

# ECON: a Kernel Basis Pursuit Algorithm with Automatic Feature Parameter Tuning, and its Application to Photometric Solids Approximation

Manuel Loth<sup>1,2\*</sup> and Philippe Preux<sup>1,2</sup> and Samuel Delepoulle<sup>3</sup> and Christophe Renaud<sup>3</sup>

<sup>1</sup>Team-Project SequeL, INRIA Lille Nord Europe

<sup>2</sup>LIFL (UMR CNRS), Université de Lille, France

{manuel.loth@inria.fr, philippe.preux}@inria.fr

<sup>3</sup>Laboratoire d'Informatique du Littoral, Université du Littoral Côte d'Opale, Calais, France

{delepoulle, renaud}@lil.univ-littoral.fr

## Abstract

*This paper introduces a new algorithm, namely the Equi-Correlation Network (ECON), to perform supervised classification, and regression. ECON is a kernelized LARS-like algorithm, by which we mean that ECON uses an  $l_1$  regularization to produce sparse estimators, ECON efficiently rides the regularization path to obtain the estimator associated to any regularization constant values, and ECON represents the data by way of features induced by a feature function. The originality of ECON is that it automatically tunes the parameters of the features while riding the regularization path. So, ECON has the unique ability to produce optimally tuned features for each value of the constant of regularization. We illustrate the remarkable experimental performance of ECON on standard benchmark datasets; we also present a novel application of machine learning in the field of computer graphics, namely the approximation of photometric solids.*

## 1 Introduction

We consider supervised learning, classification as well as regression problems: given a set of  $N$  examples  $(\mathbf{x}_i, y_i)_{i \in \{1, \dots, N\}}$ ,  $\mathbf{x}_i \in \mathcal{D} \subset \mathbb{R}^P$ , we wish to predict the label  $y(\mathbf{x})$  of any data  $\mathbf{x} \in \mathcal{D}$  as accurately as possible. If  $y_i \in \mathbb{R}$ , this is a regression problem, whereas if  $y_i \in \mathcal{C}$  with  $\mathcal{C}$  a discrete set of values, this is a supervised classification task.  $y_i$  is a noisy measurement of some function  $y$  at point  $\mathbf{x}_i$ ; the set of examples is the only knowledge we have about this function  $y$  that we try to estimate at any point  $\mathbf{x} \in \mathcal{D}$ .

\*ML acknowledges the support of an INRIA-Nord-Pas de Calais PhD grant during this work.

Each data  $\mathbf{x}$  is supposed to be a real-valued vector, thus  $\mathbf{x} \in \mathbb{R}^P$ . Then, each data  $\mathbf{x}$  may be represented in a feature space as  $(\phi_1(\mathbf{x}), \dots, \phi_M(\mathbf{x}))$ , where each feature function  $\phi_m$  associate a real number to a data ( $\phi_m : \mathcal{D} \rightarrow \mathbb{R}, \forall m \in \{1, \dots, M\}$ ). Each  $\phi_m$  is generally a non-linear combination of the original attributes of the data. Typically, the number of features  $M$  is very big<sup>1</sup>. We denote  $\Phi$  the set of feature functions, that is  $\Phi \equiv \{\phi_1, \dots, \phi_m, \dots, \phi_M\}$ , with  $M \gg P$ . From a geometrical point of view, this can be seen as representing data in an other, generally huge, space, named the "feature space". Our goal is to build an accurate predictor  $\hat{y}$ , linear in the feature space, of the form  $w_0 + \sum_{k=1}^{k=K} w_k \phi_k$ , which is as sparse as possible, that is,  $K$  is as small as possible (where  $\phi_k \in \Phi$ , and  $w_0, w_k \in \mathbb{R}$ ). To this end, we define a loss function  $L(y(\mathbf{x}), \hat{y}(\mathbf{x}))$  which quantifies the cost of mis-predicting the label  $y(\mathbf{x})$  of a data  $\mathbf{x}$ ; in this paper, we use the quadratic loss function  $L(y(\mathbf{x}), \hat{y}(\mathbf{x})) \equiv \|y(\mathbf{x}) - \hat{y}(\mathbf{x})\|^2$ . Accordingly, we measure the error  $E(\hat{y})$  as the average over the whole domain of definition of the data of this loss:  $E(\hat{y}) \equiv \int_{\mathcal{D}} (y(\mathbf{x}) - \hat{y}(\mathbf{x}))^2 d\mathbf{x}$ . We use the empirical error to estimate this loss:  $E_{\text{emp}}(\hat{y}) \equiv \sum_i (y_i - \hat{y}(\mathbf{x}_i))^2$ . To meet the sparsity expectation, we regularize the problem by minimizing the loss function with the  $l_1$ -norm of  $\hat{y}$ , that is  $l_1(\hat{y}) \equiv \sum_{k \geq 1} |w_k|$ . We end-up with the well-known LASSO problem of finding  $y^*$ :

$$y^* \equiv \arg \min_{\hat{y} \in \mathcal{Y}} \sum_i (y_i - \hat{y}(\mathbf{x}_i))^2 + \lambda \sum_k |w_k| \quad (1)$$

where  $\lambda \in \mathbb{R}^+$  is the regularization constant, and  $\mathcal{Y}$  the set of all biased, linear combinations of the elements of  $\Phi$ .

<sup>1</sup>the kernel method informed reader should not think that we assume here the existence of a kernel function, and so on: a feature function is merely a non linear combination of the attributes of the data.

The optimal solution to the LASSO problem depends on  $\lambda$ , and we note  $\hat{y}(\lambda)$  this optimal estimator for a given value of  $\lambda$ . In practice, the value of  $\lambda$  is difficult to set. A common practice to tune it is to try different values, and pick the best. However, efficient algorithms have been proposed to overcome this difficulty by computing all the solutions of the LASSO problem for potentially all values of  $\lambda$  [3, 4]. These algorithms, LARS and kernel basis pursuit, provide a (non-countable) collection of solutions, for each single value of  $\lambda$ , thus  $\hat{y}(\lambda)$ . The set of all couples  $(\lambda, \hat{y}(\lambda))$  is called the “regularization path”.

The features  $\phi_m$  often have parameters themselves; for instance, Gaussian multivariates are often used, for which one has to set the center and the covariance matrix. Setting these parameters is yet another practical problem which has also been studied [14, 18, 5].

The resolution of both problems at once has not yet been much studied. Rosset *et al.* [15] have studied this issue from a fundamental point of view, but the practical issues are very loosely tackled. This is precisely the aim of this paper. Hence, we propose an algorithm that builds the regularization path and tunes automatically the feature parameters at the same time. In the sequel, we will first introduce the necessary material on LARS and kernel basis pursuit algorithms in Sec. 2. Then, we present our algorithm named the “equi-correlated network” (ECON) in Sec. 3. Then, we provide an experimental study of ECON based on standard datasets in Sec. 4: there, we show that we obtain “state-of-the-art” performance on supervised classification, and regression problems, in particular meeting the performance of SVM and boosting. In Sec. 5, we present an original application of this work to the field of computer graphics, a field which is mostly a virgin field of applications for machine learning. Finally, we conclude and discuss future work.

## 2 The LARS and Kernel Basis Pursuit

We wish to solve the LASSO problem (*cf.* eq. (1)). The Least-Angle Regression (LARS) algorithm [3] solves the LASSO in the case where the data are represented by their original attributes, that is, are not shattered into a feature space. The case where the data are shattered into a feature space is dealt by the kernel basis pursuit (KBP) [4]. Clearly, both LARS and KBP may be seen as essentially the same algorithm<sup>2</sup>, and we use the term LARS as a generic term to designate an algorithm to build the regularization path of a LASSO problem.

Let us denote  $\zeta$  the function we wish to minimize in the LASSO:  $\zeta(\hat{y}) = E_{emp}(\hat{y}) + \lambda l_1(\hat{y})$ .

To build the regularization path, the LARS draws from the homotopy idea: we wish to minimize  $E_{emp}(\hat{y})$  which

<sup>2</sup>this being said without wishing to diminish the merits of the authors of the Kernel Basis Pursuit algorithm.

is a non trivial task. To the opposite, we know how to minimize  $l_1(\hat{y})$ : simply take  $K = 0$ ; then,  $l_1(\hat{y}) = 0$ , and  $\hat{y}$  is simply  $\hat{y} = w_0$ . This solution corresponds to  $\lambda = +\infty$ ; so,  $\hat{y}(+\infty) = w_0 = \frac{1}{N} \sum_{i=1}^N y_i$ . Starting from this solution, the LARS looks for the largest value of  $\lambda$  for which  $K$  is 1, that is the value of  $\lambda$  at which it pays to use one of the features to obtain a smaller error than merely predicting  $w_0$  for any data. At this point, one feature enters the expansion  $\hat{y}$ ; we denote this value of  $\lambda$  by  $\lambda_1$ , and for this value, the estimator has the form:  $\hat{y}(\lambda_1) = w_0 + w_{1\lambda_1} \phi_{i_1}$ , with  $\phi_{i_1}$ , a certain feature drawn among the set of  $M$  (potential) features, which is the one that precisely minimizes  $\zeta$ ; please note that for all values  $\lambda > \lambda_1$ ,  $\hat{y}(\lambda) = w_0$ . This  $\phi_{i_1}$  enters the set of “active features”  $\mathcal{A}$ , and leaves the set of potential features  $\mathcal{P}$ . Then, the second iteration of the LARS starts which will provide the value  $\lambda_2 < \lambda_1$  at which a second feature enters the expansion, thus yields:  $\hat{y}(\lambda_2) = w_0 + w_{1\lambda_2} \phi_{i_1} + w_{2\lambda_2} \phi_{i_2}$ . Please, note that  $\forall \lambda_2 < \lambda \leq \lambda_1$ ,  $\hat{y} = w_0 + w(\lambda) \phi_{i_1}$ , where  $w(\lambda) \equiv w_{i_2} + \frac{w_{i_1} - w_{i_2}}{\lambda_1 - \lambda_2} \lambda$ . As demonstrated in the seminal paper on the LARS [3], the search for  $\phi_{i_2}$  may be done greedily; furthermore, once  $\lambda_i$  has been computed for the  $i^{\text{th}}$  iteration, we can compute the next  $\lambda_{i+1} < \lambda_i$ ; at the  $i + 1^{\text{th}}$  iteration, either a potential feature enters the active set and is added to the expansion, or an active feature leaves the expansion and gets back to the set potential features, which means that in this case, the expansion shrinks. Furthermore, between two values of  $\lambda$  corresponding to two subsequent iterations  $i$  and  $i + 1$ , the value of the weights of the active features vary linearly; so, it is only necessary to keep track of the values of these weights at points where a feature becomes active, or when an active feature is inactivated. The fact that the weights of active features are not constant led us to use this cumbersome notation  $w_{i_\lambda}$  to denote the dependence of its value on the value of  $\lambda$ .

A crucial point here is that each time a feature becomes active, the associated weight is not set so as to correct the error as much as this feature could do it, but so that the next feature to enter the expansion, at the next iteration, will be as much correlated with the residual error as all other active features (thus the “equi-correlation” in ECON name)). This is an important property of the LARS, but tricky, and we refer, again, the interested reader to the seminal paper [3].

The LARS may iterate either until  $\lambda$  gets to 0, but this would yield non sparse solutions since at this point, the complexity of the estimator would not be penalized any longer, and all features would be used. In practice, one uses a stopping criterion, such as until the error measured on a test set reaches a minimum.

The computational complexity of the LARS is linear in the number of potential features and quadratic in the number of active features which tends to remain small as will be seen in the experimental section. The LARS is sketched in

---

**Algorithm 1** Sketch of the LARS algorithm.

---

**Require:** A regression problem defined by  $N$  examples  $(\mathbf{x}_i, y_i)$ ,  $\mathbf{x}_i \in \mathcal{D} \subset \mathbb{R}^P$ ,  $y_i \in \mathbb{R}$ .

**Require:** A set of  $M$  potential features  $\Phi$ .

$\lambda \leftarrow +\infty$ .

Set of active units:  $\mathcal{A} \leftarrow \emptyset$ .

Set of potential units:  $\mathcal{P} \leftarrow \Phi \setminus \mathcal{A}$ .

Compute the bias:  $w_0 \leftarrow \frac{1}{N} \sum_i y_i$

Number of active units:  $K \equiv |\mathcal{A}|$ .

Let  $\hat{y} \equiv w_0 + \sum_{k=1}^{K} w_k \phi_k$ .

**while** stopping criterion not fulfilled **do**

  Compute the residual  $\mathbf{r}$  on the training set (i.e.,  $r_i \leftarrow y_i - \hat{y}(\mathbf{x}_i)$ ).

**if** the sign of the weight of an active feature has changed at the previous iteration “update active features” step **then**  
    remove it from  $\mathcal{A}$ , and put it back into  $\mathcal{P}$ , and  $K --$ ;

**else**

**Best feature selection:** select  $\phi_{K+1} \equiv \phi^*$  the feature among  $\mathcal{P}$  which is most correlated with  $\mathbf{r}$ .

**Update the active features:** update the weights of all active features,

**Include the new feature to the active set:** compute its weight  $w_{K+1}$ , add  $\phi_{K+1}$  to  $\mathcal{A}$ , remove it from  $\mathcal{P}$ .

**Udpute:** Update  $\lambda$ , and set  $K ++$ .

**end if**

**end while**

---

Algorithm 1<sup>3</sup>.

### 3 Description of ECON

Since all potential features are tested to enter the expansion at each iteration of the LARS, needless to point out that if the number of potential features is large, each iteration will be costly (even if the complexity is linear in the number of potential units), and in any case, should remain finite. However, the features generally have some hyper-parameters, and, as done in the kernel basis pursuit, one has to select a finite set of hyper-parametrizations *a priori* to apply the LARS. In practice, the choice of this set is not easy. We want to go beyond this limitation, and be able to consider all possible hyper-parametrizations, which typically define a continuous space, for instance of dimension  $P + P^2$  for multivariate Gaussian features. To that end, we propose the equi-correlated network algorithm (ECON) that precisely aims at selecting the best feature to enter the expansion, along with its best hyper-parametrization.

#### 3.1 The principle of ECON

The key difference between ECON and the LARS lies in the function that is minimized in order to select the best

<sup>3</sup>We use the following notations:  $i$  indexes examples,  $k$  indexes active features;  $a \equiv b$  means “ $a$  is defined by the concept  $b$ ”, whereas  $a \leftarrow b$  is the assignment operator of procedural programming language, that is: the value of the variable named  $a$  gets the current value of  $b$ ; in this case, the value of  $a$  does not change whether the value of  $b$  changes later, or not; in the former case ( $\equiv$ ), the value of  $a$  is always the same as the value of  $b$ . Boldfaces, as in  $\mathbf{x}$ , indicates vectors.

feature to enter the  $\hat{y}$  at each iteration, and how the minimization is performed. Otherwise, ECON retains the same principle of iterating over  $\lambda$ , starting from  $+\infty$ , and down to a certain value which corresponds to the stopping criterion used in the algorithm. In the LARS, the set of potential features  $\Phi$  is finite. Let  $\mathbf{r} \in \mathbb{R}^N$  the vector of residual of an estimator  $\hat{y}$ , that is the  $i^{\text{th}}$  component of  $\mathbf{r}$  is  $r_i = \hat{y}(\mathbf{x}_i) - y_i$ . At each iteration, the LARS selects the feature  $\phi^* \in \mathcal{P} \subset \Phi$  which maximizes the correlation between  $\phi$  and  $\mathbf{r}$ . Since  $\Phi$  is finite, this is merely done by enumerating  $\mathcal{P}$ . Now, let us turn to features that have  $p$  hyper-parameters. We denote them as  $\theta \in \Theta \subset \mathbb{R}^p$ , so that features are really  $\phi_\theta$ , where the  $\theta$  is a free parameter which value has to be set. So, now, we really look for the optimal combination  $(\phi^*, \theta^*)$ , and there is thus a non-countable number of potential features to consider.

Technically, at the step which selects  $\phi^*$  in Algorithm 1, the LARS solves the following minimization problem:

$$\min \left\{ \min_{i \in 1..N} \left( \frac{\lambda - \langle \mathbf{x}_i, \mathbf{r} \rangle}{1 - \langle \mathbf{x}_i, \Delta \mathbf{r} \rangle} \right)^+, \min_{i \in 1..N} \left( \frac{\lambda + \langle \mathbf{x}_i, \mathbf{r} \rangle}{1 + \langle \mathbf{x}_i, \Delta \mathbf{r} \rangle} \right)^+ \right\}$$

where  $\langle \cdot, \cdot \rangle$  denotes the inner product,  $\Delta \mathbf{r}$  is the latest correction to the residual, and:

$$(x)^+ \equiv \begin{cases} x & \text{if } x \geq 0 \\ +\infty & \text{if } x < 0 \end{cases}$$

Instead of that, ECON solves the following minimization problem:

$$\xi \equiv \min \left\{ \min_{\theta \in \Theta} \xi_+(\theta), \min_{\theta \in \Theta} \xi_-(\theta) \right\}$$

where:

$$\xi_+(\boldsymbol{\theta}) \equiv \left( \frac{\lambda + \langle \boldsymbol{\phi}_{\boldsymbol{\theta}}, \mathbf{r} \rangle}{1 + \langle \boldsymbol{\phi}_{\boldsymbol{\theta}}, \boldsymbol{\Delta} \mathbf{r} \rangle} \right)^+, \xi_-(\boldsymbol{\theta}) \equiv \left( \frac{\lambda - \langle \boldsymbol{\phi}_{\boldsymbol{\theta}}, \mathbf{r} \rangle}{1 - \langle \boldsymbol{\phi}_{\boldsymbol{\theta}}, \boldsymbol{\Delta} \mathbf{r} \rangle} \right)^+$$

where  $\boldsymbol{\phi}_{\boldsymbol{\theta}} = (\phi_{\boldsymbol{\theta}}(\mathbf{x}_1), \dots, \phi_{\boldsymbol{\theta}}(\mathbf{x}_N))^T$ . If  $\phi$  is continuous and differentiable,  $\xi_+$  and  $\xi_-$  are continuous and differentiable everywhere except at rare unfeasible points, and  $\xi$  is also continuous, and loses differentiability only at the frontiers where  $\arg \min(\xi_+(\boldsymbol{\theta}), \xi_-(\boldsymbol{\theta}))$  changes, that is precisely at each iteration of ECON (which exactly corresponds to where  $\lambda$  is computed in the LARS). Thus, the main task becomes to minimize, at each step, a relatively smooth function of  $\mathbb{R}^p$  (smooth but not convex).

Assuming the features are multivariate Gaussians, we know that ECON is optimal since the results of [15] apply straightforwardly.

### 3.2 Solving the minimization problem

To exemplify things, let us assume without loss of generality, that the features are multivariate Gaussians, that is  $\phi_{\boldsymbol{\theta}=(\boldsymbol{\mu}, \mathbf{C})} \equiv e^{(\mathbf{x}-\boldsymbol{\mu})^T \mathbf{C}(\mathbf{x}-\boldsymbol{\mu})}$  in which  $\boldsymbol{\mu} \in \mathbb{R}^P$ , and the covariance matrix  $\mathbf{C} \in \mathbb{R}^{P \times P}$ . We have to find  $\boldsymbol{\mu}^*$  and  $\mathbf{C}^*$  that minimizes  $\langle \phi_{\boldsymbol{\mu}, \mathbf{C}}, \mathbf{r} \rangle$ . There is no method to solve this minimization problem analytically, so we have to resort to a numerical minimization. Using a numerical minimization instead of an analytical solution of the minimization problem leads to sub-optimal solutions. This seems a strong weakness, so we would like to discuss this point.

We use DiRect [8], a global optimization algorithm which is guaranteed to converge asymptotically towards the global optimum of the function<sup>4</sup>.

Regarding the issue of sub-optimal solutions, two cases may happen: DiRect returns a non optimal point which is close, to the optimum; or, DiRect returns a point which is not even close to the global optimum, but from a local one. In the first case, though we do not do it (yet), post-processing may be done by way of a gradient descent to get to the global optimum. In the second case, this means that ECON makes active a certain feature instead of an other one which is better. Then, the missed feature may be very well found by DiRect in a subsequent iteration of ECON; actually, the odds for this do increase as a feature is missed because the missed optimum will get more and more attractive (in terms of the energy landscape), so that it will quickly be caught; furthermore, the sub-optimal features that have been included instead of the best feature may well be removed in subsequent ECON iterations, since ECON removes active features that are detected as no longer useful.

<sup>4</sup>We use the implementation of DiRect available in the NL-OPT library [7].

## 4 Experimental study

In this section we provide experimental data regarding the performance of ECON. First, we compare ECON with published results of state of the art algorithms on standard supervised classification, and regression datasets. Then, we present an original application to image synthesis.

### 4.1 Methodology

We compare the performance of ECON with published results on support vector machine (SVM) [13], Relevance Vector Machine (RVM) [17], various boosting algorithms investigated by Rätsch [13] and the Fisher Kernel Discriminant (KFD) [11] as reported on the website [12], the kernel basis pursuit [4], and the LOGREG-LASSO algorithm [16]. These published results provide an interesting basis of comparison; SVM are well-known to perform extremely well in supervised classification tasks in terms of accuracy, though not being sparse; RVM is meant to be sparser than SVM; boosting algorithms are known to perform very well too for supervised classification; KFD was reported as also very good for classification. In the LARS family which is meant to provide sparse estimators, kernel basis pursuit and LOGREG-LASSO both perform very well on classification, and regression tasks. This all means that an algorithm that reaches the same kind of performance as the best results available in these publications may be considered as performing very well, as long as it uses the same amount of computational resources.

Basically, we split each dataset into a training set, and a test-set. We use the same amount of data in each set as in the cited publications. We perform 100 such splits on each dataset, and thus perform 100 runs on each dataset. This provide performance on which statistics may be performed.

To cope with the published results, we compare the mean-square error; its standard deviation provides a measure of the variability of the precision of predictions; importantly, the number of terms in the estimator is also measured, to be related to the support vectors of SVM, or the number of weak learners in boosting.

In the performance reported below on supervised classification, and regression tasks, we perform a fixed amount of iterations (500), and then retain the best MSE measured on the test set during these iterations. The figures below are averaged over the 100 training-testing splits. On the photometric solids representation task presented in Sec. 5, the stopping criterion has been investigated further, and is thus discussed in this section.

500 iterations is in general far too much iterations to obtain the best test error. Actually, the test error first decreases along the iterations, and then begins to increase. The iteration at which the test error is minimal is the iteration at

which to stop, and this can effectively be used as the stopping criterion.

Interestingly, while ECON may in principle add a new feature at each iteration, we observe that the number of features increases sub-linearly, and finally saturates after a certain number of iterations. After that, features keep on being removed and added, keeping the same number of features  $K$  in  $\hat{y}$ . This is a very good property of ECON that helps prevent overfitting. The value at which  $K$  saturates varies from a dataset to another; it may remain very small (10 for the Titanic dataset for instance). We think that this value provides some interesting information about the complexity of the dataset; this issue has yet to be investigated more thoroughly.

## 4.2 Supervised classification

ECON is essentially a regression algorithm. To deal with supervised binary classification, we merely encode the class as  $\pm 1$ ; to predict the label of a data, we consider the sign of the prediction for this data:  $\text{sign}(\hat{y}(\mathbf{x}))$ .

Table 1 presents the results: they are striking: on 5 datasets out of 10, ECON obtains the best results in accuracy; ECON improves the best accuracy by 3.5 % on these 5 datasets.

On the other 5 datasets, one (Ringnorm) seems to exhibit an anomaly since there is an order of magnitude between ECON performance and other algorithms; this dataset does have a training set which is much smaller than the test set (15 times smaller), but the situation is the same for the Titanic dataset where ECON performs the best, and Twonorm where the difference in performance is not so large (by far); furthermore, the dimension  $P$  of the data is the same for Ringnorm and Twonorm, so that we conclude that this is also not the reason for this order of magnitude. So, let alone this anomalous Ringnorm, on the other 4 datasets on which ECON is not the best, ECON is only beaten by a small factor; on Thyroid, and Twonorm, the performance of ECON is better than that of SVM, and of the same order as boosting (please, refer to [12] for these figures not reported here).

So despite being an algorithm for regression, and definitely not relying on any principle of margin maximization, ECON obtains “state of the art” performances on supervised classification problems.

Furthermore, ECON provides sparse estimators, as expected. They are much sparser than those provided by SVM, and a bit less sparse than those provided by LOGREG-LASSO, and RVM, but more accurate.

In more details, table 1 provides the following information about classification results on UCI benchmark datasets. For each dataset, we provide the size of the training set  $N_{\text{train}}$  and the size of the test set  $N_{\text{test}}$ , as well as the number of attributes of data  $P$ , and the results of different algo-

rithms. The column entitled “best from [12]” provides the best results available from Rätsch’s data available on his website [12] and discussed in [13, 11]; they compared 7 algorithms: RBF-networks with a fixed number of Gaussian units, Adaboost, and variants ( $\text{LP}_{\text{Reg}}$ -AdaBoost,  $\text{QP}_{\text{Reg}}$ -AdaBoost,  $\text{AdaBoost}_{\text{Reg}}$ ), SVM, and Kernel Fisher Discriminant; we only provide the best results among these (highly performing) algorithms. SVM uses Gaussian kernels. For each algorithm, we provide a pattern of the form  $x(y)/z$  where  $x$  denotes the average MSE,  $y$  its standard deviation, and  $z$  is the mean number of terms in the best  $\hat{y}$ , that is the best value of  $K$  (this figure is provided when it is relevant and available). The best performance is highlighted with boldface font. (There is an annoyance here that the results provided in [12] are not always compatible with those provided in [16]; actually, the results provided here show a better accuracy than the latter reference; so, in short, they are more difficult to beat.)

## 4.3 Regression problems

For regression problems, we illustrate ECON performance on a few standard benchmark datasets in this section. Then, we delve into the real application presented in this paper, the representation of photometric solids, into the next section.

Again, we use the same datasets as Rosset [16], using the same training, and testing sets. Some results are presented in table 2. We compare the performance of ECON with those published in [16], concerning the Support Vector Machine, the Relevance Vector Machine, and the LOGREG-LASSO. For each algorithm, we provide the accuracy measured on a test set. In the table, boldface font highlights the best performance for each dataset.

We note the very good performance of ECON on Friedman’s F1 and F3 functions. On Abalone, though not the best, ECON performs quite well with regards to other algorithms. On Boston housing, it is a little bit behind SVM and RVM, though still very competitive with most other algorithms and published results. Noticeably, the estimators produced by ECON are quite sparse again.

## 5 Application to Photometric Solids representation

In this section, we present and discuss the application of ECON to a real application, originating from computer graphics.

### 5.1 Photometric Solids

In computer graphics, one of the main challenges of the last two decades has been to develop efficient algorithms for

**Table 1. Supervised classification problems. See text for explanations.**

Dataset	$N_{\text{train}}, N_{\text{test}}$	$P$	best from [12]	RVM	LOGREG-LASSO	ECON
Banana	4900, 400	2	LP <sub>Reg</sub> -AB: 10.73(0.43)/18	10.8/11.4	10.7(0.5)/21.8	<b>10.1</b> (1.9)/42.5(10.6)
Breast-cancer	200, 77	9	KFD: 24.77(4.63)	29.9/6.3	26.1(4.6)/17.3	<b>23.3</b> (4.3)/21.3
Diabetes	468, 300	8	KFD: 23.21(1.63)	NA	23.5(1.9)/7.6	<b>22.7</b> (1.8)/28.0
Flare-solar	666, 400	9	SVM: <b>32.43</b> (1.82)	NA	33.3(1.6)/6.4	32.5(1.7)/8.4
German	700, 300	20	SVM: 23.61(2.07)	22.2/12.5	23.63(2.3)/22.3	<b>23.2</b> (2.1)/72.1
Heart	170, 100	13	SVM: <b>15.95</b> (3.26)	NA	16.0(3.1)/10.7	16.2(3.5)/28.3
Ringnorm	400, 7000	20	KFD: 1.49(0.12)	NA	1.8(0.3)/12.3	10.4(3.0)/53.0
Thyroid	140, 75	5	KFD: <b>4.2</b> (2.07)	NA	4.8(2.3)/6.3	4.6(2.3)/21.6
Titanic	150, 2051	3	SVM: 22.42(1.02)	23.0/93.7	22.9(1.2)/5.0	<b>22.0</b> (0.8)/5.5
Twonorm	400, 7000	20	KFD: 2.61(0.15)	NA	<b>2.6</b> (0.2)/8.7	2.9(0.4)/56

**Table 2. Regression problems. See text for explanations.**

DATASET	$N_{\text{TRAIN}}, N_{\text{TEST}}$	$P$	SVM	RVM	LOGREG-LASSO	ECON
			AVE. MSE	AVE. MSE	AVE. MSE	AVE. MSE
FRIEDMAN’S F1	240, 1000	10	2.92/116.2	2.80/59.4	2.84/73.5	<b>2.16</b> /18
FRIEDMAN’S F2	240, 1000	4	4140/110.3	<b>3505</b> /6.9	3808/14.2	4062.46/10
FRIEDMAN’S F3	240, 1000	4	0.0202/106.5	0.0164/11.5	0.0192/16.4	<b>0.0139</b> /24
ABALONE	1000, 2000	8	4.37/972	<b>4.35</b> /12	<b>4.35</b> /19	4.59/63 ± 0.21/22
BOSTON HOUSING	481, 25	13	8.04	<b>7.46</b>	NA	11.51/87

simulating light propagation and light/matter interactions. Accurate approaches have been studied that now allow both computer scientists, and computer artists to generate accurate simulations, or photorealistic, images [9][6].

One key to these simulations lies in an elaborate description of the light sources as they play the central role of any lighting (figure 1 illustrates the effect of different light sources onto the final scene lighting)<sup>5</sup>. Furthermore, the direction of the light emission has to be carefully taken into account when accuracy is required.

Numerous sorts of lighting devices exist: these range from basic incandescent light bulbs, neon tubes, ... to laser beams; we may also consider particles of dust, or droplets that reflect light, as sources of light. Goniophotometers are used to record the light intensity of the device for a large number of emitting directions [1]. This set of directions and intensities is called a *Photometric Solid* (PS). These photometric solids can then be used in place of the accurate description of the entire device (under some assumptions) to obtain the intensity emitted from the device along any direction (see figure 2).

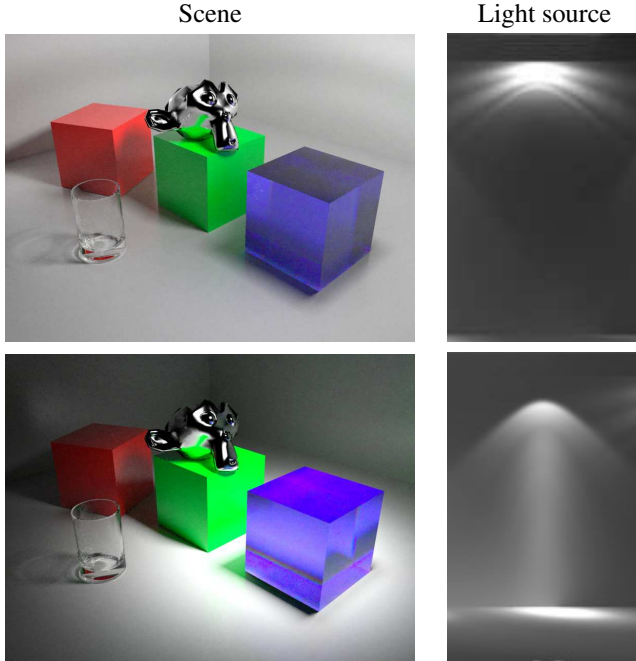
<sup>5</sup>Images have been computed with the Indigo photorealistic renderer (<http://www.indigorenderer.com>).

One of the main drawbacks of PS lies in the large number of sampling directions they require when complex luminaries have to be used: sampling the directions for each degree of angle in spherical coordinates system generates  $180 \times 360 = 64800$  samples that have to be stored and managed during all the simulation process.

## 5.2 Approximating PS with ECON

The current practice is to perform linear interpolation based on measurements. Dealing with several light sources, each being finely sampled, leads to the use of a large amount of memory in any lighting simulation software. We thus studied the use of ECON in order to approximate the PS.

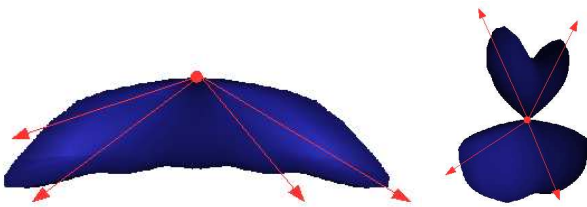
The first step is to provide data to ECON: a subset of the  $(\theta, \phi, L)$  values stored in the PS are provided to the learning algorithm (with  $(\theta, \phi)$  being the lighting direction from the light source expressed in spherical coordinates, and  $L$  the light intensity (luminance) along this direction). Classically 80% of the data are used during the learning stage, the remaining 20% being used for error computation. A quadratic error threshold is used for the ANN, convergence being assumed when the error threshold is less than 5%. In ECON, we investigated this issue a lot because the estimator that minimizes the MSE on a test set is not the most satisfying estimator; the reason is that the data are far from being uniformly distributed in the domain  $(\theta, \phi)$  because of the



**Figure 1. Illustration of the effect of the lighting devices. See text for explanations.**

way the measurements are acquired, and that merely minimizing the MSE leads to overfit the data. We have tested various approaches to this model selection problem, in particular Akaike information criterion (AIC). After a lot of inquiry, we found that a good stopping heuristic is based on the second order difference of the MSE at two subsequent iterations, which is a traditional hint to detect a good model (intuitively, we detect an elbow in the learning curve).

ECON provides an estimator  $\hat{y}$  that is used in the lighting software: when an intensity value along a certain lighting direction is required, this direction is input to the estimator  $\hat{y}$  to approximate the PS in this direction.



**Figure 2. A 3D representation of two Photometric Solids: each one represents the light distribution from the location of the light device that is symbolized by the red point. Some lighting directions are represented with red arrows.**

**Table 3. RMSE for images rendered with ECON, and a neural network approach (a multilayer perceptron). Lighting of the reference image has been computed using a spiral arc interpolation of the PS. Lighting of each image use path tracing algorithm.**

PS dataset	ANN	ECON
2518T1EF	1919.61 (2.93 %)	1340.89 (2.04 %)
8013H1EN	2348.77 (3.58 %)	1176.28 (1.79 %)
8816H1PN	2675.73 (4.08 %)	3164.57 (2.11 %)
LTL14123	3098.26 (4.73 %)	1722.58 (2.63 %)

### 5.3 Some results

The use of ECON solutions has been implemented into a global illumination framework. The results we obtained can be evaluated according to two criterions: on one hand, the absolute error between PS and ECON; on the other hand, we can estimate the quality of the generated images by comparing their content to images generated by more the classical approach. In the end, the second criterion is really what matters here: whether the final image is what we, humans, expect and consider as realistic.

The error should be measured by comparing the values generated by an ECON solution to physical measures. But these measures are generally not available, and only the captured values stored into the PS can be used. In table 3, we report the Root Mean Square Error (RMSE) between a spiral arc interpolation and the approximations provided by a neural network (ANN) [2], and by ECON. In each case, the difference over all pixels is computed to obtain the RMSE. The reference image use arc spiral interpolation. All images are rendered with path-tracing with estimation of the next event [10]. The RMSE is generally small for both ANNs and ECON solutions. ECON consistently provides a better accuracy than ANN<sup>6</sup>.

From a visual perception criterion, Fig. 3 displays two images that have been computed respectively with spiral arc interpolated PS, and with a solution provided by ECON. The 3D scenes that are visible into these images have been voluntarily chosen as simple as possible in order to allow any observer to compare them easily. It appears that no visible difference is visible between the two images, confirming the interest in the solutions obtained by ECON.

<sup>6</sup>The photometric data originate from the Ledalite (<http://www.ledalite.com/>), and Lithonia (<http://www.lithonia.com/>) databases.

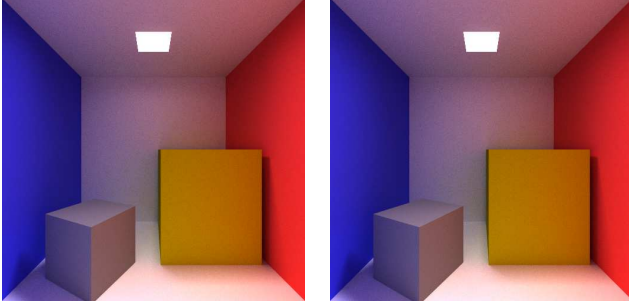


Figure 3. See text for comments.

## 6 Conclusion and future directions of work

In this paper, we have presented the equi-correlated network (ECON) which is a featurized LARS-like algorithm. Its unique ability is that while riding the regularization path, it automatically tunes the parameters of the features used to represent the data. On standard datasets, we have shown that ECON obtains state of the art performance on supervised classification, and regression problems. Then, we have illustrated its use in a novel field of application in computer graphics, aiming at providing a compact representation of photometric solids.

Regarding ECON itself, we have the feeling that its current implementation may be still slightly improved by studying even more carefully the optimizer that tunes the parameters of the features. We have yet observed large improvements by working on it, and we feel that some more refinements are possible.

From a data mining point of view, as we have noted, the number of terms in the estimator built by ECON saturates after a certain amount of iterations. This is likely to convey interesting information about the complexity of the dataset; the location, and shape of the kernels being used by ECON also provide some information that has to be exploited: ECON does not yield a black box estimator.

From the point of view of computer graphics, the use of ECON for approximating photometric solids appears to provide results that are both numerically, and visually convincing. Furthermore, the memory requirements are considerably reduced as compared to the classical use of PS.

### Acknowledgment

Part of the experiments were carried out using the Grid'5000 experimental testbed, an initiative from the French Ministry of Research, INRIA, CNRS and RENATER and other contributing partners.

## References

- [1] I. Ashdown. Near-Field Photometry: Measuring and Modeling Complex 3-D Light Sources. In *ACM SIGGRAPH '95 Course Notes - Realistic Input for Realistic Images*, pages 1–15, 1995.
- [2] S. Delepuille, C. Renaud, and P. Preux. Light source storage and interpolation for global illumination: a neural solution. In *Intelligent Computer Graphics*, Studies in Computational Intelligence, chapter 5, pages 87–104. Springer, 2009.
- [3] B. Efron, T. Hastie, I. Johnstone, and R. Tibshirani. Least-angle regression. *Annals of statistics*, 32(2):407–499, 2004.
- [4] V. Guigue, A. Rakatomamonjy, and S. Canu. Kernel basis pursuit. In *Proc. ECML*, volume 3720, pages 146–157. Springer, LNAI, 2005.
- [5] B. Han, D. Comaniciu, Y. Zhu, and L. Davis. Sequential kernel density approximation and its application to real-time visual tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(7):1186–1197, July 2008.
- [6] H. W. Jensen. *Realistic Image Synthesis Using Photon Mapping*. A. K. Peters LTD, Natick, Massachusetts, 2001.
- [7] S. Johnson. The NLOpt nonlinear-optimization package. <http://ab-initio.mit.edu/nlopt>.
- [8] D. Jones, C. Perttunen, and B. Stuckman. Lipschitzian optimization without the lipschitz constant. *J'nal of Opt. Theory and Applications*, 79(1):157–181, 1993.
- [9] J. Kajiya. The rendering equation. *ACM Computer Graphics*, 20(4):143–150, Août 1986.
- [10] E. Lafortune. *Mathematical Models and Monte Carlo Algorithms for Physically Based Rendering*. PhD thesis, Katholieke Universiteit Leuven, 1996.
- [11] S. Mika, G. Rätsch, J. Weston, B. Schölkopf, and K. Müller. Fisher discriminant analysis with kernels. In Y.-H. Hu, J. Larsen, E. Wilson, and S. Douglas, editors, *Proc. NNSP*, pages 41–48. IEEE, 1999.
- [12] G. Rätsch. Benchmark repository. <http://ida.fraunhofer.de/projects/bench/>.
- [13] G. Rätsch, T. Onoda, and K. Müller. Soft margins for adaboost. *Machine Learning*, 42(3):287–320, 2001.
- [14] V. C. Raykar and R. Duraiswami. Very fast optimal bandwidth selection for univariate kernel density estimation. Technical Report CS-TR-4774, Department of computer science, University of Maryland, College Park, 2005.
- [15] S. Rosset, G. Swirszcz, N. Srebro, and J. Zhu.  $l_1$  regularization in infinite dimensional feature spaces. In *Proc. Computational Learning Theory (COLT)*, volume 4539 of *Lecture Notes in Computer Science (LNCS)*, pages 544–558. Springer, 2007.
- [16] V. Roth. Generalized LASSO. *IEEE Trans. on NN*, 15(1):16–28, 2004.
- [17] M. Tipping. Sparse bayesian learning and the relevance vector machine. *Journal of Machine Learning Research*, 1:211–244, 2001.
- [18] G. Wang, D.-Y. Yeung, and F. Lochovsky. The kernel path in kernelized lasso. In *Proc. of the Eleventh International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 580–587, Mar. 2007.