testing), we set $\lambda = 1$, and we select m using cross-validation in a grid varying from 0.1 to 2.9 with a step-size of 0.1. The value of m with the least mean RMSE over ten run is selected.Then, with m now fixed, λ is chosen by a ten-fold cross validation in a logarithmic grid of 7 values, ranging from 10^{-5} to 10^2. Likewise, λ_2 for KRR is chosen by 10-fold cross-validation on a larger logarithmic grid of 25 equally spaced values between 10^{-7} and 10^3.

RMSE and standard deviation (STD) results for M-RLSR and KRR are reported in Table 1. It is important to note that the double cross-validation on m and λ for M-RLSR, and the cross-validation on the greater grid for the KRR takes a similar amount of time. Table 1 shows that the m-power RLSR algorithm is capable of achieving a good performance results when $m < 2$. Note that the difference between the performance of the two algorithms M-RLSR and KRR decreases as the grid of λ becomes larger, but in practice we are limited by computational reasons.

6.3 Speed of Convergence

We compare here the convergence speed of M-RLSR with $m \leq 1$ and KRR on Concrete compressive strength, Yacht Hydrodynamics, Housing, and Synthetic datasets. As before, each dataset is randomly split into two parts (70% for learning and 30% for testing). The parameters m and λ are selected using cross-validation : we first fix λ to 1 and choose m over a grid ranging from 0.1 to 1, then λ is set by cross-validation when m is fixed. For KRR, λ_2 is computed from λ and m using Lemma 4.4.

Figure 2 shows the mean of RMSE over ten run for the four datasets with M-RLSR and KRR when varying the number of examples of training data from 10% to 100% with a step size of 5%. In this figure, we can see that

TABLE 1 – Performance (RMSE and STD) of m-power RLSR (M-RLSR) and KRR algorithms on synthetic and UCI datasets. m is chosen by cross-validation on a grid ranging from 0.1 to 2.9 with a step-size of 0.1.

	KRR		M-RLSR		
Dataset	RMSE	STD	m	RMSE	STD
Compressive	8.04e-2	3.00e-3	1.6	7.31e-2	3.67e-3
Slump	3.60e-2	5.62e-3	1.1	3.52e-2	6.49e-3
Yacht Hydro	0.165	1.13e-2	0.1	1.56e-2	7.53e-3
Wine	8.65e-2	6.18e-3	1.3	8.17e-2	6.07e-3
Energy	4.12e-2	1.79e-3	1.1	3.79e-2	2.87e-3
Housing	10.6e-2	7.98e-3	1.3	7.26e-2	9.92e-3
Parkinson	8.05e-2	4.51e-3	0.3	5.56e-2	3.29e-3
Synthetic	3.19e-2	1.56e-3	0.4	1.26e-2	5.85e-4

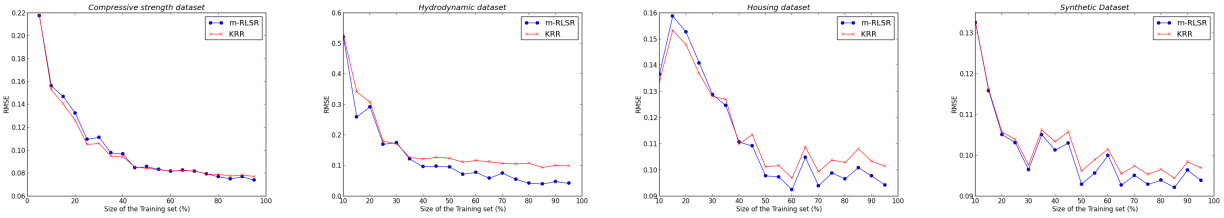

FIGURE 2 – RMSE curve of M-RLSR (blue) and KRR (red) algorithms as a function of the dataset size. (top left) Concrete compressive strength ($m = 0.1$). (top right) Yacht Hydrodynamics ($m = 0.5$). (bottom left) Housing ($m = 0.4$). (bottom right) Synthetic ($m = 0.1$).

M-RLSR with $m < 1$ can improve the speed of convergence of KRR. This confirms the theoretical expectation for this situation [MN10], that is a regularization exponent that grows significantly slower than the standard quadratic growth in the RKHS norm can lead to better convergence behavior.

7 Conclusion

In this paper we proposed m-power regularized least squares regression (RLSR), a supervised regression algorithm based on a regularization raised to the power of m, where m is with a variable real exponent. Our results show that proposed algorithm achieves good accuracy and improves the convergence speed of the standard kernel ridge regression algorithm. This supports previous suggestions that one can use a regularization term that grows significantly slower than the standard quadratic growth in the RKHS norm. In the future, it will be interesting to study the effect of the exponent m on other kernel-based learning algorithms such as SVM. It will also be useful to develop online learning algorithms for m-power regularized problems.

Références

[BE02] O. Bousquet and A. Elisseeff. Stability and generalization. *Journal of Machine Learning Research*, 2 :499–526, 2002.

[Cha07] O. Chapelle. Training a support vector machine in the primal. *Neural Computation*, 19(5) :1155–1178, 2007.

[DS12] F. Dinuzzo and B. Schölkopf. The representer theorem for Hilbert spaces : a necessary and sufficient condition. Advances in Neural Information Processing Systems, 2012.

[KBSZ11] M. Kloft, U. Brefeld, S. Sonnenburg, and A. Zien. ℓ_p-norm multiple kernel learning. *Journal of Machine Learning Research*, 12 :953 :997, 2011.

[MN10] S. Mendelson and J. Neeman. Regularization in kernel learning. *The Annals of Statistics*, 38(1) :526–565, 2010.

[SGV98] C. Saunders, A. Gammerman, and V. Vovk. Ridge regression learning algorithm in dual variables. Advances in Neural Information Processing Systems, 1998.

[SHS+09] I. Steinwart, D. Hush, C. Scovel, et al. Optimal rates for regularized least squares regression. In *COLT Proceedings*, 2009.

[SS02] B. Schölkopf and A. J. Smola. *Learning with kernels*. The MIT Press, 2002.

[STC04] John Shawe-Taylor and Nello Cristanini. *Kernel Methods for Pattern Analysis*. Cambridge University Press, 2004.

[Tik63] A.N. Tikhonov. Regularization of incorrectly posed problems. *Soviet Mathematics Doklady*, 4 :1624–1627, 1963.

[TKL05] I. W. Tsang, J. T. Kwok, and K. T. Lai. Core vector regression for very large regression problems. ICML, 2005.