Utility metrics for assessment and subset selection of input variables for linear estimation

Alexander Bertrand, *Senior Member, IEEE*

***Abstract*—This tutorial paper introduces the utility metric and its generalizations, which allow for a 'quick-and-dirty' quantitative assessment of the relative importance of the different input variables in a linear estimation model. In particular, we show how these metrics can be cheaply calculated, thereby making them very attractive for model interpretation, online signal quality assessment, or greedy variable selection. The main goal of this paper is to provide a transparent and consistent framework that consolidates, unifies, and extends the existing results in this area. In particular, we (a) introduce the basic utility metric and show how it can be calculated at virtually no cost, (b) generalize it towards group-utility and noise impact metrics, and (c) further extend it to cope with linearly dependent inputs and minimum norm requirements.**

***Index Terms*—Utility, least squares, model interpretation, variable selection, quantization, ridge regression**

I. Introduction

When solving a regression problem, one often wants to have some quantitative insights into the relevance of each input variable, i.e., how much it contributes to the reduction of a loss function. Such information can be used to interpret the model, to assess the predictive value of specific input variables or signals, or to perform a greedy variable subset selection [1]–[4]. The latter allows to reduce the dimensionality of a model, e.g., to avoid overfitting [5], to make the model more interpretable, or to reduce computational complexity, data storage, data transmission, or sensor costs [3], [6], [7].

For example, consider the example of linear least squares (LS) regression, which will also be the focus of this tutorial[1]. A naive heuristic that is remarkably commonly used for variable assessment is the magnitude of the weights of the LS solution, thereby (incorrectly) assuming that important input variables will also receive a large weight in the LS solution. However, it is not difficult to see that this reasoning is flawed. For example, if the observations of one of the input variables would all be scaled with a factor α, then the corresponding weight in the LS solution will be scaled with α^{-1}, whereas the information content of that input variable obviously remains unchanged.

A more relevant metric would consist of calculating the effective loss, i.e., the increase in LS cost, if an input variable would be removed and if the model would be re-optimized. We refer to this resulting metric as the *utility* of that input variable. Utility is a powerful heuristic for input variable assessment [1], [3], [4], [6], and can even be shown to have some optimality properties when used for greedy variable subset selection [1], which can compete with well-known sparse regression techniques such as the least absolute shrinkage and selection operator (LASSO) [9].

However, computing the utility of M input variables by definition requires to solve M different LS problems, i.e., one for each removal of an input variable [1], [3]. As a result, the metric scales poorly with the dimensionality of the model, which can be problematic in real-time applications, and which can make a greedy variable subset selection in very high-dimensional problems even infeasible.

In this tutorial paper, we show how some simple tricks from standard linear algebra allow to compute the utility metric at virtually no cost, thereby making

The author acknowledges the financial support of the KU Leuven Research Council for project C14/16/057, FWO (Research Foundation Flanders) for projects G.0031.14, 1.5.123.16N, G.0D75.16N, and G.0A49.18N, and the European Unions Horizon 2020 research and innovation programme under grant agreement No 766456 (project AMPHORA).

The author is with KU Leuven, Department of Electrical Engineering (ESAT), Stadius Center for Dynamical Systems, Signal Processing and Data Analytics, Kasteelpark Arenberg 10, box 2446, 3001 Leuven, Belgium (e-mail: alexander.bertrand@esat.kuleuven.be).

[1]It is noted that, although we only focus on least squares problems, many results can be extended to other linear estimation frameworks as well [8].

it a highly attractive metric for model interpretation, signal quality assessment, greedy variable selection, etc., in particular in real-time or large-scale applications. We also address several generalizations and extensions of this utility metric towards:

- A *group-utility* metric, allowing to evaluate the joint utility of a group of input variables.
- A *noise-impact* metric, allowing to evaluate the impact of additive errors in the input variables, e.g., to predict the effect of quantization or measurement noise, and which contains the original utility metric as a special case.
- A *minimum-norm utility metric* for ill-conditioned cases, in which linear dependence relationships exist between the input variables.

The main goal of this paper is (a) to provide an accessible overview and unification of existing results in this context (with pointers to the original publications), and (b) introduce novel extensions and generalizations presented in a unified framework.

The outline of the paper is as follows. In Section II, we formalize the definition of the utility metric for LS regression, and provide the core equation to efficiently compute it with a complexity that scales linearly with the number of variables. In section III, we generalize these results to a group-utility metric, allowing to evaluate the joint utility of a group of input variables. A further generalization is provided in Section IV, allowing to evaluate the impact of additive errors in the input variables. Finally, in Section V, we extend the utility metric to cope with linear dependencies in the input variables.

II. Utility: definition and core equation

A. Definition

Consider the LS problem with N measurements of M input variables

$$J(Y) \triangleq \min_{\mathbf{x}} \frac{1}{N}\|Y\mathbf{x} - \mathbf{d}\|^2 \tag{1}$$

where $Y \in \mathbb{R}^{N\times M}$ is the regressor matrix, $\mathbf{d} \in \mathbb{R}^N$ is the desired response vector, and $\mathbf{x} \in \mathbb{R}^M$ is the vector with optimization variables (we consider the real-valued case for simplicity, yet all results in this paper can be easily generalized to the complex-valued case). Note that $J(Y)$ is defined as an operator that evaluates the LS cost for the case where the information in Y is available, which includes an implicit optimization of $\mathbf{x}$ (the reason for making the dependency on Y explicit will become clear later). We will refer to the columns of Y as the input variables of the model. Depending on the context, these input variables could represent, e.g., different sensors or channels (in a sensor array), time lags (in a temporal filter), observations of independent variables (in a regression model), etc. Assuming Y has full rank, the LS solution $\hat{\mathbf{x}}$ that minimizes (1) is given by

$$\hat{\mathbf{x}} = R^{-1}\mathbf{r} \tag{2}$$

with $R = \frac{1}{N}Y^TY$ and $\mathbf{r} = \frac{1}{N}Y^T\mathbf{d}$.

In order to quantify the relevance of each input variable, we define the utility metric [3], which will be the focus of this tutorial. The utility of the k-th input variable is defined as the increase in LS cost if the k-th input variable would be removed and if the LS problem would be re-optimized, i.e.,

$$U_k \triangleq J(Y_{-k}) - J(Y) \tag{3}$$

where Y_{-k} denotes the matrix Y with the k-th column removed. Note that a naive computation of U_k would in principle require to solve a second LS problem based on Y_{-k}, of which the computational complexity scales cubically with the number of input variables, i.e., $O(M^3)$. Calculating the utility of *all* M input variables would then have a complexity of $O(M^4)$, which can be unacceptably high for, e.g., real-time systems or for large-scale problems with hundreds or thousands of input variables. In the sequel, we will show how some simple linear algebra tricks allow to derive an efficient and elegant equation to calculate (3) for *all* input variables, with a total complexity of merely $O(M)$.

B. Core equation

Once the full LS solution (2) has been calculated, we will show that calculating the utility (3) does not require to solve an extra LS problem to evaluate $J(Y_{-k})$, i.e., it can be calculated as

$$U_k = \frac{|x_k|^2}{q_k} \tag{4}$$

where q_k is the k-th diagonal element of R^{-1}, and where x_k is the k-th element of $\hat{\mathbf{x}}$ in (2). This has originally been proven in [3], but we will derive a more general form of (4) in Section III, which will

then also prove (4) as a special case. From (4), it follows that the vector $\mathbf{u} = [U_1 \dots U_M]^T$ containing the utilities of *all* input variables can be calculated as

$$\mathbf{u} = \Lambda^{-1} |\hat{\mathbf{x}}|^2 \tag{5}$$

where $|\cdot|^2$ represents an element-wise squaring and $\Lambda = \mathcal{D}(R^{-1})$ with $\mathcal{D}(\cdot)$ the operator that creates a diagonal matrix by setting all the off-diagonal elements of the matrix in its argument to zero.

Note that the equations (4) and (5) are remarkably simple and elegant. Since R^{-1} is readily available from the computation of $\hat{\mathbf{x}}$ in (2), the utility can be calculated with a complexity of merely $O(1)$ for a single variable, or $O(M)$ for all variables. This should be contrasted to a naive computation of U_k or $\mathbf{u}$ based on the original utility definition (3), resulting in a complexity of $O(M^3)$ and $O(M^4)$, respectively.

III. Group-utility

In some applications, the input variables are naturally clustered in specific pre-defined groups, in which case it could make more sense to investigate the utility of groups of variables, rather than of individual variables. For example, in a multi-channel filter, the utility of a channel is the joint utility of all the filter taps in that channel's delay line. Similarly, in a sensor network with multi-sensor nodes, the utility of a node is the joint utility of all the sensor signals within that node [4].

Similar to (3), the group-utility of a pre-defined group of G input variables, denoted by the set $\mathcal{G}$, is defined as [4]

$$U_{\mathcal{G}} \triangleq J(Y_{\_\mathcal{G}}) - J(Y) \tag{6}$$

where $Y_{\_\mathcal{G}}$ is the matrix Y with all columns corresponding to the input variables in $\mathcal{G}$ removed. In the sequel, we assume that $\mathcal{G}$ consists of the last G columns of Y, which is without loss of generality (w.l.o.g.), as the order of the inputs can be arbitrarily rearranged. Define the following block partitioning of the (known) inverse of R

$$R^{-1} = \begin{bmatrix} A & B \\ B^T & Q \end{bmatrix} \tag{7}$$

where Q is the $G \times G$ matrix capturing the rows and columns with indices corresponding to the variables in $\mathcal{G}$ (here at the bottom right w.l.o.g.). As shown in **[Pop-out box 1]**, the group-utility $U_{\mathcal{G}}$ can efficiently be calculated as

$$U_{\mathcal{G}} = \mathbf{x}_{\mathcal{G}}^T \, Q^{-1} \, \mathbf{x}_{\mathcal{G}} \tag{8}$$

where $\mathbf{x}_{\mathcal{G}}$ contains the last G entries of $\hat{\mathbf{x}}$ (we do not add a hat to $\mathbf{x}_{\mathcal{G}}$ as it is not an optimal LS solution in itself). Note that this group-utility equation reduces to the original utility equation (4) if $G = 1$. Obviously, if $G \ll M$, computing (8) is much cheaper than evaluating $J(Y_{\_\mathcal{G}})$ in (6) by explicitly computing the reduced LS solution.

Although the derivation of (8) is not necessary to follow the rest of this tutorial, we include it in **[Pop-out box 1]** for completeness and because it also reveals two interesting by-products, namely two equations (10) and (13) that allow to recursively update (a) the inverse autocorrelation matrix R^{-1}, and (b) the LS solution $\hat{\mathbf{x}}$, after the removal of G input variables. This is interesting if the (group-)utility metric would be used for greedy variable selection, where (groups of) input variables are deleted one by one (see Section VI).

IV. Generalization towards noise impact

The utility metric as defined in (3) measures the increase in the LS cost when the k-th column of Y is removed. Another relevant metric would be to measure the increase in the LS cost when adding some random noise in the k-th column of Y, rather than fully removing that column. This is interesting in situations where one has some freedom in controlling the accuracy of each individual input variable. For example, in quantization or lossy compression, one can often modify the bit depth or the compression rate of each individual signal to reduce the resources required to store or to transmit it, while increasing its noise level. Similar trade-offs appear when deciding between cheap or accurate sensors, in applications or experiments where the accuracy depends on the measurement time, etc. In all these cases, it is important to be able to efficiently assess and quantify how additive noise on each particular input variable would affect the estimation performance, in particular when used in greedy or adaptive resource allocation schemes.

To quantify the effect of additive noise, the noise impact metric was originally defined in [11] with the

Pop-out box 1: Derivation of the group-utility core equation (8)

Note that evaluating $J(Y_{-\mathcal{G}})$ in (6) requires to solve the reduced LS solution

$$\hat{\mathbf{x}}_{-\mathcal{G}} = R_{-\mathcal{G}}^{-1}\,\mathbf{r}_{-\mathcal{G}} \tag{9}$$

with $R_{-\mathcal{G}} = \frac{1}{N}Y_{-\mathcal{G}}^T\,Y_{-\mathcal{G}}$ and $\mathbf{r}_{-\mathcal{G}} = \frac{1}{N}Y_{-\mathcal{G}}^T\,\mathbf{d}$. The first step in our derivation is to find a more efficient way to calculate $R_{-\mathcal{G}}^{-1}$, based on a subresult of the blockwise matrix inversion theorem [10]:

Lemma III.1. *Consider the block partitioning of a matrix V and its inverse V^{-1} as follows*

$$V = \begin{bmatrix} A & B \\ C & D \end{bmatrix}, \quad V^{-1} = \begin{bmatrix} E & * \\ * & * \end{bmatrix}$$

with A and E square matrices of equal size. If D and E are invertible, then $E^{-1} = A - BD^{-1}C$.

By setting $V = R^{-1}$ (consequently $E = R_{-\mathcal{G}}$), and using the notation in (7), the lemma immediately yields the following important result

$$\boxed{R_{-\mathcal{G}}^{-1} = A - BQ^{-1}B^T}\,. \tag{10}$$

By plugging (10) in (9), the reduced LS solution is given by

$$\hat{\mathbf{x}}_{-\mathcal{G}} = (A - BQ^{-1}B^T)\mathbf{r}_{-\mathcal{G}}\,. \tag{11}$$

Using the partitioning in (7), we can define the following partitioning of the LS solution (2):

$$\hat{\mathbf{x}} = \begin{bmatrix} \mathbf{x}_{-\mathcal{G}} \\ \mathbf{x}_{\mathcal{G}} \end{bmatrix} = \begin{bmatrix} A\,\mathbf{r}_{-\mathcal{G}} + B\,\mathbf{r}_{\mathcal{G}} \\ B^T\mathbf{r}_{-\mathcal{G}} + Q\,\mathbf{r}_{\mathcal{G}} \end{bmatrix} \tag{12}$$

where $\mathbf{r}_{-\mathcal{G}}$ and $\mathbf{r}_{\mathcal{G}}$ denote subvectors of $\mathbf{r}$ containing the first $M-G$ and last G entries, respectively. From (11) and (12), it can be easily verified that

$$\boxed{\hat{\mathbf{x}}_{-\mathcal{G}} = \mathbf{x}_{-\mathcal{G}} - BQ^{-1}\mathbf{x}_{\mathcal{G}}} \tag{13}$$

which allows to efficiently update the LS solution. By expanding the LS cost functions in (6) in their quadratic terms, and plugging in the corresponding LS solutions (2) and (9) for $\hat{\mathbf{x}}_{-\mathcal{G}}$ and $\hat{\mathbf{x}}$, respectively, it can be straightforwardly found that

$$U_{\mathcal{G}} = \mathbf{r}^T\hat{\mathbf{x}} - \mathbf{r}_{-\mathcal{G}}^T\hat{\mathbf{x}}_{-\mathcal{G}}\,. \tag{14}$$

By plugging in (13), and by partitioning $\hat{\mathbf{x}}$ in $\mathbf{x}_{-\mathcal{G}}$ and $\mathbf{x}_{\mathcal{G}}$, we immediately find that

$$U_{\mathcal{G}} = \mathbf{r}_{\mathcal{G}}^T\mathbf{x}_{\mathcal{G}} + (\mathbf{r}_{-\mathcal{G}}^T B)Q^{-1}\mathbf{x}_{\mathcal{G}}\,. \tag{15}$$

From the bottom half of (12), it follows that $\mathbf{r}_{-\mathcal{G}}^T B = \mathbf{x}_{\mathcal{G}}^T - \mathbf{r}_{\mathcal{G}}^T Q$, such that (15) eventually yields (8).

purpose of performing a greedy signal quantization. For the sake of completeness and unification, we generalize this result to a group-impact metric in this paper, which has the result of [11] as a special case, and which generalizes the group-utility metric (8). Let $Y_{\mathcal{G},\Sigma}$ denote the matrix Y in which zero-mean random noise is added to the input variables in $\mathcal{G}$, with a positive definite noise covariance matrix $\Sigma \in \mathbb{R}^{G\times G}$. In most cases, the noise will be uncorrelated across the input variables, in which case Σ is a diagonal matrix.

In line with (6), we define the noise impact on the group $\mathcal{G}$ as

$$I_{\mathcal{G}}(\Sigma) \triangleq \min_{\mathbf{x}} \frac{1}{N}\mathcal{E}\{\|Y_{\mathcal{G},\Sigma}\,\mathbf{x} - \mathbf{d}\|^2\} - J(Y) \tag{18}$$

where $\mathcal{E}\{\cdot\}$ denotes the expected value operator,

Pop-out box 2: Derivation of the noise-impact core equation (19)

As the added noise is zero mean and uncorrelated to Y and $\mathbf{d}$, the LS solution of the first term is equal to

$$\hat{\mathbf{x}}_{\mathcal{G},\Sigma} = R_F^{-1}\mathbf{r} = \left(R + FF^T\right)^{-1}\mathbf{r}$$

where $F = \begin{bmatrix} O & \Sigma^{1/2} \end{bmatrix}^T$ with O the all-zero matrix. Note that $\mathbf{r}$ is unaffected by the noise due to the expected value operator and the fact that the noise is uncorrelated to $\mathbf{d}$. Applying the Sherman-Morrison-Woodbury identity [10] to R_F yields

$$R_F^{-1} = R^{-1} - R^{-1}F\left(I + F^T R^{-1} F\right)^{-1} F^T R^{-1} .$$

With the partitioning of R^{-1} in (7), this becomes

$$R_F^{-1} = R^{-1} - \begin{bmatrix} B \\ Q \end{bmatrix} \Sigma^{\frac{1}{2}} \left(I + \Sigma^{\frac{1}{2}} Q \Sigma^{\frac{1}{2}}\right)^{-1} \Sigma^{\frac{1}{2}} \begin{bmatrix} B^T & Q \end{bmatrix} \tag{16}$$

$$= R^{-1} - \begin{bmatrix} B \\ Q \end{bmatrix} \left(\Sigma^{-1} + Q\right)^{-1} \begin{bmatrix} B^T & Q \end{bmatrix} . \tag{17}$$

Similar to (14), it can easily be verified that the noise impact (18) is equal to

$$\begin{aligned} I_{\mathcal{G}}(\Sigma) &= \mathbf{r}^T\hat{\mathbf{x}} - \mathbf{r}^T\hat{\mathbf{x}}_{\mathcal{G},\Sigma} \\ &= \mathbf{r}^T(R^{-1} - R_F^{-1})\mathbf{r} . \end{aligned}$$

Plugging (16) into this equation, and using the fact that $\mathbf{x}_{\mathcal{G}} = [B^T\ Q]\mathbf{r}$ (see (12)), we eventually find (19).

which is introduced due to the stochastic nature of the first term. In **[Pop-out box 2]**, we show that (18) can be efficiently calculated as

$$I_{\mathcal{G}}(\Sigma) = \mathbf{x}_{\mathcal{G}}^T \left(\Sigma^{-1} + Q\right)^{-1} \mathbf{x}_{\mathcal{G}} \tag{19}$$

where we again assumed that $\mathcal{G}$ contains the last G columns of Y w.l.o.g. This equation allows to compute the noise impact of the input variables in $\mathcal{G}$ using the original LS solution (remember that $\mathbf{x}_{\mathcal{G}}$ consists of the last G entries of $\hat{\mathbf{x}}$ as in (8)). For the case where $G = 1$, i.e., when evaluating the impact of noise with variance σ^2 on a *single* input variable, (19) reduces to the elegant equation (compare with (4))

$$I_k(\sigma^2) = \frac{|x_k|^2}{\frac{1}{\sigma^2} + q_k} . \tag{20}$$

This noise impact metric I_k can be viewed as a generalization of the utility metric U_k, as a comparison between (4) and (20) shows that $I_k \to U_k$ if $\sigma^2 \to \infty$. This should not come as a surprise, as adding infinitely large noise to the observations of the k-th input variable essentially results in the same loss of information as when the k-th input variable would be removed. Similarly, the group-impact equation (19) reduces asymptotically to the group-utility equation (8) if the diagonal entries in Σ grow to infinity.

V. Redundant Input Variables

If there is redundancy in the set of input variables, i.e., there is a linear dependency or almost perfect correlation between some of the columns in Y, then the solution of (1) becomes non-unique or ill-conditioned. A common strategy is then to compute the LS solution with the smallest $\ell 2$-norm, which is advantageous against overfitting [5], [12]. In the sequel, we denote $\mathcal{R}$ as the set containing all input variables that are redundant, i.e., all columns of Y that consist of a linear combination of the other columns of Y. Note that, by definition, $U_k = 0$ for $k \in \mathcal{R}$, as the removal of a redundant variable does not impact the LS cost.

If $\mathcal{R}$ is non-empty, then R^{-1} does not exist and the LS solution of (1) is not unique, in which case the LS solution with minimal $\ell 2$-norm is given by

$$\hat{\mathbf{x}} = Y^{+}\mathbf{d} = R^{+}\mathbf{r} \tag{21}$$

where R^{+} denotes the Moore-Penrose pseudo-inverse of R, and where the second equality follows from the identity $X^{+} = (X^{T}X)^{+}X^{T}$, which holds for the pseudo-inverse of any matrix X [10], [13]. As R^{-1} simply has to be replaced with R^{+} in (2), it is then tempting to also compute the utility U_k by setting q_k in (4) equal to the k-th diagonal element of R^{+} instead of R^{-1}. Although it can be shown that this yields the correct utility values U_k for the non-redundant variables $k \notin \mathcal{R}$, it will result in incorrect (non-zero) utility values for the redundant variables $k \in \mathcal{R}$. The proof of this statement is omitted for conciseness, but follows relatively straightforwardly from some subresults in [14].

To fix this issue, we have to modify (4) to enforce that U_k is small (near-zero) for $k \in \mathcal{R}$, while non-redundant variables $k \notin \mathcal{R}$ should receive a non-zero U_k that approximates (3). Furthermore, although the removal of a redundant variable will not affect the LS cost, it will increase the $\ell 2$-norm, i.e., $\|\hat{\mathbf{x}}_{-k}\| \geq \|\hat{\mathbf{x}}\|$ for $k \in \mathcal{R}$. To maximally avoid overfitting, we would like the modified utility measure to also reflect this change in norm, such that removing the redundant input value with the lowest modified utility also induces the least increase of the $\ell 2$-norm. We will show that both of these goals can be achieved if we generalize the utility definition to a standard ridge regression framework .

In ridge regression, an $\ell 2$-norm penalty is added to the LS cost function [10], i.e., (1) becomes

$$\min_{\mathbf{x}} \frac{1}{N}\left(\|Y\mathbf{x} - \mathbf{d}\|^2 + \lambda\|\mathbf{x}\|^2\right) \tag{22}$$

where λ is a user-defined regularization parameter, and which has

$$\hat{\mathbf{x}} = R_{\lambda}^{-1}\mathbf{r} = (R + \lambda I)^{-1}\,\mathbf{r} \tag{23}$$

as a minimizer [10]. Let us now define utility as we did before in (3), but this time based on the regularized cost function (22) instead, i.e.,

$$\begin{aligned} U_k(\lambda) \triangleq \frac{1}{N}\left(\|Y_{-k}\hat{\mathbf{x}}_{-k} - \mathbf{d}\|^2 - \|Y\hat{\mathbf{x}} - \mathbf{d}\|^2\right) \\ + \frac{\lambda}{N}\left(\|\hat{\mathbf{x}}_{-k}\|^2 - \|\hat{\mathbf{x}}\|^2\right) \end{aligned} \tag{24}$$

Note that we have not used the '$\min$' operators this time, but instead we plugged in the minimizers to explicitly separate the increase in LS cost in the first term and the increase in $\ell 2$-norm in the second term.

The following three theoretical results, which are proven in the Appendix, demonstrate that this new utility definition (24) indeed resolves the aforementioned issues and can still be calculated efficiently (these results can also easily be extended to the group-utility framework):

Result 1 (efficient calculation): The modified utility $U_k(\lambda)$ defined in (24) can be calculated using the efficient formula (4) where q_k is now set to the k-th diagonal element of R_{λ}^{-1} instead of R^{-1}.

Result 2 (consistency): If $0 < \lambda \ll \epsilon$, with ϵ the smallest non-zero eigenvalue of R, then $U_k(\lambda) \approx 0$ if $k \in \mathcal{R}$, and $U_k(\lambda) \approx U_k$ if $k \notin \mathcal{R}$, where the approximations become asymptotically exact for an arbitrarily small λ.

Result 3 (minimum-norm revealing): If λ is sufficiently small, $U_k(\lambda)$ will be smallest for $k \in \mathcal{R}$ that results in the smallest $\ell 2$-norm $\|\hat{\mathbf{x}}_{-k}\|$ after removal of input k. More specifically,

$$\forall\, k \in \mathcal{R}:\ \frac{U_k(\lambda)}{\lambda} \approx \|\hat{\mathbf{x}}_{-k}\|^2 - \|\hat{\mathbf{x}}\|^2$$

where the approximation becomes asymptotically exact for an arbitrarily small λ.

To validate these results, we have calculated the modified utility metric (24) on a toy example with random data with $M = 20$ input variables. The last 5 columns of Y are generated as random linear combinations of columns 11 to 15, such that $\mathcal{R} = \{11, \ldots, 20\}$. We set $\lambda = \epsilon/100$, where ϵ denotes the smallest non-zero eigenvalue of R. Fig. 1 (left figure) shows the values $U_k(\lambda)$ for $k = 1 \ldots 20$, in blue for a naive calculation based on the definition (24) using (21) to find $\hat{\mathbf{x}}$ and $\hat{\mathbf{x}_{-k}}$, and in red when calculated using the efficient equation (5) where R is replaced with R_{λ} (see Result 1). It can be observed that both calculation methods result in the same utility value. Note that the 10 redundant variables have an almost-zero utility, which is consistent with Result 2. To validate Result 3, we zoom in on

the 10 redundant variables (right plot in Fig. 1) and plot the difference $\|\hat{\mathbf{x}}_{-k}\|^2 - \|\hat{\mathbf{x}}\|^2$ (in green) versus the value $\frac{U_k(\lambda)}{\lambda}$, where we observe that both result in the same value. This allows to select the redundant input variable that will yield the smallest increase in $\ell 2$-norm when removed (in this case variable 14).

VI. Computational benefits and implications for variable subset selection

To demonstrate the impressive reduction in computation time achieved by the core equations (4)-(5) and their generalizations/extensions, we measured the calculation times on a standard laptop running Matlab. In Fig. 2, we compare the time to compute the complete utility vector $\mathbf{u} = [U_1 \ \dots \ U_M]^T$ when using the efficient equation (5) and using a naive calculation based on the definition (3), as a function of the number of input variables M. We performed the naive calculation two times: once with and once without redundant input variables, where (21) is used to find the minimum-norm LS solution in the former case. In both cases, the computation time is several orders of magnitude lower when using (5).

These strong computational simplifications facilitate the use of (group-)utility metrics for a backward greedy variable selection procedure -even in large-scale problems- in which the input variables with lowest utility are recursively removed one by one, until a sufficiently small set is obtained or until any removal would result in a too large increase in LS cost [1]–[4], [6]. This can be viewed as an alternative for the well-known (group-)LASSO algorithm [9], [15]. The backward greedy algorithm can even be shown to be optimal if an exact or almost-exact sparse solution exists [1]. In a similar fashion, the noise impact metrics (19)-(20) can be used for, e.g., a greedy adaptive quantization, where in each iteration a certain amount of quantization noise is added to the input with the lowest noise-impact [11], [16]. Finally, a utility-based greedy variable selection based on (24) yields a combination of variable selection with $\ell 2$-norm minimization, which is akin to the so-called elastic net procedure [17].

Overall, utility-based greedy versions have some useful properties, viz., they bypass the tedious tuning of sparsity-inducing regularization parameters, they are cheap to compute, and they are easy to implement. Furthermore, the low computational complexity is particularly attractive for online utility tracking, e.g., in recursive LS adaptive filters to (temporarily) eliminate signals of which the utility goes under a pre-defined threshold [3] or to guarantee that the overall loss does not exceed a pre-defined threshold. Note that, every time a (group of) input variable(s) $\mathcal{G}$ is removed, the LS solution and inverse autocorrelation matrix have to be updated according to the remaining set of variables. These updates can be efficiently calculated using equations (10) and (13) in **[Pop-out box 1]**. Indeed, after the removal of the input variables in $\mathcal{G}$, these equations allow to recursively update the inverse of the reduced autocorrelation matrix $R_{\mathcal{G}}^{-1}$ and the corresponding LS solution at a low cost, based on the original R^{-1} and $\hat{\mathbf{x}}$. For the non-grouped case, i.e., $G = 1$, the matrix inversion of Q in (10) and (13) reduces to a simple scalar inversion.

If one also wants to monitor the utility of input variables that are to be *added* to the model, e.g., for a forward greedy variable selection instead of backwards deletion, there also exist additive versions of the utility metric with efficient calculation schemes [3], [4]. However, it should be noted that these metrics are less elegant and computationally less attractive than the deletion-based utility metrics that were introduced above, yet they are still more efficient than a naive 'brute-force' computation.

VII. Conclusions

In this paper, we have reviewed and unified the core equations for the efficient calculation of utility metrics, and extended these towards group-metrics and a minimum-norm revealing utility metric. All these metrics can be elegantly and cheaply calculated, thereby making them attractive as a 'quick-and-dirty' tool for model interpretation, online signal quality assessment, or greedy variable selection.

References

[1] C. Couvreur and Y. Bresler, "On the optimality of the backward greedy algorithm for the subset selection problem," *SIAM Journal on Matrix Analysis and Applications*, vol. 21, no. 3, pp. 797–808, 2000. [Online]. Available: https://doi.org/10.1137/S0895479898332928

[2] L. Scott and B. Mulgrew, "Sparse LCMV beamformer design for suppression of ground clutter in airborne radar," *IEEE Transactions on Signal Processing*, vol. 43, no. 12, pp. 2843–2851, Dec 1995.

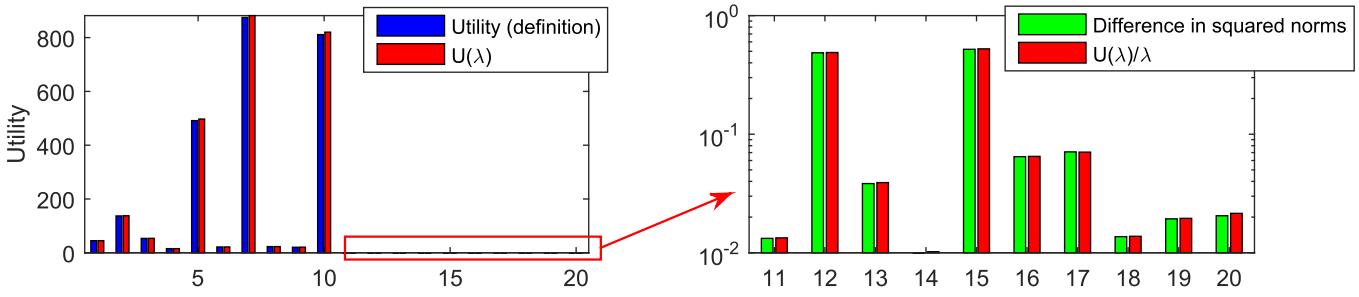

Fig. 1. Validation of the extended utility metric for the case where the last 10 input variables are redundant.

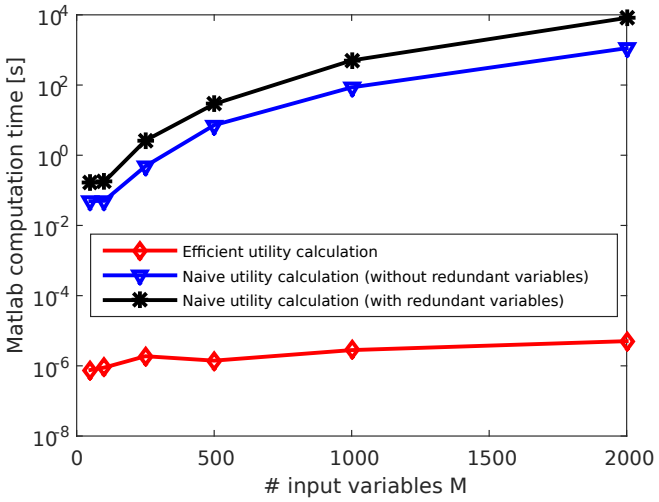

Fig. 2. Computing time in Matlab to calculate the utility of M input variables using the efficient equation (5) and using a naive calculation based on the definition (3).

[3] A. Bertrand and M. Moonen, "Efficient sensor subset selection and link failure response for linear MMSE signal estimation in wireless sensor networks," in *Proc. of the European signal processing conference (EUSIPCO)*, Aalborg - Denmark, Aug. 2010, pp. 1092–1096.

[4] J. Szurley, A. Bertrand, P. Ruckebusch, I. Moerman, and M. Moonen, "Greedy distributed node selection for node-specific signal estimation in wireless sensor networks," *Signal Processing*, vol. 94, pp. 57–73, Jan. 2014.

[5] C. Bishop, *Pattern Recognition and Machine Learning*. New York: Springer-Verlag, 2006.

[6] J. Zhang, S. P. Chepuri, R. C. Hendriks, and R. Heusdens, "Microphone subset selection for MVDR beamformer based noise reduction," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 26, no. 3, pp. 550–563, March 2018.

[7] S. P. Chepuri and G. Leus, "Sensor selection for estimation, filtering, and detection," in *2014 International Conference on Signal Processing and Communications (SPCOM)*, July 2014, pp. 1–5.

[8] A. Bertrand, J. Szurley, P. Ruckebusch, I. Moerman, and M. Moonen, "Efficient calculation of sensor utility and sensor removal in wireless sensor networks for adaptive signal estimation and beamforming," *IEEE Transactions on Signal Processing*, vol. 60, no. 11, pp. 5857–5869, 2012.

[9] R. Tibshirani, "Regression shrinkage and selection via the lasso: a retrospective," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 73, no. 3, pp. 273–282, 2011.

[10] G. H. Golub and C. F. van Loan, *Matrix Computations*, 3rd ed. Baltimore: The Johns Hopkins University Press, 1996.

[11] F. de la Hucha Arce, F. Rosas, M. Moonen, M. Verhelst, and A. Bertrand, "Generalized signal utility for LMMSE signal estimation with application to greedy quantization in wireless sensor networks," *IEEE Signal Processing Letters*, vol. 23, no. 9, pp. 1202–1206, Sept 2016.

[12] C. Zhang, S. Bengio, M. Hardt, B. Recht, and O. Vinyals, "Understanding deep learning requires rethinking generalization," 2017. [Online]. Available: arXiv:1611.03530v2

[13] K. B. Petersen and M. S. Pedersen, "The matrix cookbook." [Online]. Available: https://archive.org/details/imm3274

[14] C. A. Rohde, "Generalized inverses of partitioned matrices," *Journal of the Society for Industrial and Applied Mathematics*, vol. 13, no. 4, pp. 1033–1035, 1965.

[15] M. Yuan and Y. Lin, "Model selection and estimation in regression with grouped variables," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 68, no. 1, pp. 49–67, 2006.

[16] F. de la Hucha Arce, M. Moonen, M. Verhelst, and A. Bertrand, "Adaptive quantization for multichannel Wiener filter-based speech enhancement in wireless acoustic sensor networks," *Wireless Communications and Mobile Computing*, vol. 2017, Article ID 3173196, 2017.

[17] H. Zou and T. Hastie, "Regularization and variable selection via the elastic net," *Journal of the Royal Statistical Society, Series B*, vol. 67, pp. 301–320, 2005.

[18] S. M. Selby, *Standard Mathematical Tables*. CRC press, 1974.

APPENDIX

In this appendix, we provide the outline of the proofs for the three results listed in Section V.

Result 1: It can be easily verified that the derivation in **[Pop-out box 1]** is also valid for $U_k(\lambda)$ if R is replaced with R_λ everywhere.

Result 2: If $\lambda \to 0$, the second term in (24) will vanish, so it can be ignored. Furthermore, it is a known fact that

$$\lim_{\lambda \to 0} R_\lambda^{-1}\mathbf{r} = R^{+}\mathbf{r} \tag{25}$$

which holds even if R^{-1} does not exist (see, e.g., page 263 in [10]). This means that $\hat{\mathbf{x}}$ and $\hat{\mathbf{x}}_{-k}$ get

asymptotically close to the minimum-norm LS solution of the non-regularized cost (1) when $\lambda \to 0$. As a result, the first term gets asymptotically close to U_k according to the original definition of the non-regularized utility in (3), which is by definition equal to zero if $k \in \mathcal{R}$.

Result 3: If $k \in \mathcal{R}$, then both terms in (24) will vanish if $\lambda \to 0$, i.e., $U_k(\lambda) \to 0$ (see Result 2). Note that the second term vanishes linearly with λ. Therefore, to prove Result 3, we have to show that the first term vanishes superlinearly with λ, $\forall\, k \in \mathcal{R}$, such that the second term dominates over the first term if λ becomes small. To this end, we study $\lim_{\lambda\to 0} U_k(\lambda)/\lambda$ instead. Based on l'Hôpital's rule, we find that

$$\forall\, k \in \mathcal{R}:\ \lim_{\lambda\to 0} \frac{U_k(\lambda)}{\lambda} = \lim_{\lambda\to 0} \frac{dU_k(\lambda)}{d\lambda} \tag{26}$$

i.e., the limit is obtained by taking the derivative of $U_k(\lambda)$. By plugging (23) into (24) and expanding it in its quadratic terms, we find that the derivative of (24) can be calculated as

$$\frac{dU_k(\lambda)}{d\lambda} = \frac{d}{d\lambda}\left(\mathbf{r}^T R_\lambda^{-1}\mathbf{r} - \mathbf{r}_{-k}^T R_{\lambda,-k}^{-1}\mathbf{r}_{-k}\right) \tag{27}$$

where $R_{\lambda,-k}$ denotes the matrix R_λ with the k-th row and column removed, and $\mathbf{r}_{-k}$ denotes $\mathbf{r}$ with the k-th entry removed. The following basic identity is found in [13], [18] for an invertible matrix A:

$$\frac{dA^{-1}}{dx} = -A^{-1}\,\frac{dA}{dx}\,A^{-1}\ .$$

Using this identity, it can be straightforwardly found that (27) reduces to

$$\frac{dU_k(\lambda)}{d\lambda} = \left(\mathbf{r}_{-k}^T R_{\lambda,-k}^{-2}\mathbf{r}_{-k} - \mathbf{r}^T R_\lambda^{-2}\mathbf{r}\right)\ .$$

Taking the limit for $\lambda \to 0$ and using (25) and (26), this reduces to

$$\begin{aligned}\forall\, k \in \mathcal{R}:\ \lim_{\lambda\to 0} \frac{U_k(\lambda)}{\lambda} &= \|R_{-k}^{+}\mathbf{r}_{-k}\|^2 - \|R^{+}\mathbf{r}\|^2 \\ &= \|\hat{\mathbf{x}}_{-k}\|^2 - \|\hat{\mathbf{x}}\|^2\end{aligned}$$

where the last equality immediately follows from (21). This shows that selecting the redundant variable $k \in \mathcal{R}$ with the lowest utility, will induce the lowest increase in $\ell 2$-norm, which proves Result 3.