

# Data-Driven Parameter Estimation

George V. Moustakides

Department of Electrical and Computer Engineering  
University of Patras  
Patras, GREECE  
moustaki@upatras.gr

**Abstract**—Optimum parameter estimation methods require knowledge of a parametric probability density that statistically describes the available observations. In this work we examine Bayesian and non-Bayesian parameter estimation problems under a data-driven formulation where the necessary parametric probability density is replaced by available data. We present various data-driven versions that either result in neural network approximations of the optimum estimators or in well defined optimization problems that can be solved numerically. In particular, for the data-driven equivalent of non-Bayesian estimation we end up with optimization problems similar to the ones encountered for the design of generative networks.

**Index Terms**—Parameter estimation, Neural networks, Data-driven estimation.

## I. INTRODUCTION

THE theory of Detection and Estimation constitutes a major background knowledge in Engineering and Statistics. The corresponding methodologies find application in numerous scientific problems and either provide the actual solution or serve as a starting point for developing techniques that are practically implementable. It is remarkable that with very introductory knowledge of Probability Theory one can derive optimum Detection and Parameter Estimation methods [1], [2]. In parameter estimation, common denominator in all the optimum approaches is the key assumption that we have a complete statistical description in the form of a joint probability density functions of the observations and the parameters to be estimated (Bayesian) or the observations given the parameters to be estimated (non-Bayesian).

Despite the availability of several popular classes of probability densities, these statistical models tend to fail dramatically when they are used to capture the statistical behavior of modern datasets. The reason is that nowadays data are mostly images or videos enjoying a more structured form which cannot be adequately explained by the usual classes of probability density families (e.g. Gaussian). It is therefore clear that it is necessary to develop techniques that do not rely on specific density models.

In most applications there exist sufficient amount of prior data that can be used for *training*, consequently it would be interesting to attempt to develop detection and estimation methods that are *data-driven*, namely do not require exact (or partial) knowledge of probability densities and therefore rely solely on data. Such techniques were developed in [3] for several versions of the binary hypothesis testing problem based on the direct estimation of the *likelihood ratio* of the two

unknown densities which, as we know, is a sufficient statistic for the detection problem. Similar developments for parameter estimation, to our knowledge, do not seem to exist in any systematic way. Of course it is possible to find pure data-driven estimators for specific estimation problems as for example the estimate of a location parameter but there is no systematic method that provides an answer for a general estimation class.

It is this gap we attempt to fill. In particular we offer solutions that are neural network approximations of well known Bayesian estimators for the case where the statistical description of the data is either completely unknown or partially known. But, to our opinion, the most important contribution of this work is the treatment of non-Bayesian parameter estimation. We show that for a rich class of such estimation problems we can obtain solutions using probability density matching techniques. These methods can either be adversarial similar to the ones employed in the design of GANs or based on pure maximization methods. We present simulation results for a very common parameter estimation problem where our idea appears to be antagonistic to the corresponding robust estimator. On the other hand our approach enjoys applicability to all the problems in the class without requiring the development of a specialized method for each separate estimation problem.

## II. DATA-DRIVEN BAYESIAN ESTIMATION

In classical Bayesian parameter estimation we assume that a random vector  $\mathcal{X}$  follows the conditional parametric probability density  $f(X|\theta)$ , where  $\theta$  is the parameter vector that we like to estimate from realizations of  $\mathcal{X}$ . Vector  $\theta$  is also considered a realization of a random vector  $\vartheta$  for which we assume knowledge of a prior density  $p(\theta)$ . Combining the two densities we conclude that  $f(X, \theta) = f(X|\theta)p(\theta)$  is the joint density of the random *pair*  $(\mathcal{X}, \vartheta)$ . Clearly knowing  $f(X, \theta)$  is equivalent to knowing the two densities  $f(X|\theta)$ ,  $p(\theta)$ . We recall that an *estimator* of  $\theta$  is *any* deterministic vector function  $\hat{\theta}(X)$  that has, of course, the same size as  $\theta$ .

In order to produce the optimum estimator, according to classical Bayesian theory [1], [2], we need to select a cost function  $C(U, \theta)$  and define the average cost

$$\mathcal{C}(\hat{\theta}) = E_{\mathcal{X}, \vartheta} [C(\hat{\theta}(\mathcal{X}), \vartheta)] = \iint C(\hat{\theta}(X), \theta) f(X, \theta) dX d\theta, \quad (1)$$

where  $E_{\mathcal{Z}}[\cdot]$  denotes expectation with respect to  $\mathcal{Z}$ . The average cost must be minimized over the vector function  $\hat{\theta}(X)$  in order

to produce the optimum estimator  $\hat{\theta}_o(X)$ , that is,

$$\hat{\theta}_o(X) = \arg \min_{\hat{\theta}(X)} \mathcal{C}(\hat{\theta}) = \arg \min_{\hat{\theta}(X)} \mathbb{E}_{\mathcal{X}, \vartheta} [C(\hat{\theta}(X), \vartheta)]. \quad (2)$$

From [1], [2] we also know that if we define the function

$$\begin{aligned} G(U, X) &= \mathbb{E}_{\vartheta} [C(U, \vartheta) | X] = \int C(U, \theta) f(\theta | X) d\theta \\ &= \frac{\int C(U, \theta) f(X | \theta) p(\theta) d\theta}{\int f(X | \theta) p(\theta) d\theta} = \frac{\mathbb{E}_{\vartheta} [C(U, \vartheta) f(X | \vartheta)]}{\mathbb{E}_{\vartheta} [f(X | \vartheta)]} \end{aligned} \quad (3)$$

where  $f(\theta | X)$  is the *posterior* probability density of  $\vartheta$  given  $X$  then, using (3) it is possible to recover the optimum estimator from the following optimization

$$\hat{\theta}_o(X) = \arg \min_U G(U, X) = \arg \min_U \mathbb{E}_{\vartheta} [C(U, \vartheta) f(X | \vartheta)]. \quad (4)$$

We will use (2) and (4) in order to derive data-driven versions of the optimum estimators. We distinguish the following two cases.

#### A. Unknown $f(X | \theta)$ and $p(\theta)$

This is the simplest and most straightforward version. The knowledge of the two densities or, equivalently, the knowledge of the joint density  $f(X, \theta)$  is replaced by the availability of realizations of the pair  $(X, \vartheta)$  suggesting that we must have a collection of pairs  $\{(X_1, \theta_1), \dots, (X_n, \theta_n)\}$ . We emphasize that we need realizations of the *pair* which is representative of the random relationship that exists between  $X$  and  $\vartheta$  and expressed through the joint density. If instead we only have two unrelated (i.e. independent) sets  $\{X_1, \dots, X_n\}$  and  $\{\theta_1, \dots, \theta_n\}$  then this information is clearly not sufficient to capture the random connection between  $X$  and  $\vartheta$ .

Since our estimator is a vector function  $\hat{\theta}(X)$  that we like to optimize as described above, we can limit our search within a class of vector functions  $u(X, \alpha)$  as for example the class of *neural networks* with  $\alpha$  denoting the *network parameters*. Replacing  $\hat{\theta}(X)$  in (1) with  $u(X, \alpha)$  defines a new average cost that depends only on the network parameters  $\alpha$

$$\tilde{\mathcal{C}}(\alpha) = \mathbb{E}_{\mathcal{X}, \vartheta} [C(u(X, \alpha), \vartheta)]. \quad (5)$$

As in (2), we would like to minimize this criterion over  $\alpha$  in order to optimize  $u(X, \alpha)$ , that is,

$$\alpha_o = \arg \min_{\alpha} \tilde{\mathcal{C}}(\alpha) = \arg \min_{\alpha} \mathbb{E}_{\mathcal{X}, \vartheta} [C(u(X, \alpha), \vartheta)]. \quad (6)$$

The optimization in (6) is in the classical form that accepts computation of  $\alpha_o$  using the stochastic gradient descent algorithm applied to the training data  $\{(X_1, \theta_1), \dots, (X_n, \theta_n)\}$ . Specifically we have

$$\begin{aligned} \alpha_t &= \alpha_{t-1} - \mu \nabla_{\alpha} C(u(X_t, \alpha_{t-1}), \theta_t) \\ &= \alpha_{t-1} - \mu [\mathbb{J}_{\alpha} u(X_t, \alpha_{t-1})]^T [\nabla_U C(u(X_t, \alpha_{t-1}), \theta_t)], \end{aligned} \quad (7)$$

where  $\mathbb{J}_{\alpha} u(X, \alpha)$  denotes the Jacobian of  $u(X, \alpha)$  with respect to  $\alpha$  and  $\nabla_U C(U, \theta)$  the gradient of  $C(U, \theta)$  with respect to  $U$ . We recall that  $\mu > 0$  is the step size (learning rate) and that in each iteration  $t$  we employ a pair  $(X_t, \theta_t)$  from the available training data. If the data are exhausted before we reach

convergence then we can reuse them after, possibly, applying a random permutation. Alternatively, we could approximate expectation with sample means and replace (6) with

$$\alpha_o = \arg \min_{\alpha} \frac{1}{n} \sum_{i=1}^n C(u(X_i, \alpha), \theta_i),$$

that accepts a gradient descent iterative solution of the form

$$\alpha_t = \alpha_{t-1} - \frac{\mu}{n} \sum_{i=1}^n [\mathbb{J}_{\alpha} u(X_i, \alpha_{t-1})]^T [\nabla_U C(u(X_i, \alpha_{t-1}), \theta_i)].$$

If the algorithm converges to  $\alpha_o$  and if this limit does not correspond to some local minimum, then we expect that we can approximate the optimum estimator as follows

$$\hat{\theta}_o(X) \approx u(X, \alpha_o). \quad (8)$$

In other words, we anticipate that the output of  $u(X, \alpha_o)$  will provide estimates that are close to the estimates of the optimum estimator  $\hat{\theta}_o(X)$ .

From the above it is clear that in this case we compute a mathematical formula for the estimator in the form, for example, of a neural network. We must however point out that the application of iterative solvers based on gradients is possible only if we can find the gradient of  $C(U, \theta)$  with respect to  $U$ . This is clearly the case in the MMSE criterion where  $C(U, \theta) = \|U - \theta\|^2$  or the MAE criterion with  $C(U, \theta) = \|U - \theta\|_1$ . Unfortunately the same observation does not apply in the case of the popular MAP estimator where  $C(U, \theta) = \mathbb{1}_{\{\|U - \theta\| > \delta\}}$  with  $0 < \delta \ll 1$  and  $\mathbb{1}_A$  denoting the indicator function of the set  $A$ . This is because the indicator takes values 1 or 0 with derivative that is either 0 at non-boundary points or  $\infty$  at boundary points. Of course it is always possible to approximate  $\mathbb{1}_{\{\|U - \theta\| > \delta\}}$  with some smooth and differentiable functions but we will still experience computational problems because  $\delta$  must be selected very small suggesting that in (7) we will rarely observe gradients that are not close to 0. This will clearly affect the convergence speed of the iterations producing excessive convergence delays. Consequently the MAP estimator requires a substantially different approach for which, unfortunately, we have no meaningful answer at the moment.

#### B. Known $f(X | \theta)$ and unknown $p(\theta)$

Although we argued that we are interested in abandoning the assumption of known probability densities, we would like to consider the case where  $f(X | \theta)$  is known. The reason is that in some applications this assumption is regarded as realistic. The prior  $p(\theta)$  of the random parameter  $\vartheta$  on the other hand, as in the previous case, is considered unknown and replaced by the availability of a set of realizations  $\{\theta_1, \dots, \theta_n\}$  that follow  $p(\theta)$ . The goal is to employ directly this dataset to obtain Bayesian-like estimates of the desired parameters instead of using it to estimate the prior density  $p(\theta)$  first and then apply the classical Bayesian theory.

Here we are not necessarily targeting the development of a mathematical expression for the estimator since we no longer

have realizations of  $\mathcal{X}$  that could be used for training. We recall that we have assumed that we know the functional form of the conditional density  $f(X|\theta)$ . Suppose now that we are given a realization  $X$  of  $\mathcal{X}$  for which we would like to obtain the corresponding estimate of  $\theta$ . Following (4) we have that

$$\hat{\theta}_o(X) = \arg \min_U \mathbb{E}_\vartheta [C(U, \vartheta) f(X|\vartheta)], \quad (9)$$

which involves averaging *only* with respect to  $\vartheta$  while  $f(X|\theta)$  for given  $X$  is a known function of  $\theta$ . The optimization in (9) is in the standard form that accepts a stochastic gradient descent algorithmic solution of the form

$$U_t = U_{t-1} - \mu [\nabla_U C(U_{t-1}, \theta_t)] f(X|\theta_t). \quad (10)$$

The limit of the sequence  $\{U_t\}$  generated by the iterative procedure in (10) will constitute the desired estimate  $\hat{\theta}_o(X)$ . Alternatively, one may approximate the expectation in (9) with the sample mean and attempt the minimization

$$\hat{\theta}_o(X) = \arg \min_U \frac{1}{n} \sum_{i=1}^n C(U, \theta_i) f(X|\theta_i) \quad (11)$$

which whenever not possible to solve analytically it can give rise to a gradient descent algorithm of the form

$$U_t = U_{t-1} - \frac{\mu}{n} \sum_{i=1}^n [\nabla_U C(U_{t-1}, \theta_i)] f(X|\theta_i), \quad (12)$$

every time the data vector  $X$  is given. We observe that both iterations (10) and (12), for every *given* observation vector  $X$ , employ the dataset  $\{\theta_1, \dots, \theta_n\}$  and the knowledge of the density  $f(X|\theta)$  in order to compute *numerically* the desired estimate.

There are cases where it is possible to solve (11) directly and obtain a mathematical formula for the corresponding estimator. When  $C(U, \theta) = \|U - \theta\|^2$ , that is, when we are interested in the MMSE, it is easy to verify that this leads to the following estimator function

$$\hat{\theta}_{\text{MMSE}}(X) = \frac{\sum_{i=1}^n \theta_i f(X|\theta_i)}{\sum_{i=1}^n f(X|\theta_i)}, \quad (13)$$

with the right hand side being an approximation of the conditional expectation of  $\theta$  given  $X$  which is the ideal MMSE estimator. The resulting formula is clearly *not in the form of a neural network*. A closed form solution is also possible in the case of the minimum mean absolute error (MMAE) however, due to lack of space we are not going to present it.

Completing our presentation of the Bayesian-like data-driven estimators we must add that, as in Section II-A, iterative (stochastic) gradient descent algorithms are impossible to apply in the case of the MAP estimator because, as before, we cannot compute the gradient of  $C(U, \theta)$  with respect to  $U$ .

### III. DATA-DRIVEN NON-BAYESIAN ESTIMATION

Let us now examine the far more interesting problem of non-Bayesian parameter estimation. In its classical version we are given a density  $f(X|\theta)$  that contains a parameter vector  $\theta$  which is considered *deterministic and unknown*. In other

words there exist no prior density that describes its statistical behavior. The most popular means to solve this parameter estimation problem [1], [2] is by employing the Maximum Likelihood Estimator (MLE), namely

$$\hat{\theta}(X) = \arg \max_{\theta} f(X|\theta). \quad (14)$$

This estimator, under general conditions enjoys asymptotic optimality (as the length of  $X$  tends to infinity) in the sense that in the limit its error covariance matrix *approaches the Cramer-Rao Lower Bound* (CRLB).

Defining a data-driven version for this estimation problem is not as straightforward as in the Bayesian case. First of all because there is no prior for  $\theta$  this immediately translates in the data-driven setup that *there are no realizations* of  $\theta$  which could be used for training. Since the idea is to replace probability densities with data sampled from these densities we assume that we have available a set of data  $\{X_1, \dots, X_n\}$  that follow the unknown density  $f(X|\theta)$  *for the same*  $\theta$ . If we try to solve our problem at this stage, it is possible to employ different data-driven formulations with drastically different answers but without any means to decide which is the most appropriate solution. To be able to proceed we need to impose additional structure on the problem of interest that will allow us to produce a version which makes sense from a practical as well as theoretical point of view.

A possible direction we may follow is to define the parametric density  $f(X|\theta)$  *indirectly*. Let us start with a random vector  $\mathcal{Z}$  which is distributed according to the density  $g(\mathcal{Z})$ . Consider now a *deterministic transformation*  $T(\mathcal{Z}, \theta)$  that contains the parameter vector  $\theta$ . By defining with the help of the transformation the new random vector  $\mathcal{X} = T(\mathcal{Z}, \theta)$  it is clear that  $\mathcal{X}$  will have a probability density  $f(X|\theta)$  which is a function of  $\theta$ . We note that the parametric density  $f(X|\theta)$  of course exists but we do not necessarily have its explicit form. We must also point out that the transformation does not have to be one-to-one since the resulting  $\mathcal{X}$  can be of dimension larger than the dimension of  $\mathcal{Z}$  allowing  $\mathcal{X}$  to live on a lower dimensional manifold. Consider now the following parameter estimation problem.

**Non-Bayesian Parameter Estimation Problem:** Assume a random vector  $\mathcal{Z}$  is distributed according to the density  $g(\mathcal{Z})$ . A transformation  $T(\mathcal{Z}, \theta)$  with parameters  $\theta$  is applied onto  $\mathcal{Z}$  generating the random vector  $\mathcal{X} = T(\mathcal{Z}, \theta)$  which follows the unknown density  $f(X|\theta)$ . We are given the dataset  $\{X_1, \dots, X_n\}$  comprised of independent realizations of  $\mathcal{X}$  generated with the same  $\theta$  and the dataset  $\{Z_1, \dots, Z_m\}$  with independent realizations of  $\mathcal{Z}$  that follow  $g(\mathcal{Z})$ . Assuming knowledge of the functional form of the transformation  $T(\mathcal{Z}, \theta)$  we would like to estimate the parameter vector  $\theta$  that gives rise to the first dataset.

This is clearly a parameter estimation problem which is purely data-driven since there is no knowledge of any probability density. One might argue that we do not need any densities since from the correspondence  $X_i = T(Z_i, \theta)$  it is a simple exercise to estimate  $\theta$  by minimizing some form of distance

between the two sides. However, *this is not true* because the two datasets are considered entirely unrelated being sampled independently with no actual correspondence between their samples. The first dataset is simply a representative of  $f(X|\theta)$  (containing the information about  $\theta$ ) while the second is a representative of  $g(Z)$ .

With this class of parametric densities generated with the help of parametrized transformations we cannot, of course, capture the generality of the original parameter estimation problem where  $f(X|\theta)$  can be any parametric density. However the proposed class is fairly rich with some classical parameter estimation problems being straightforward examples of the proposed data model. For instance if  $T(Z, \theta) = Z + \theta$  then this corresponds to an unknown translation of the random vector  $Z$ . In fact this simple transformation will be used in our simulation experiments in Section IV. Another well known instance of our setting is a change of scale in each element of  $Z$  where  $T(Z, \theta) = \theta \odot Z$  with “ $\odot$ ” denoting the element-by-element multiplication of the two vectors. Finally, a more challenging version would be  $T(Z, \Theta) = \Theta Z$  where  $\Theta$  is an unknown matrix that replaces the parameter vector  $\theta$ . Clearly, one can come up with more complex examples that are not necessarily linear as the cases we mentioned.

Possible solution to the non-Bayesian parameter estimation problem constitutes the moment matching method proposed in [4] where moments of  $\mathcal{X}$  estimated from the first dataset  $\{X_1, \dots, X_n\}$  are matched to the corresponding moments of  $T(Z, \theta)$  using the transformed second dataset  $\{T(Z_1, \theta), \dots, T(Z_m, \theta)\}$  thus defining suitable equations. By employing an adequate number of such equations we can solve for the unknown parameters  $\theta$ . Unfortunately there is an infinite number of moment combinations that could be used to solve the same problem and, more importantly, we recall that classical moment estimation methods are notoriously non-robust hence easily resulting in unsatisfactory performance.

#### A. Density Matching

Instead of attempting to match moments we could alternatively select the parameters *to match the two probability densities* of the two datasets  $\{X_1, \dots, X_n\}$  and  $\{T(Z_1, \theta), \dots, T(Z_m, \theta)\}$ . For density matching it is not necessary to estimate the two densities. For example, it would be sufficient to estimate the *likelihood ratio function* and by properly selecting the parameters  $\theta$  to bring this function as close as possible to 1 (perfect match).

The idea we just mentioned is motivated by the results in [5] where the notion of Generative Adversarial Networks (GANs) was first introduced. We recall from [5] that we have a random vector  $\mathcal{X}$  that follows a density  $f(X)$  and we are interested in generating realization of  $\mathcal{X}$ . This is achieved by first generating realizations of  $Z$  which follows some density  $g(Z)$  and then applying a transformation  $\mathcal{Y} = G(Z)$  with  $G(Z)$  known as the “generator” function. The generator  $G(Z)$  is designed so that *the density of  $\mathcal{Y}$  matches the density of  $\mathcal{X}$* . The “matching quality” is evaluated with the help of the “discriminator” function  $D(X)$  that tries to differentiate between the “true”

$\mathcal{X}$  and the “synthetic”  $G(Z)$ . Matching is achieved when the selected generator makes the discriminator fail in its task to distinguish the statistical behavior of the two random vectors  $\mathcal{X}$  and  $\mathcal{Y}$ . In [5] it is proved that the generator/discriminator pair which is capable of achieving the desired matching is the solution of the following min-max (adversarial) problem

$$\min_{D(X)} \max_{G(Z)} \{E_{\mathcal{X}}[\log D(\mathcal{X})] + E_Z[\log(1 - D(G(Z)))]\}. \quad (15)$$

This first original work was followed by a number of alternative methods that appeared in the literature all adopting a similar adversarial setup. We must mention [6] but also the more general result in [7], [8]. Regarding the latter approach, the problem in (15) is extended to

$$\min_{G(Z)} \max_{D(X)} \{E_{\mathcal{X}}[\phi(D(\mathcal{X}))] + E_Z[\psi(D(G(Z)))]\}, \quad (16)$$

with the two functions  $\phi(z), \psi(z)$  satisfying  $\psi'(z) = \rho(z)$ ,  $\phi'(z) = -\omega^{-1}(z)\rho(z)$ , where “ $'$ ” denotes derivative,  $\rho(z) > 0$  is a strictly positive function and  $\omega(r)$  is a strictly increasing differentiable function defined on  $r \in [0, \infty)$  with  $\omega^{-1}(z)$  being its inverse function. In [7], [8] it is then proved that the adversarial problem in (16) produces a generator/discriminator pair with the generator output  $\mathcal{Y} = G(Z)$  matching the *statistical behavior* of  $\mathcal{X}$  (i.e. its density). In [7] one can find a plethora of pairs following the above rules which are successful in identifying the right generator function.

Adversarial approaches when applied to the design of GANs are well known to suffer from convergence instability when implemented iteratively using stochastic gradients. Of course we must also not forget the fact that we are interested in designing a generator and we end up designing also a second function, the discriminator, which becomes useless once the generator is computed. One can find very few generator design techniques that do not need a discriminator function. These methods are *non-adversarial* suggesting that their implementation is going to be far more stable than their adversarial counterparts. We focus on a specific such technique introduced in [9] which is called *Maximal Correlation* method and consists in solving the following optimization problem with respect to the generator  $G(Z)$

$$\max_{G(Z)} \frac{(E_{\mathcal{X}, Z}[K(\mathcal{X}, G(Z))])^2}{E_{Z^1, Z^2}[K(G(Z^1), G(Z^2))]} \quad (17)$$

where  $K(X, Y)$  is a positive definite kernel,  $Z, Z^1, Z^2$  are random vectors with the same density  $g(Z)$  with  $Z^1, Z^2$  being independent, while  $\mathcal{X}$  follows  $f(X)$ . As it is shown in [9] the generator function that maximizes the correlation in (17) when used in the transformation  $\mathcal{Y} = G(Z)$  produces a  $\mathcal{Y}$  matching  $\mathcal{X}$  in density as in the adversarial problems. Since here we have a single optimization its implementation is far easier.

The connection of the two approaches in (15) and (17) to our parameter estimation problem is not difficult to see. In our case we do *not need to identify the generator function*  $G(Z)$  since this role is undertaken by the parametric transformation  $T(Z, \theta)$ . Focusing on (17) which we are going to adopt in our simulation experiments the optimization problem becomes

$$\max_{\theta} \frac{(\mathbb{E}_{\mathcal{X}, \mathcal{Z}} [\mathbb{K}(\mathcal{X}, \mathcal{T}(\mathcal{Z}, \theta))])^2}{\mathbb{E}_{\mathcal{Z}^1, \mathcal{Z}^2} [\mathbb{K}(\mathcal{T}(\mathcal{Z}^1, \theta), \mathcal{T}(\mathcal{Z}^2, \theta))]},$$

with the optimization over  $G(Z)$  being replaced by the optimization over the parameters  $\theta$ . By approximating expectations with sample means gives rise to the data-driven version of the optimization problem that will provide our desired parameter estimates. Specifically we are interested in

$$\max_{\theta} \frac{(\mathcal{N}(\theta))^2}{\mathcal{D}(\theta)} \quad \text{where} \quad (18)$$

$$\mathcal{N}(\theta) = \sum_{i=1}^n \sum_{j=1}^m \mathbb{K}(X_i, \mathcal{T}(Z_j, \theta)) \quad (19)$$

$$\mathcal{D}(\theta) = \sum_{i=1}^m \sum_{j=1, j \neq i}^m \mathbb{K}(\mathcal{T}(Z_i, \theta), \mathcal{T}(Z_j, \theta)),$$

and where in the denominator we avoid combining the same realizations for  $\mathcal{Z}^1$  and  $\mathcal{Z}^2$  in order to assure independence.

#### IV. EXPERIMENTS

We focus on the non-Bayesian version and in particular the first problem mentioned in Section III namely the estimation of an unknown translation of the data. For simplicity we limit ourselves to the scalar case. Let us begin with a density  $g_0(W)$  which is zero mean. We then define  $g(Z) = g_0(Z - \mu)$  where  $\mu$  is an initial unknown mean of  $Z$ . Next we apply a translation  $\theta$  that results in  $f(X|\theta) = g(X - \theta) = g_0(X - \mu - \theta)$ . The goal is to estimate  $\theta$  using the two datasets  $\{X_1, \dots, X_n\}$  and  $\{Z_1, \dots, Z_m\}$  sampled from  $f(X|\theta)$  and  $g(Z)$  respectively. This suggests that we could first estimate  $\mu$  using  $\{Z_1, \dots, Z_m\}$  and then  $\mu + \theta$  from  $\{X_1, \dots, X_n\}$ . The two estimates must then be subtracted  $\hat{\theta} = \mu + \theta - \hat{\mu}$  to produce the desired estimate of  $\theta$ . Since each estimate employs a different dataset and since the two datasets are independent we can write

$$\mathbb{E}[(\hat{\theta} - \theta)^2] = \mathbb{E}[(\widehat{\mu + \theta} - \mu - \theta)^2] + \mathbb{E}[(\hat{\mu} - \mu)^2] \geq \frac{1}{\text{FI}} \left( \frac{1}{n} + \frac{1}{m} \right),$$

where we have lower bounded each error power by its corresponding CRLB with the Fisher Information satisfying  $\text{FI} = \int \frac{(\dot{g}_0(W))^2}{g_0(W)} dW$ . No estimator of  $\theta$  in the form  $\hat{\theta} = \widehat{\mu + \theta} - \hat{\mu}$  can enjoy an error power smaller than the lower bound we have specified. We know [1], [2] that this lower bound is attained asymptotically for large  $n, m$  by the MLE with the corresponding estimate being

$$\hat{\theta}_{\text{MLE}} = \arg \max_{\nu} \sum_{i=1}^n \log g_0(X_i - \nu) - \arg \max_{\mu} \sum_{i=1}^m \log g_0(Z_i - \mu),$$

where  $\nu$  replaces the sum  $\mu + \theta$ . Of course the previous estimate is not data-driven since it requires knowledge of the density  $g_0(W)$ .

The most obvious data-driven estimator of  $\theta$  is clearly the one that matches the first moments by combining the two sample means, that is

$$\hat{\theta}_{\text{M}} = \frac{1}{n} \sum_{i=1}^n X_i - \frac{1}{m} \sum_{i=1}^m Z_i. \quad (20)$$

As before, because of independence of the two datasets we can easily show that

$$\mathbb{E}[(\hat{\theta}_{\text{M}} - \theta)^2] = \sigma_0^2 \left( \frac{1}{n} + \frac{1}{m} \right),$$

where  $\sigma_0^2 = \int W^2 g_0(W) dW$  is the corresponding variance. Since sample means are well known to be non-robust one can develop robust alternatives by employing an  $M$ -estimator as in [10]

$$\hat{\theta}_{\text{R}} = \arg \min_{\nu} \sum_{i=1}^n \varphi(X_i - \nu) - \arg \min_{\mu} \sum_{i=1}^m \varphi(Z_i - \mu),$$

where  $\varphi(W)$  is a proper convex function. For example  $\varphi(W) = \frac{1}{2}W^2$  results in the estimator in (20), while selecting  $\varphi(W)$  to be the Huber function [10]

$$\varphi(W) = \begin{cases} \frac{1}{2}W^2 & \text{for } |W| \leq c \\ c|W| - \frac{c^2}{2} & \text{for } |W| > c, \end{cases}$$

constitutes a popular method to robustify the estimates of the two means (location parameters) in (20). Parameter  $c$  is selected as  $c = 1.9941 \times \text{median}(|X_i - \text{median}(X_j)|)$  a value which delivers a 95% efficiency in the Gaussian case. Finally, for the proposed maximal correlation method in (18),(19) we consider the Gaussian kernel  $\mathbb{K}(X, Y) = e^{-\frac{\|X-Y\|^2}{h}}$  with  $h = 2(\text{median}(|X_i - \text{median}(X_j)|))^2$  which is an estimate of the square of the scale of the data.

Regarding the sizes of the two datasets, we examine the case  $n = m = 100$ . For the density  $g_0(W)$ , we simulate three cases: 1) Gaussian with  $g_0(W) = (2\pi)^{-1/2} e^{-W^2/2}$ ,  $\text{FI}=1$ ,  $\sigma_0^2 = 1$ ; 2) Laplace with  $g_0(W) = 0.5e^{-|W|}$ ,  $\text{FI}=1$ ,  $\sigma_0^2 = 2$  and 3) Cauchy with  $g_0(W) = \frac{1}{\pi} \frac{1}{1+W^2}$ ,  $\text{FI}=0.5$ ,  $\sigma_0^2 = \infty$ , that exhibit increasing tail fatness. In all three examples we select  $\theta = \mu = 1$ . For the MLE, the moment matching, the Huber robust estimator and the maximal correlation (proposed), the error power is computed by averaging over 100,000 independent runs.

TABLE I  
ERROR POWER OF TRANSLATION ESTIMATES.

	Gaussian	Laplace	Cauchy	
CRLB	0.020	0.020	0.040	
MLE	0.020	0.023	0.041	
Moment Matching	0.020	0.040	$\infty$	Data-driven
Huber Estimator	0.021	0.029	0.073	Data-driven
Maximal Correlation	0.027	0.026	0.043	Data-driven

In Table I we present the error power for each estimator. In the case of Gaussian data we know that the simple moment matching estimator in (20) is the same as the MLE and attains the CRLB for every *finite*  $n, m$ . However performance degrades rapidly as we diverge from Gaussianity and use data from fat-tailed densities. The main observation is that our method is antagonistic to Huber's robust estimator being also very close to the MLE which is not data-driven. At the same time our idea enjoys the advantage of being applicable to *any* transformation  $\mathcal{T}(Z, \theta)$  as opposed to Huber's robust approach which is primarily employed for the location parameter problem.

## REFERENCES

- [1] H.V. Poor, *An Introduction to Signal Detection and Estimation*, 2nd Ed, Springer, 1994.
- [2] P. Moulin, V.V. Veeravalli, *Statistical Inference for Engineers and Data Scientists*, Cambridge, NY, 2019.
- [3] G.V. Moustakides, K. Basioti, "Training neural networks for likelihood/density ratio estimation," *arXiv: 1911.00405*, Nov. 2019.
- [4] C.-L. Li et al., "MMD GAN: Towards deeper understanding of moment matching network," *arXiv: 1705.08584*, 2017.
- [5] J. Goodfellow et al., "Generative adversarial networks," *arXiv: 1406.2661*, 2014.
- [6] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein generative adversarial networks," *International Conference on Machine Learning*, PMLR, pp. 214–223, 2017.
- [7] K. Basioti, G.V. Moustakides, "Designing GANs: A likelihood ratio approach," *arXiv: 2002.00865*, Feb. 2020.
- [8] K. Basioti, G.V. Moustakides, "Generative adversarial networks: A likelihood ratio approach," *International Joint Conference on Neural Networks*, Shenzhen, China, July 2021.
- [9] K. Basioti, G.V. Moustakides, E.Z. Psarakis, "Maximal correlation: An alternative criterion for training generative networks," *24th European Conference on Artificial Intelligence*, Santiago de Compostela, Spain, June 2020.
- [10] P.J. Huber, "Robust estimation of a location parameter," *Ann. Math. Statist.* vol. 35, no. 1, pp. 73–101, 1964.