Mechanical Systems and Signal Processing 117 (2019) 383–402

Contents lists available at ScienceDirect

Mechanical Systems and Signal Processing

journal homepage: www.elsevier.com/locate/ymssp

A probabilistic compressive sensing framework with applications to ultrasound signal processing

Ramon Fuentes [a,*], Carmelo Mineo [b], Stephen G. Pierce [b], Keith Worden [a], Elizabeth J. Cross [a]

[a] Dynamics Research Group, The University of Sheffield, United Kingdom
[b] Centre for Ultrasonic Engineering, Strathclyde University, United Kingdom

ARTICLE INFO

Article history:
Received 20 January 2018
Received in revised form 21 June 2018
Accepted 21 July 2018
Available online 16 August 2018

Keywords:
NDT
Compressive sensing
Ultrasound
Signal processing
Relevance Vector Machine
Bayesian methods
Sparse representation

ABSTRACT

The field of Compressive Sensing (CS) has provided algorithms to reconstruct signals from a much lower number of measurements than specified by the Nyquist-Shannon theorem. There are two fundamental concepts underpinning the field of CS. The first is the use of random transformations to project high-dimensional measurements onto a much lower-dimensional domain. The second is the use of sparse regression to reconstruct the original signal. This assumes that a sparse representation exists for this signal in some known domain, manifested by a dictionary. The original formulation for CS specifies the use of an l_1 penalised regression method, the Lasso. Whilst this has worked well in literature, it suffers from two main drawbacks. First, the level of sparsity must be specified by the user, or tuned using sub-optimal approaches. Secondly, and most importantly, the Lasso is not probabilistic; it cannot quantify uncertainty in the signal reconstruction. This paper aims to address these two issues; it presents a framework for performing compressive sensing based on sparse Bayesian learning. Specifically, the proposed framework introduces the use of the Relevance Vector Machine (RVM), an established sparse kernel regression method, as the signal reconstruction step within the standard CS methodology. This framework is developed within the context of ultrasound signal processing in mind, and so examples and results of compression and reconstruction of ultrasound pulses are presented. The dictionary learning strategy is key to the successful application of any CS framework and even more so in the probabilistic setting used here. Therefore, a detailed discussion of this step is also included in the paper. The key contributions of this paper are a framework for a Bayesian approach to compressive sensing which is computationally efficient, alongside a discussion of uncertainty quantification in CS and different strategies for dictionary learning. The methods are demonstrated on an example dataset from collected from an aerospace composite panel. Being able to quantify uncertainty on signal reconstruction reveals that this grows as the level of compression increases. This is key when deciding appropriate compression levels, or whether to trust a reconstructed signal in applications of engineering and scientific interest.

1. Introduction

The Nyquist-Shannon theorem is at the centre of most traditional signal processing applications. It states that the frequency resolution obtainable in a signal is given by half of its time resolution, or sample rate. It is at the heart of frequency

* Corresponding author.
E-mail address: ramon.fuentes@sheffield.ac.uk (R. Fuentes).

https://doi.org/10.1016/j.ymssp.2018.07.036
0888-3270/

spectrum analysis methods based on Fourier and wavelet transforms. Compression is an important task in today's scientific computing. From a signal processing perspective, one of the most successful compression schemes is that of shrinkage, or thresholding [1] in a transform domain such as Fourier or wavelet. The problem with this, and any other compression scheme that relies on the application of a transform, is in the inherent wastefulness in the process of first acquiring a full data set in order to then compute the transform and the compression scheme.

Compression usually leverages sparsity: the idea that a signal that is dense in one domain can be sparse in another domain. The field of Compressive Sensing (often also called Compressive Sampling) (CS) was developed following the insight that compression can be achieved whilst also by-passing the usual procedure of first acquiring a full signal, and then transforming it into another "sparse" domain [2,3]. In the particular field of ultrasound signal processing, for example, it has already been shown that using a wavelet transform can achieve as much as a 95% compression ratio [4,5]. However, this requires both the acquisition of the full data set and the computation of a wavelet, or other transform. The main idea in CS is to circumvent the wastefulness of acquiring a large number of samples if one knows that most of them will be discarded anyway.

The contribution of this paper is a formulation of a probabilistic CS scheme. This is brought about through the use of the Relevance Vector Machine (RVM), a sparse Bayesian learning technique developed by Tipping [6]. The approach taken here is to replace the Lasso with an RVM in the sparse coding step of CS. This is a simple idea with profound implications. The result is a signal reconstruction that is fully probabilistic: it involves a mean and variance around the prediction, so that a confidence interval can be established regarding the quality of the signal estimate. Being a Bayesian method, it also naturally solves the problem of the appropriate level of sparsity with little user intervention. The estimation of uncertainty in the predictions is useful; it adds a layer to the understanding of prediction quality, which is paramount if compressing signals of scientific interest, or in safety critical applications.

2. Ultrasound-based NDT

The motivation for developing this CS framework is as an aid in the analysis of ultrasound waveforms, commonly used for Non-Destructive Testing (NDT). Ultrasound-based NDT has long been used in the structural integrity assessment of engineering components. The method, akin to the echo-location principle used by bats, relies on estimating the distance to an object by emitting a sound wave and listening to the response. The time it takes to receive the sound wave back can be used as a proxy to the location of the object, given some knowledge of the speed of sound properties of the medium. In order to use this principle to detect flaws in materials, one has to assume that a measurable amount of energy will be reflected back at the boundary between the medium and the flaw. Back at the source of the sound, this is measured as an echo. By mapping the time-of-flight (TOF) of these echoes across the surface of a material, it is possible to create a "depth map" otherwise known as a C-scan.

One of the characteristic features of ultrasound-based NDT is the acquisition of large quantities of high frequency data in the form of sound waveforms. Due to the high frequencies involved, often in the range between one and ten MHz, high sample rates are required to capture these waveforms, and this results in large quantities of data.

There is, however, a large disparity between the information content in a given ultrasound waveform, and the number of data points recorded in the time history of a waveform. So far, industry has solved this problem by extracting two key features from the echoes of ultrasound pulses: their attenuation and TOF difference. These features can yield useful information about material specimens and engineering structures if they are related to the material properties. The simplest and most widely used being the connection between TOF and material thickness, given the speed of sound of a material [7].

TOF estimation has therefore attracted significant attention from the NDT community. The methods developed over the years can be split into two categories: 1) those methods that use thresholds and changes in signal phase in order to separate the main pulse from the resulting echoes and compute the time differences between these two [8], and 2) those that use physical insight and attempt to solve a deconvolution problem to recover the impulse response function of the material being scanned [9,10]. The idea of estimating the full impulse response function of the material under question can be more attractive than characterising it with a few summarising features (such as a TOF) as this would capture all the information contained through the depth of a material. One interesting thing, from the point of view of this paper, is that the blind deconvolution problem is equivalent to the sparse coding step in compressive sensing, under an appropriate dictionary.

2.1. Features of ultrasound pulses

The Bayesian CS framework being presented will be demonstrated on ultrasound C-scan data from a carbon fibre composite wing panel. Although the results for this will be presented in Section 6, some key features of ultrasound signal processing will be introduced here, mostly as a motivation for the development of the method.

A typical ultrasound pulse is shown in Fig. 1a, with two time indices marked as t_a and t_b. These times correspond to reflections from the front and back wall of the composite panel respectively. A pulse of this kind effectively constitutes an A-scan. The information extracted from this is the time difference $t_f = t_b - t_a$, and this is often referred to as the ultrasonic TOF. This can be related to the thickness of the plate, if the propagation speed of bulk waves for the material is known. Another feature of interest is the ratio $x(t_a)/x(t_b)$ (where $x(t)$ is the measured amplitude of the ultrasound pulse), as this contains information

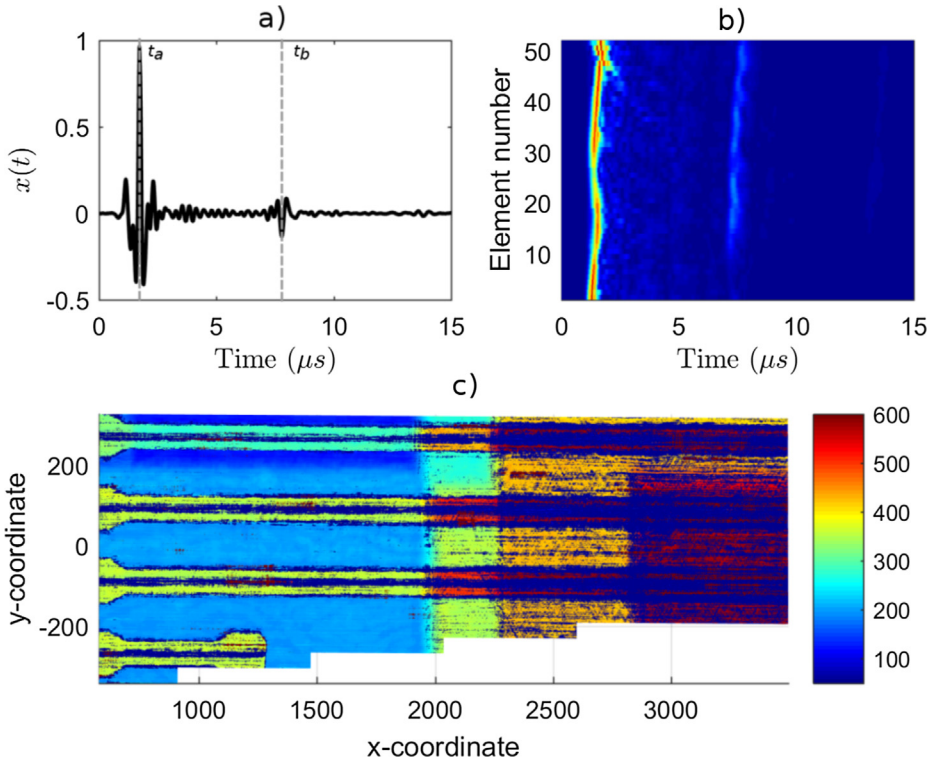

Fig. 1. Illustration of a) single ultrasound reflection (A-scan), b) reflections along a phased array probe (B-scan) and c) time of flight map along the two dimensions of a composite plate (C-scan).

about the attenuation of the wave as it travelled through the thickness of the plate. An A-scan thus gives information about a single physical coordinate on a surface.

A B-scan can be formed by collecting a series of A-scans along a line (illustrated in Fig. 1b), while a C-scan is formed by collecting a series of B-scans, to give a two-dimensional grid of ultrasound pulse information (illustrated in Fig. 1c). Note that higher times of flight in Fig. 1c imply wider plate thickness. The salient features in Fig. 1c are the stringers and the variable thickness of the wing panel. The purpose of the stringer is to increase the flexural stiffness in the main direction of bending. They are, therefore, characterised as thicker regions of material, typically along one direction of a plate. The stringers are evident by a low TOF index, due to their thickness being higher than the maximum value that can be captured within the collected data. These are evident as horizontal lines in Fig. 1c. Immediately surrounding the stiffeners, the stringer's feet are evident, as a reinforcement and to alleviate stress concentrations in the joint between the stringer and the main plate. The addition of layers is evident as discrete increases in TOF (implying higher material thickness).

2.2. TOF inference as a blind de-convolution problem

One way in which a greater level of physical insight could be extracted from a pulse-echo ultrasound measurement is by interpreting the measured response as a convolution process between the transducer impulse response function and the reflectivity function of the medium by which the sound is propagating [11],

$$x(t) = f(t) * r(t) \tag{1}$$

where $x(t)$ is the measured signal in the transducer, $f(t)$ is the transducer impulse response function, and $r(t)$ is a reflectivity function. This can be written down in discrete form as,

$$x(t) = \sum_{m=1}^{M} f(t - \tau_m) r(m) \tag{2}$$

where τ_m is a lag term. It is common to assume an impulse response function in the form of a windowed Gaussian tone burst [11]. This leads to a real-valued Gabor function,

$$f(\boldsymbol{\theta}, t) = Be^{-\alpha t^2} cos(\omega t + \phi). \tag{3}$$

The vector $\boldsymbol{\theta} = (\alpha, \omega, \phi)$ contains the parameters of this tone burst. Solving for the appropriate $\boldsymbol{\theta}$ that best represents the measured signal, $x(t)$, is a problem of de-convolution and it is an ill-posed one. It has been tackled successfully [9] using some of the sparsity tools discussed later in Section 3.3. Furthermore, this model is important as it represents one way of viewing what will shortly be referred to as a dictionary, in the context of sparse coding. Some attempt by the authors has been made at treating this de-convolution problem with CS tools [12], although the present paper presents a much more general view of the problem, in which the de-convolution interpretation is just a special case. This will be more evident later in Section 5 where dictionary learning will be discussed.

3. Compressive sensing

The concept of CS is to circumvent the process of explicitly using a forward transform in order to compress a data set. The idea being that if the compression is done by means of random sampling (which map a highly sampled signal into a much lower-dimensional domain), the original signal can be reconstructed later on.

There are three key ingredients in CS that allow accurate signal reconstructions based on a very low number of measurements. The first ingredient is the assumption that the signal in question can be represented by a very low number of coefficients in some suitable transformation. In other words, it is sparse in that domain. This domain is represented by a *dictionary*. The second ingredient is the use of random transformations of the signal. It can be shown that when a signal undergoes certain random transformations, pairwise distances between measured points are preserved. This is a result of the Johnson-Lindenstrauss Lemma [13]. However, recovering this compressed version of the signal from an over-complete dictionary is an ill-posed regression problem. This is where the third and final ingredient comes into play: sparse coding, or sparse regression methods. These allow the solution of these type of ill-posed problems by assuming that most of the coefficients in the solution will be zero.

3.1. Dictionaries and the sparsity assumption

Whilst the Fourier and wavelet transforms are not the topic of this paper, they do play hidden roles in the background, and so it is worth starting with a brief discussion of sparsity and signals, with how these fit in the context of traditional Fourier and wavelet analysis. The uninterested reader may safely skip this section. Sparsity (in the context of statistical inference) is the general idea that a dataset can be explained by a compact set of variables. In a signal processing context, this is clearly illustrated with the idea that a measurement that is dense in one domain may be sparse in another domain. A classical example in signal processing is a single sine tone; in the time domain, the data density is dictated by Nyquist theorem, but in the Fourier domain this could be accurately represented by a single (complex) number at the right frequency location. The Fourier transform is fundamentally limited to modelling stationary (and infinite) signals. Several ideas have developed since, starting from the idea of sliding and overlapping time windows, which led to the Gabor transform, general time-frequency representations, and eventually led to the development of the wavelet transform. The wavelet transform represents a signal as a combination of orthogonal wavelet functions, which are localised in time and scale, and so the transform effectively fits shifted and stretched versions of a "mother" wavelet to all time points in the signal. This is the continuous wavelet transform, and it provides the complete opposite of sparsity; it provides redundancy in the representation, as it can be evaluated at any arbitrary time and scale. An efficient wavelet representation emerged in the form of the Discrete Wavelet Transform (DWT), which discretises the mother wavelets into half-band filters (for both decomposition and reconstruction), that recursively split a signal into low and high frequency components, while decimating at every step. At this point, no sparsity is achieved in the decomposition yet, only a representation that is not redundant (the number of coefficients in the representation is equal to the original measured time points). However, sparsity is only a small step away, and is in fact provided by the observation that in a discrete wavelet transform, some, and sometimes the majority of the coefficients can be set to zero, providing a potentially high level of signal compression. When the coefficients are reconstructed (using the reconstruction filters of the mother wavelet), the resulting signal will only contain information encoded in those wavelet coefficients that were not switched off. This is often referred to as wavelet thresholding, and it was pioneered by Donoho [1] who also determined a simple procedure for identifying appropriate thresholds for the wavelet coefficients.

Fourier, Gabor and wavelet transforms are somewhat strict in the representations they allow. They belong to the class of *complete* bases; they do not allow a representation of greater length than the signal itself. What if the most appropriate representation of a signal came from a combination of wavelet, Fourier or other basis? This question led to the development of over-complete bases [14–16], which pose the signal representation problem as a linear combination of basis functions,

$$\mathbf{x} = \mathbf{D}\boldsymbol{\beta} \tag{4}$$

where $\mathbf{x}$ is the signal being represented (this notation will be used throughout), $\mathbf{D}$ is an $N \times K$ *dictionary* of K basis functions on N observations. $\boldsymbol{\beta}$ is a vector containing the coefficients linking the rows of $\mathbf{D}$ back to $\mathbf{x}$. The advantage that this simple formulation introduces is the flexibility of introducing any functional form, and in fact any type of fundamental signal into the columns of the dictionary $\mathbf{D}$. The reader will recognise that Eq. (4) effectively presents a linear regression problem. However, in the case of an over-complete basis, this is an ill-posed problem given that there is a much greater number of basis functions than there are observations. This is where the sparsity assumption becomes particularly useful, as it makes this ill-

posed regression problem tractable. This problem is often referred to as *sparse coding*, and various different algorithms have been suggested to solve it. Some of these are reviewed in Section 3.3.

3.2. Random matrix projection

There is particular interest in the problem of dimensionality reduction, for the purposes of algorithm design, in a wide range of scientific disciplines; this is also central to the compression step of CS. A way of "compressing" a dataset is to project the N-dimensional measurement vector $\mathbf{x}$ to a lower, M-dimensional space using a linear or nonlinear transformation. One popular approach is to use transformations, such as Principal Component Analysis (PCA), Independent Component Analysis (ICA) or factor analysis [17]. These particular examples project the measurements into spaces with certain constraints. For example, PCA is designed to rotate a data set such that the resulting vectors are forced to explain as much of the variance as possible. Such a linear transformation could be written down as,

$$\mathbf{z} = \Phi\mathbf{x} \tag{5}$$

where $\mathbf{z}$ is now a low dimensional representation of $\mathbf{x}$. An interesting projection results if the rotation matrix, Φ, is set to be a random matrix. Johnson and Lindenstrauss [13] have shown that if Φ is distributed according to a Gaussian, or Bernoulli distribution, this linear dimensionality reduction preserves, with low error, certain features of $\mathbf{x}$, such as pairwise distances. A Gaussian or Bernoulli random matrix projection also yields an orthogonal transformation. This is a central result within research in metric embedding [13]. This random matrix transform is a key ingredient in the formulation of the CS problem. In this paper, the elements of Φ have been drawn from a Gaussian distribution: $\mathcal{N}(0,1)$, and used in order to project the original measurement vector $\mathbf{x}$ into a lower dimensions, thus compressing it.

3.3. Sparse coding

The last step in a CS scheme is to reconstruct the signal, based on the compressive measurements and a dictionary [2], which is concerned with finding a sparse set of coefficients $\boldsymbol{\beta}$ that best describe the random matrix projection $\Phi\mathbf{x}$ (the compressed signal representation). What is available to the regression problem is not the full signal, but rather a projection of it through Φ. The coefficient set can be inferred if the basis dictionary is also projected through the sensing matrix to yield the following regression problem,

$$\Phi\mathbf{D}\boldsymbol{\beta} = \Phi\mathbf{x} \tag{6}$$

where, as before, Φ is a random matrix projection, $\mathbf{D}$ is a basis function set, and $\mathbf{x}$ is the (uncompressed n-dimensional) signal of interest.

The solution to Eq. (6), proposed in the original formulation of CS [3] is an l_1 regularised linear regression scheme. This type of regression was earlier proposed in [18], in the more general context of deriving sparse solutions to ill-posed problems where sparsity can be assumed. It also goes under the name of Least absolute shrinkage and selector operator (Lasso). One of the major limitations of the Lasso is that it does not give a definite answer to the appropriate level of sparsity that represents the signal. This is due to its non-probabilistic formulation.

Common to all sparse coding schemes is the need to balance the accuracy of the solution, level of sparsity and the computational complexity. An optimal solution to this problem would involve checking all possible combinations of subsets of $\boldsymbol{\beta}$ set to zero, and pick the one that provides the best representation of the signal. This is effectively a linear regression problem with a penalty term based on the l_0 norm of $\boldsymbol{\beta}$ (number of non-zero coefficients), where the following cost function is minimised,

$$\text{minimise}: \left\{\frac{1}{2N}||\mathbf{x} - \mathbf{D}\boldsymbol{\beta}||_2^2 + ||\boldsymbol{\beta}||_0\right\} \tag{7}$$

However, achieving the global minimum in (7) is a non-convex, combinatorially hard optimisation problem, so an approximation is required in practice. A solution to this was provided by the matching pursuit algorithm [14], which is a greedy iterative algorithm for finding a sparse solution to $\boldsymbol{\beta}$. Mallat originally developed MP in order to extend wavelet analysis to over-complete bases; this has already been discussed above.

While using an l_0 penalty would result in an optimum sparse representation, the Lasso tackles this problem by observing that if the penalty is relaxed to an l_1 norm, the optimisation problem becomes convex, and thus more tractable. This is an acceptable step because an l_1 penalty still encourages sparse solutions to the regression problem. The Lasso uses the following formulation of the cost function,

$$\text{minimise}: \left\{\frac{1}{2N}||\mathbf{x} - \mathbf{D}\boldsymbol{\beta}||_2^2 + \lambda||\boldsymbol{\beta}||_1\right\} \tag{8}$$

Note that the l_1 penalty is regularised by the term λ. A general l_q penalty could be computed using the sum $||\boldsymbol{\beta}||_q = \sum_{j=1}^{N}|\boldsymbol{\beta}|^q$, and the Lasso is the special case when $q = 1$. The regularisation parameter, λ, dictates the degree of sparsity in the solution. A

high value of λ encourages a low number of non-zero coefficients, and vice versa. Therefore, an appropriate value of λ needs to be chosen for each problem in particular.

The level of sparsity in the regression problem can have meaningful interpretation in physical and engineering applications. This, however, will largely depend on the type of dictionary being used within the sparse coding problem. Certain dictionaries may force interesting solutions of $\boldsymbol{\beta}$, that can have physical interpretations. For example, in the case of ultrasound signal representation, it has been shown in [12] that the use of a Hankel dictionary built using examples of pulses, solves a de-convolution problem and thus yields the impulse response function of the system. In this case, the sparsity level is related to the number of echoes received back at the transducer, so it is clearly an important quantity to estimate correctly.

The approaches developed for solving the Lasso problem focus on providing solutions across an entire regularisation path: from high to low values of λ. The result is a transparent view of how the solution changes when different levels of sparsity are assumed. Two key algorithms for finding efficient solutions across the entire regularisation path are the Least Angle Regression (LARS) [19] and one based on coordinate descent [20]. LARS solves the Lasso problem by stepping from one highly sparse solution, to another slightly less sparse solution, until all the coefficients in $\boldsymbol{\beta}$ are non-zero. It is particularly suited for large scale problems, but it is naturally greedy. Both coordinate descent and LARS offer efficient solutions over the entire regularisation path. When it comes to actually choosing an appropriate level of sparsity, Tibshirani suggests using cross-validation [18]. This approach works in practice, but it is computationally more expensive, and may be prohibitive in applications where estimation over large quantities of data are required.

Whilst the Lasso was the sparse coding scheme used originally in the development of CS, it generally suffers from the lack of a systematic and efficient way of tuning λ. Alternatives to this exist. Shortly before the development of CS, a series of algorithms were developed to tackle the problem of over-complete bases in Fourier, wavelet and other domains. Worth mentioning here are basis pursuit [16] and matching pursuit [14]. The idea in matching pursuit is to start with an empty representation, check which entry in $\mathbf{D}$ best matches the signal, $\mathbf{x}$, add that to the active representation, and iterate over this process. As such, it is greedy and prone to finding sub-optimal solutions. Basis pursuit provides a better alternative here, since it directly solves the l_1 regression problem of Eq. (8), but (unlike LARS and coordinate descent) uses global optimisation techniques to achieve this.

From the point of view of this paper, the drawback of these methods is the lack of a probabilistic interpretation; there is no quantification of uncertainty in the estimation of the parameters, and consequently in the signal reconstructions that these yield. This is the primary motivation for turning to the sparse Bayesian learning techniques developed by Tipping [6], based on RVMs. These are discussed in Section 4.

4. Sparse Bayesian learning

The sparse coding algorithms that have been discussed so far: matching pursuit, basis pursuit, Lasso, all have two major drawbacks. The first is the lack of a systematic way of dealing with uncertainty both in the measurements and in the parameters. In other words, they are not probabilistic. The second drawback is the lack of a sound methodology for tuning the sparsity parameter, without resorting to cross-validation. The framework of Bayesian inference is particularly well suited to deal with both of these problems. Its underlying idea is to derive a probability distribution over the parameters of the model, thus giving a measure of uncertainty in these estimates.

The particular flavour of Bayesian inference that will be used, as it is applicable to the problem of sparse coding, is the Relevance Vector Machine (RVM) [6]. This is a flexible model that can be applied to a wide range of regression and, therefore, more general signal representation problems. Owing to its Bayesian formulation, the key difference between the regression problem solved by the RVM and other (non-Bayesian) sparse coders is that it seeks a probabilistic solution for both $\boldsymbol{\beta}$ and $\mathbf{x}$.

4.0.1. A brief refresher on Bayesian inference

Bayes' theorem, applied to parameter estimation, takes the usual form,

$$p(\boldsymbol{\theta}|\mathbf{Y}) = \frac{p(\mathbf{Y}|\boldsymbol{\theta})p(\boldsymbol{\theta})}{p(\mathbf{Y})} \tag{9}$$

where $\boldsymbol{\theta}$ are the unknown parameters to be estimated and $\mathbf{Y}$ is the set of (multivariate) observed data. There are three probabilities on the right-hand side of Eq. (9): the prior, the likelihood and the marginal. The prior, $p(\boldsymbol{\theta})$, should represent a prior belief, about the process before it is observed. In sparse signal representation, this will encode the desire for a sparse solution. The likelihood, $p(\mathbf{Y}|\boldsymbol{\theta})$ represents the distribution of the model error, with respect to the parameters. Finally, the marginal, $p(\mathbf{Y})$, can be expanded using the sum rule of probability to yield the following integral,

$$P(\mathbf{Y}) = \int_{-\infty}^{\infty} p(\mathbf{Y}|\boldsymbol{\theta})p(\boldsymbol{\theta})d\boldsymbol{\theta} \tag{10}$$

which sums the product of the prior and likelihood (often called the marginal), over all possible parameter values $\boldsymbol{\theta}$. This is often an intractable integral, with no closed form solution available. The solution of the marginal integral often leads to either of two paths: approximations, or sampling schemes. The Laplace approximation and variational inference lie in the

approximation paths, while Gibbs sampling and Markov Chain Monte Carlo (MCMC) methods lie in the sampling path. Whilst sampling methods may be a feasible solution to many Bayesian problems, they are not practical in the case of CS on large amounts of data. The Lasso and matching pursuit algorithms are both extremely fast if compared to a sampling-based solution. One of the main reasons the authors developed an interest in the RVM for this task is not only its Bayesian formulation, but due to the existence of a practical, fast computation for the parameters [21].

4.0.2. Formulation of the RVM

The presentation of the RVM in this paper essentially follows that of Tipping [6]. The RVM solves the following regression model,

$$\mathbf{y} = \sum_{i=1}^{N} \mathbf{d}_i(\mathbf{x})\beta_i. \tag{11}$$

The reader will recognise this as a standard regression model, where as before, the weight vector is represented by $\boldsymbol{\beta} = [\beta_1, \ldots, \beta_N]$. The basis function set is represented by $\mathbf{D}(\mathbf{x}) = [\mathbf{d}_1, \ldots, \mathbf{d}_N]$. An important point to note here is that in the RVM, the basis function set is assumed to be a function of some training data $\mathbf{x}$. The RVM was originally derived as a more optimal and most importantly, sparse, alternative to the Support Vector Machines (SVM) model, which solves the same problem as Eq. (11), but defines the basis set as a kernel function,

$$\mathbf{d}_i = \kappa(x, x') \tag{12}$$

where x and x' represent two distinct points on the input space. The RVM addresses two key drawbacks of the SVM. The first is that the basis function set, $\mathbf{D}(\mathbf{x})$ does not have to satisfy Mercer's condition. Also, unlike the SVM which selects a number of columns from $\mathbf{D}(\mathbf{x})$ that scales with the number of available training data points, the RVM is designed to only select a sufficient and appropriate number of relevant vectors in $\mathbf{D}(\mathbf{x})$ that explain the observed data well, using sparsity ideas. These two key-points, underpinning the design of the RVM, are exactly what is needed in a Bayesian compressive sensing scheme: sparsity, and the ability to use arbitrary basis functions $\mathbf{D}(\mathbf{x})$ as long as they are both redundant and representative of the data. Note that the second requirement, of $\mathbf{D}(\mathbf{x})$ being representative of the data, is a key point here and hence why the dictionary has been written down as a function of $\mathbf{x}$. This is to highlight the need to somehow train the dictionary against a representative set. Whilst this notation will be dropped in the rest of the discussion, the reader should remember this point.

For the benefit of the reader, a condensed summary of the RVM model is provided below, which simply follows the requirements of Bayesian regression set out in Section 4.0.1 above. A more complete version can be found in [6].

The observations of the model are assumed to be corrupted with noise, and this is modelled by a target vector, $\mathbf{t}$,

$$\mathbf{t} = \mathbf{y} + \varepsilon \tag{13}$$

where ε is the noise term and $\mathbf{y}$ is the representation of the signal, as defined by Eq. (11). If the noise term is assumed to be Gaussian distributed $\varepsilon \sim \mathcal{N}(0, \sigma^2)$, the likelihood function, $p(\mathbf{t}|\boldsymbol{\beta}, \sigma^2)$, can be written down as,

$$p(\mathbf{t}|\boldsymbol{\beta}, \sigma^2) = (2\pi)^{-N/2}\sigma^{-N}\exp\left\{-\frac{||\mathbf{t}-\mathbf{y}||_2^2}{2\sigma^2}\right\} \tag{14}$$

The key ingredient, however, is the form of the prior distribution of the parameter vector, $p(\boldsymbol{\beta}|\boldsymbol{\alpha})$ (where $\boldsymbol{\alpha}$ is a hyperparameter), as it encodes one's prior belief about the form of the coefficients. Most importantly, it is through the form of this prior that a sparse solution to the regression problem can be enforced. The RVM enforces sparsity through the use of a hierarchical Gaussian prior. This is a conjugate prior to a Gaussian distribution and thus yields algebraic forms that are tractable when multiplied by the likelihood function of Eq. (14). The form of the prior is,

$$p(\boldsymbol{\beta}|\boldsymbol{\alpha}) = \prod_{i=1}^{M}\mathcal{N}(\beta_i|0, \alpha_i^{-1}). \tag{15}$$

The term $\boldsymbol{\alpha}$ is a *hyperparameter* vector, that defines the variance of the prior distribution of the parameters. It is formally defined as $\boldsymbol{\alpha} = \{\alpha_1, \ldots, \alpha_n\}$ (where n is the number of coefficients in $\boldsymbol{\beta}$). This prior is hierarchical since a hyper-prior over the hyperparameters also needs to be defined. This includes both the variance terms for the prior, $\boldsymbol{\alpha}$, as well as the signal noise variance σ^2. Instead of setting a prior over the variance directly, a prior is set over its inverse $\rho = \sigma^{-2}$:

$$p(\boldsymbol{\alpha}) = \prod_{i=1}^{M}\Gamma(a)^{-1}b^a\alpha^{a-1}e^{-b\alpha} \tag{16}$$

$$p(\rho) = \Gamma(c)^{-1} d^c \alpha^{c-1} e^{-d\rho} \tag{17}$$

where Γ is the Gamma function and a, b and c, d are hyperparameters of the prior and noise variance respectively (Section 4.3 provides a small discussion on these). With the likelihood function of (14) and the prior of (15), the posterior distribution over the parameters can be written using Bayes' theorem as,

$$p(\boldsymbol{\beta}|\mathbf{t}, \boldsymbol{\alpha}, \sigma^2) = \frac{p(\mathbf{t}|\boldsymbol{\beta}, \sigma^2) p(\boldsymbol{\beta}|\boldsymbol{\alpha})}{p(\mathbf{t}|\boldsymbol{\alpha}, \sigma^2)}. \tag{18}$$

Using standard Gaussian identities, this yields a Gaussian distribution,

$$p(\boldsymbol{\beta}|\mathbf{t}, \boldsymbol{\alpha}, \sigma^2) = \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma}) \tag{19}$$

where the mean and variance are given by,

$$\boldsymbol{\Sigma} = (\mathbf{A} + \sigma^{-2}\mathbf{D}^\top \mathbf{D})^{-1} \tag{20}$$

$$\boldsymbol{\mu} = \sigma^{-2} \boldsymbol{\Sigma} \mathbf{D}^\top \mathbf{t} \tag{21}$$

$\mathbf{A}$ is a diagonal matrix with the elements of $\boldsymbol{\alpha}$ along its diagonal. Eqs. (21) and (20) define the mean and covariance of the coefficient vector $\boldsymbol{\beta}$.

In order to make predictions with this model, one would wish to evaluate the distribution $p(\mathbf{t}_\star|\mathbf{t}, \boldsymbol{\alpha}, \sigma^2)$(where $\mathbf{t}_\star$ is a set of testing data points), which can be shown to be a multivariate Gaussian with mean and covariance [6],

$$\mathbf{y}_\star = \mathbf{D}\boldsymbol{\mu} \tag{22}$$

$$\mathbf{V}_\star = \sigma^2 + \mathbf{D}^\top \boldsymbol{\Sigma} \mathbf{D} \tag{23}$$

The predictive mean, defined by Eq. (22), is an intuitive application of the linear transformation that defines the sparse coding problem through dictionary representation, defined earlier in Eq. (4). The predictive variance in Eq. (23) is the sum of two terms: the signal noise, σ^2 and the predictive uncertainty, arising from the term $\mathbf{D}^\top \boldsymbol{\Sigma} \mathbf{D}$. It is clearly important to derive an accurate estimate of the signal noise term as it can effectively dictate how much of the uncertainty in the prediction is governed by measurement noise, and how much of it is explained by actual predictive uncertainty against the given dictionary. This is effectively a problem of optimising the hyperparameters. For example, a trivial case would be to assume a very high level of measurement noise, σ^2. In this setting, the error term, given by the likelihood function of Eq. (14) would render bad predictions as being within the acceptable range.

So far, the key equations of the RVM have been laid out, leading to Eqs. (22) and (23) which allow one to make a prediction over the mean and variance of a sparse coding problem. This leads directly to their application within a CS scheme. In the sparse coding step, two aspects are missing so far, and are given in the following two subsections: the optimisation of the hyperparameter terms σ^2 and $\boldsymbol{\alpha}$, and the formulation of the CS problem in terms of the RVM.

4.1. Hyperparameter optimisation

There are two hyperparameters that have a strong influence over the predictive distribution: $\boldsymbol{\alpha}$ and σ^2. A completely Bayesian approach would be to derive full posterior distributions over these, but this is not practical given that the integral resulting in the formulation of $p(\boldsymbol{\beta}, \boldsymbol{\alpha}, \sigma^2|\mathbf{t})$ is intractable. The approach taken in [6] to solve this is to relax the requirement of solving for a full distribution over $\{\boldsymbol{\alpha}, \sigma^2\}$, and instead optimise a point estimate. This approach is often called *type-II maximum likelihood estimation*, as one optimises the likelihood of the hyperparameters with respect to the data.

The problem is formulated as the maximisation of a (log) likelihood with respect to the parameters, and is given as

$$\mathcal{L}(\boldsymbol{\alpha}) = \log p(\mathbf{t}|\boldsymbol{\alpha}, \sigma^2) = \log \int_{-\infty}^{\infty} p(\mathbf{t}|\boldsymbol{\beta}, \sigma^2) p(\boldsymbol{\beta}|\boldsymbol{\alpha}) d\boldsymbol{\beta} \tag{24}$$

$$= -\frac{1}{2}[N \log 2\pi + |\mathbf{C}| + \mathbf{t}^\top \mathbf{C}^{-1} \mathbf{t}] \tag{25}$$

The $\mathbf{C}$ matrix in Eq. (24) represents the covariance function of the conditional distribution $p(\mathbf{t}|\boldsymbol{\alpha}, \sigma^2)$, and is defined as

$$\mathbf{C} = \sigma^2 \mathbf{I} + \mathbf{D}\mathbf{A}^{-1}\mathbf{D}^\top \tag{26}$$

The original RVM paper presents a procedure for finding the optimal $\boldsymbol{\alpha}$ using this log likelihood based on the Expectation Maximisation (EM) algorithm. EM is an iterative algorithm that maximises the likelihood of the parameters in the presence of missing or latent variables. The algorithm (in [22]) is rather general and allows for formulation of a large class of optimisation problems as iterative steps between evaluations of the expectation over the hidden variables, and updates to the model parameters that guarantee an increase in the likelihood function at every step. While EM suffers from the lack of a guarantee of a global optimum solution, this can be often alleviated in practice through a good, or informed, choice of initial parameters. In the case of the RVM, however, the major drawback of the approach is that the matrix inversion required for $\mathbf{C}$ in Eq. (25) has a computational complexity of $\mathcal{O}(N^3)$ (where recall N is the number of measurement points). Because the

application of CS effectively reduces the value of N, this implies a significant improvement in the computational complexity during hyperparameter optimisation (as compared to a sparse coding without compression), where repeated inversion of $\mathbf{C}$ is required. On the other hand, evaluation of the parameter covariance matrix, in Eq. (20) involves an inversion where the computational complexity scales with $\mathcal{O}(K^3)$, where K is the number of basis functions, or columns of $\mathbf{D}$. Usually, optimisation of the hyperparameters will involve evaluation of both of these quantities.

The requirement that CS places on the over-completeness of $\mathbf{D}$ makes this computational burden even worse, as it implies that the better the dictionary gets at representing a broad class of signals, the computational burden will increase in a cubic fashion. This places a limit on the number of basis vectors that are practical to use in $\mathbf{D}$, in the case of sparse Bayesian learning, and this goes against the requirement for over-completeness.

The EM algorithm described in [6], includes a pruning step on every iteration, where any vector $\mathbf{d}_i$ in $\mathbf{D}$ that is deemed to be "irrelevant" is removed from the set, so the effective number of columns of $\mathbf{C}$ is reduced. This pruning technique works well, but it is still a top-down approach, where the first few iterations of the algorithm will still be computationally expensive. Tipping himself devised a bottom-up approach in [21] where an iterative "fast" maximum likelihood estimation is provided for this class of sparse Bayesian models.

A fast approach to this marginal likelihood optimisation was developed in [21], and this is the approach adopted here for the hyperparameter optimisation. The idea is to start with a single basis (column) vector, $\mathbf{d}_i$ from $\mathbf{D}$ and to iteratively add and/or remove basis functions from the set of columns of $\mathbf{D}$. Some relevant criteria are required to do so in a principled sense, and in particular one that maximises $\mathcal{L}(\boldsymbol{\alpha})$. Tipping achieves this by decomposing $\mathbf{C}$ (since this is the quantity of interest) into two parts, so that,

$$\mathbf{C} = \mathbf{C}_{-i} + \alpha_i^{-1}\mathbf{d}_i^{\top}\mathbf{d}_i. \tag{27}$$

Here, $\mathbf{C}_{-i}$ denotes the covariance matrix $\mathbf{C}$ without the contribution of the i_{th} basis vector, $\mathbf{d}_i$. In this form, the inverse and determinant of the covariance can be written in the following convenient form,

$$|\mathbf{C}| = |\mathbf{C}_{-i}^{-1}||1 + \alpha_i^{-1}\mathbf{d}_i^{\top}\mathbf{C}_{-i}^{-1}\mathbf{d}_i| \tag{28}$$

$$\mathbf{C}^{-1} = \mathbf{C}_{-i}^{-1} - \frac{\mathbf{C}_{-i}^{-1}\mathbf{d}_i^{\top}\mathbf{d}_i\mathbf{C}_{-i}^{-1}}{\alpha_i + \mathbf{d}_i^{\top}\mathbf{C}_{-i}^{-1}\mathbf{d}_i} \tag{29}$$

This is helpful because it allows writing the likelihood function as the sum of the contribution of $\mathbf{d_i}$ and the set $\mathbf{D}_{-i}$ that excludes $\mathbf{d}_i$ as [21],

$$\mathcal{L}(\boldsymbol{\alpha}) = \mathcal{L}(\boldsymbol{\alpha}_{-i}) + l(\alpha_i) \tag{30}$$

$$= \mathcal{L}(\boldsymbol{\alpha}_{-i}) + \frac{1}{2}\left[\log\alpha_i - \log(\alpha_i + s_i) + \frac{q_i^2}{\alpha_i + s_i}\right] \tag{31}$$

where for simplification of the above expression

$$s_i = \mathbf{d}_i^{\top}\mathbf{C}_{-i}^{-1}\mathbf{d}_i \tag{32}$$

$$q_i = \mathbf{d}_i^{\top}\mathbf{C}_{-i}^{-1}\mathbf{t}. \tag{33}$$

These two terms are referred to in [21] as sparsity and quality factors respectively. The sparsity factor can be seen as a measure of how much $\mathbf{d}_i$ overlaps with the basis vectors already in the current model. The quality factor, q_i, can be interpreted as a measure of the discrepancy that $\mathbf{d}_i$ introduces to the error of the model exclusive of $\mathbf{d}_i$.

It is shown in [23] based on an analysis of $l(\alpha_i)$ that $\mathcal{L}(\boldsymbol{\alpha})$ can be maximised with respect to α_i, based on the following conditions,

$$\alpha_i = \frac{s_i^2}{q_i^2 - s_i} \qquad \text{if } q_i^2 > s_i, \tag{34}$$

$$\alpha_i = \infty \qquad \text{if } q_i^2 \leqslant s_i \tag{35}$$

This is an incredibly useful observation, from the point of view of sparsity and optimisation. Recall that α_i represents the inverse variance of the prior distribution over the model weights $p(\boldsymbol{\beta}|\boldsymbol{\alpha})$. A value of $\alpha_i = \infty$ implies that the weight for basis $\mathbf{d}_i$ is infinitely peaked around zero, thus deeming it irrelevant. The iterative procedure for fast optimisation of $\mathcal{L}(\boldsymbol{\alpha})$ described in [21] therefore uses these observations in order to derive two important learning rules. For this, an "active set" $\mathbf{R}$ is defined, which contains the set of vector deemed relevant. The two basic rules are:

- If d_i not in active set $\mathbf{R}$, and $q_i^2 > s_i$, add $\mathbf{d}_i$ to $\mathbf{R}$.
- If d_i already in $\mathbf{R}$ and $q_i^2 \leqslant s_i$, remove $\mathbf{d}_i$ from $\mathbf{R}$.

The procedure is slightly more complicated than this, but based around these two learning rules. More details can be found in [21]. Note that unlike other pursuit methods such as matching or basis pursuit [14,16], the ability of not only adding but deleting basis vectors makes the greediness of the search for relevant vectors somewhat more optimal. In practice one could

prioritise the deletion of basis vectors, yielding a more greedy algorithm that is computationally faster. On the other hand, favouring the addition of basis vectors would yield an algorithm that is not greedy at all in the limit of the number of function evaluations, though this may not be computationally practical. In practice, the authors have found that accepting some level of greediness by adopting an algorithm that favours subtraction works well in practice, provided one has a reasonably good dictionary; this is further discussed in Section 5.

4.2. CS formulation with the RVM

The key ingredients in the formulation of a Bayesian CS scheme have been laid out and discussed, namely random transformations and sparse coding, together with an efficient Bayesian formulation for sparse coding based on the RVM. This section formally defines the procedure required for reconstructing a randomly compressed signal with an RVM and provides some discussion over the issues one may encounter while applying this in practice.

So far, the RVM has been discussed in the general case of regression. Eq. (11) describes an input-output relationship between $\mathbf{x}$ and $\mathbf{y}$. In the CS case, one wishes to recover the underlying true signal based on an incomplete, or compressed measurement which has been acquired through a random transformation of the original "true" signal, as described by Eq. (5). With this in mind, the output of the basic RVM model, $\mathbf{y}$ in Eq. (11), can be set to the randomly transformed signal $\mathbf{y} = \boldsymbol{\Phi}\mathbf{x}$, to give

$$\boldsymbol{\Phi}\mathbf{x} = \sum_{i=1}^{N} \boldsymbol{\phi}_i \mathbf{d}_i(\mathbf{x})\beta_i \tag{36}$$

where as before, the random transformation is given by $\boldsymbol{\Phi} = [\boldsymbol{\phi}_1, \ldots, \boldsymbol{\phi}_M]$. Using the product $\boldsymbol{\Phi}\mathbf{D}$ as a dictionary, a solution for $p(\boldsymbol{\beta}|\mathbf{t}, \boldsymbol{\alpha}, \sigma^2)$ can be found using the fast marginal likelihood optimisation procedure described in Section 4.1. Eqs. (21) and (20) can be used to derive the mean and variance over the weights $\boldsymbol{\beta}$, while Eqs. (22) and (23) can be used to reconstruct the new signal.

There is an interesting point to be made regarding the evaluation of the resulting uncertainty over $\boldsymbol{\beta}$ and ultimately over the predictions. When reconstructing a signal based on the compressed or incomplete measurements $\boldsymbol{\Phi}\mathbf{x}$, it is the product $\boldsymbol{\Phi}\mathbf{D}$ that is passed as a dictionary to the sparse coder. In the RVM, the form of the covariance for the model weights $\boldsymbol{\beta}$ becomes, from Eq. (20),

$$\boldsymbol{\Sigma} = (\mathbf{A} + \sigma^{-2}(\boldsymbol{\Phi}\mathbf{D})^{\top}(\boldsymbol{\Phi}\mathbf{D}))^{-1} \tag{37}$$

Considering that $\mathbf{A}$ and σ^2 are hyperparameters, it can be safe to say a priori, that the uncertainty in $\boldsymbol{\beta}$ depends entirely on $\boldsymbol{\Phi}\mathbf{D}$. The random transformation $\boldsymbol{\Phi}$ maps $\mathbf{D}$ from an N-dimensional "complete" space to an M-dimensional "compressed" space. Based on the Johnson-Lindenstrauss lemma, the transformation $\boldsymbol{\Phi}$ preserves pair-wise distances as long as M and N stay within certain bounds. In other words, as long as M is not too low (not too much compression), then the information in $\mathbf{D}$ will be well preserved in $\boldsymbol{\Phi}\mathbf{D}$.

This process is visualised in Fig. 2, where the map from $\mathbf{D}$ to $\boldsymbol{\Phi}\mathbf{D}$ to $\boldsymbol{\Phi}\mathbf{R}$ is illustrated. The particular dictionary used for that visualisation is a (truncated) k-means dictionary of ultrasound pulses, and $\boldsymbol{\Phi}$ is a Gaussian random matrix. The visualisation shows that the random transform effectively shrinks the rows of $\mathbf{D}$. Because $\boldsymbol{\Phi}\mathbf{D}$ is not square any more, it is its pseudo-

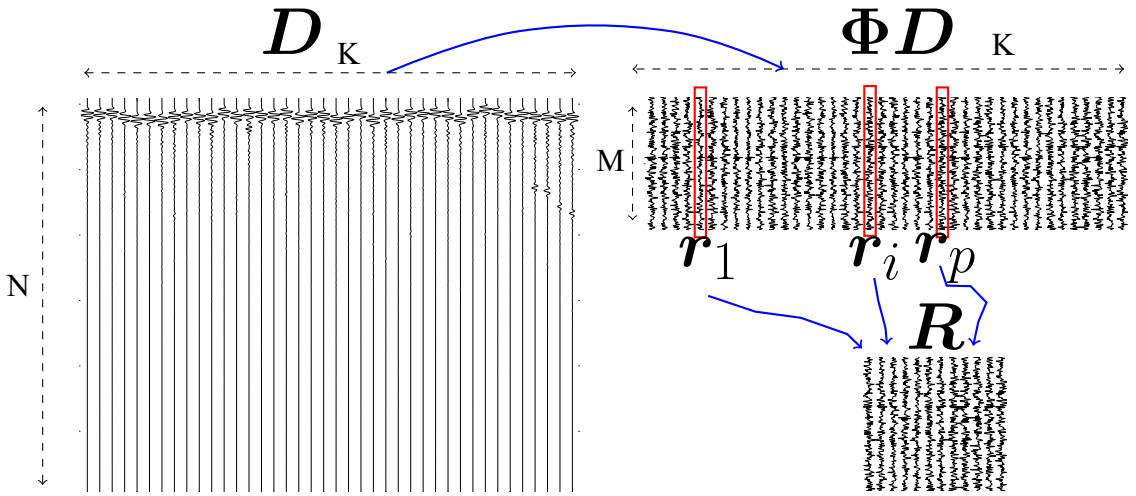

Fig. 2. Illustration of application of random transform to a dictionary and the selection of relevant vectors resulting from that transformation.

inverse that needs to be invertible: exactly the product in Eq. (37). As the compression level grows (low M) so does the "aspect ratio" of $\mathbf{\Phi D}$. If all the columns of $\mathbf{D}$ were to be kept, this matrix would be ill-conditioned at relatively high M. This is where keeping an active set, $\mathbf{R}$, becomes useful. By choosing an active set of columns from the (transformed) dictionary, $\mathbf{\Phi R}$, the RVM is basically keeping the problem invertible.

There will be a dimension below which Eq. (37) will not be suitable for inversion. However, the number of columns of $\mathbf{R}$ will depend on the level of sparsity of the problem; on how many relevant vectors are chosen during the hyperparameter optimisation step. Therefore, the effective level of compression will be limited by inherent level of sparsity in the signal, and how well this is represented in the dictionary. If the specific signal reconstruction problem requires a high number of columns of $\mathbf{\Phi D}$, but there are very few rows (high compression), then only the most relevant columns will be chosen that keep $\mathbf{\Phi R}$ well-conditioned (close to square). In general, if $\mathbf{D}$ represents the data well, choosing more columns will lead to reconstructions with lower error, and vice versa. Therefore, this act of choosing an active set that keeps the problem well conditioned can be seen as adjusting the accuracy of the solution to the level of compression. It will be shown in Section 6, and in particular in Fig. 8 that higher levels of compression results in the selection of fewer relevant vectors.

Under very high levels of compression, the inversion in (37) will not be well defined and $\mathbf{\Sigma}$ will collapse. This will be illustrated in the poor predictions of the confidence interval on the top row of Fig. 7.

4.3. Relationship between Lasso and RVM

The RVM and the Lasso could be seen as different solutions to the same underlying Bayesian linear regression problem. The RVM can be summarised as a Bayesian linear regression model that uses a hierarchical Gaussian prior in order to enforce sparsity. It is not entirely obvious how this is the case and so this will be discussed here in more detail. Sparsity is induced through the choice of a prior distribution that has most of its probability mass centred around zero. This implies that in the absence of strong evidence to the contrary (in the form of a likelihood), the weight β_i will be zero, and so the corresponding column in $\mathbf{D}$ would be deemed "irrelevant".

The prior for the RVM is defined by Eq. (15). On its own this prior, conditioned on the hyperparameters $p(\boldsymbol{\beta}|\boldsymbol{\alpha})$, is Gaussian and does not, as such, encourage any sparsity. To see how sparsity is introduced, one has to marginalise out the weights $p(\beta_i)$, which leads to,

$$p(\beta_i) = \int p(\beta_i|\alpha_i)p(\alpha_i)d\alpha_i \tag{38}$$

where $p(\alpha_i)$ is defined from Eq. (16) as a Gamma distribution. It is a fairly standard result that this integral over the product of a Gaussian and a Gamma distribution yields a Student's t distribution. This is a key point here, because a Student's t distribution with low degrees of freedom places most of its probability mass around the centre and is thus a useful sparsity-inducing prior. Recall the two sets of parameters of the Gamma distributions that describe the priors $p(\boldsymbol{\alpha})$ and $p(\rho)$ in Eqs. (16) and (17) respectively (a, b, c and d). These control the degrees of freedom of the Student's t distribution resulting from the integral in Eq. (38). In the case of $p(\alpha_i)$, setting $a = b = 0$ results in $p(\alpha_i) \propto 1/|\beta_i|$, which is a prior that is sharply peaked around zero. With low values of a and b the resulting Student's t distribution will still be sharply peaked around zero, but increasing this significantly beyond one will result in the Student's t loosing its concentration of mass around the centre. In the limit of infinite values of the parameters, the Student's t becomes a Gaussian distribution, and thus looses its sparsity-inducing characteristics. For sparsity to be induced, a and b should be kept to as close to zero as machine precision allows.

In the case of c and d which control the (inverse) variance prior $p(\rho)$, their values should be kept to a small number so that its posterior is dominated by the data rather than the prior. This is common practice when doing Bayesian modelling of Gaussian distribution parameters [24].

On the other hand, the Lasso is the result of a Bayesian linear regression formulation with a Laplace prior [25]. The mode of this formulation yields the l_1-penalised linear regression optimisation given in (8). The issue with the Laplace prior is that the resulting marginal likelihood is still intractable and so sampling or other approximation methods are required in order to derive the full posterior. In contrast with the Student's t prior, the Laplace distribution sets a prior $p(\beta_i) \propto \exp(-|\beta_i|)$. Some papers have been published concerning the problem of the Bayesian Lasso [26], or even Bayesian compressive sensing [27], which take the Laplace prior approach, and normally involve some form of sampling in order to get the full posterior over $\boldsymbol{\beta}$.

Fundamentally, the difference between the original compressive sensing formulation in terms of Lasso regression, and the one given by the RVM is the choice of prior distributions. Fig. 3 illustrates this difference, comparing a Laplace, a Student's t and a Gaussian distribution. All three distributions in this comparison were generated so as to have the same variance. It is clear that, compared with a Gaussian, both the Laplace and Student's t place most of their probability mass around the centre. The Student's t distribution is, however, much smoother than the Laplace distribution.

The application of the RVM as a sparse coding step in a CS setting has previously been carried out in [28], in an image processing context, although no application or discussion over the benefits or issues of the probabilistic interpretation is given.

The contribution the authors attempt to make with this paper is a thorough treatment of the probabilistic interpretation of this CS framework, which involves a discussion of the signal reconstruction uncertainty estimates given by the RVM. The reader will have by now noticed that the quantification of uncertainty depends on the given dictionary, so a discussion of how different dictionary choices affect the signal reconstruction is imperative.

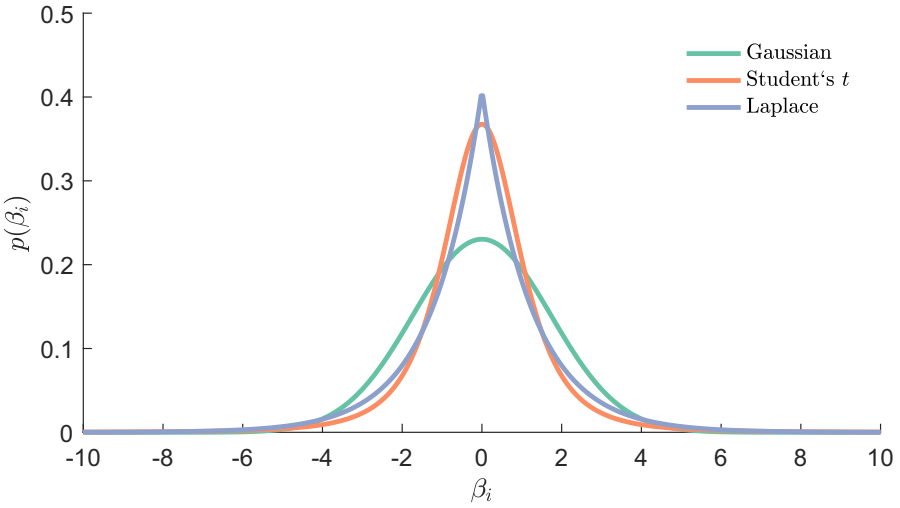

Fig. 3. Illustration of Student's t and Laplace sparsity-inducing priors, against a Gaussian distribution. All three distributions are shown with a variance of three.

5. Dictionary learning

It should be clear by now that the choice of dictionary used in any CS scheme predicates the quality of the signal estimates. Broadly speaking, there are two strategies for dictionaries: data-based and model-based[1]. The field of sparse signal representation started, in fact, from the idea of extending traditional model-based dictionaries (such as Fourier and wavelets), to non-orthogonal, over-complete versions of these [14,16]. The idea of a model-based dictionary, however, is not restricted to signal models and can be generalised to physical models as well. In the context of ultrasound signal representation, for example, Lamb wave propagation models have been used to assemble dictionaries in a CS setting [29]. The advantage of defining a dictionary based on a physical, or signal model is that it only requires some broad prior assumptions about the what the signal will "look like", without the need for measurements to be taken a priori. This does mean that for the signal reconstruction to be successful, the assumed functional forms present in $\mathbf{D}$ must be roughly correct, and this is where model-based dictionaries may perform poorly.

The relevant model-based dictionary for the ultrasound examples given here is the windowed tone burst already discussed in Section 2.2.

If the measured signals are complex and their functional form cannot be easily summarised by simplistic mathematical formulae, creating dictionaries based on available examples of what the data might look like may be a much better strategy. This led to the development of *dictionary learning* strategies shortly after the ideas of sparse coding became available [30,31]. The use of dictionaries has already been discussed at length in the context of sparse coding for compressive sensing. However, this section focusses on the specific issue of the impact that the dictionary has on the Bayesian interpretation of CS. The dictionary lies at the centre of the probability computations. In fact, it could be interpreted as defining the covariance structure of the data. Recall that the covariance of a data set $\mathbf{X}$ can computed using the matrix product,[2]

$$cov(\mathbf{X}) = \frac{1}{N}\mathbf{X}\mathbf{X}^{\top} \tag{39}$$

where N is the number of observations. From this, it is easy to see that in order to make reasonable predictions of the uncertainty of the recovered signals, the structure of the product $\mathbf{D}\mathbf{D}^{\top}$ should resemble that of $\mathbf{X}_t\mathbf{X}_t^{\top}$ (where the subscript t denotes the training set). A trivial way to assemble a dictionary would be to simply set $\mathbf{D} = \mathbf{X}_t$ as this would perfectly capture the covariance of the training data set. It is trivial because it does not summarise or decompose the structure of the data in any way. Using the training data matrix as a dictionary is also not a scalable solution since, as has already been discussed in Section 4, even though the RVM scales well with low-dimensional representation, the inversion required in Eq. (20), to compute the posterior covariance, scales with order $\mathcal{O}(N^3)$.

Fig. 4 illustrates this trivial dictionary for a training set assembled taking 2500 ultrasound pulses at random from the C-scan data (representing a small subset of the entire data-set in this case). This will be used as an illustrative point of comparison against other dictionary types. The reader can refer back to Section 1 in order to read Fig. 1 for an example of a single ultrasound pulse, noting that (typically) there are two main reflections, from the front and a back wall. The front

[1] The term "model-based" is used here to denote a dictionary assembled using either a mathematical or physical model

[2] Assuming $\mathbf{X}$ has been centred, so that it has zero-mean

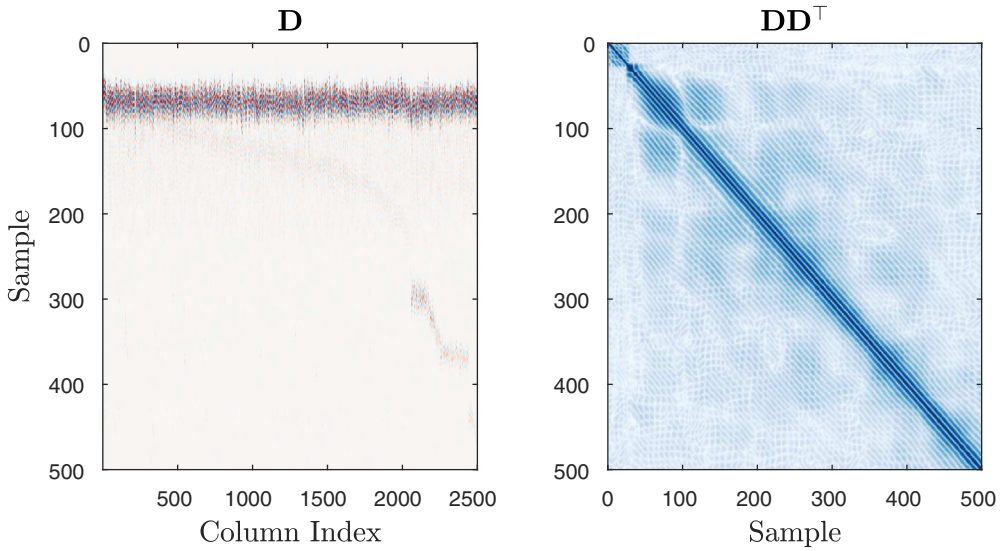

Fig. 4. Left: Training data set, $\mathbf{X}_t$, sorted by increasing time-of-flight index. Right: Data covariance, (square rooted and scaled by diagonal variance).

wall reflection arrives almost always at the same time, and with high energy, whilst the back wall reflection changes its arrival time depending on the local material characteristics, and is significantly attenuated. The dictionary entries (columns) in Fig. 4 have been sorted in ascending order of their time-of-flight (TOF) index. This is to make the visualisation of the data matrix interpretable; the columns on the left represent low TOF pulses, whilst the columns on the right represent higher TOF indices. The result is a subtle shift of the second (back-wall) pulse as the index increases. On the right, Fig. 4 shows a visualisation of the covariance structure of the dictionary. This simply shows the correlation structure between the different sample indices of the signal.

Back to the dictionary learning discussion, there are two general ways in which one can force non-trivial summaries of the data: through projection or clustering. The following two sections provide an overview of dictionary learning under these two interpretations. However, the discussion will be focused around the problem of estimating good uncertainty bounds, so the respective discussions will not go into a great amount of detail.

5.1. Projection interpretation

The early developments of dictionary learning [30–32] built on the idea of using projections of the data, inspired by models such as Principal Component Analysis (PCA) and Independent Component Analysis (ICA). The projection interpretation to dictionary learning uses the following generative model formulation,

$$\mathbf{x} = \mathbf{D}\boldsymbol{\beta} + \eta. \tag{40}$$

In this interpretation, $\mathbf{D}$ is the projection matrix, $\boldsymbol{\beta}$ is a latent, generative variable and η is the noise process. PCA learns an interpretation of the model in Eq. (40) that forces the dictionary to be the set of eigenvectors of the data covariance matrix. PCA achieves this through the assumption of a Gaussian, isotropic noise process, η, which in turn results in $\mathbf{D}$ being orthogonal. ICA overcomes this problem by effectively allowing the modelling of non-Gaussian noise, and a non-orthogonal representation of $\mathbf{D}$. However, both PCA and ICA are in breach of the condition of over-completeness required in CS.

The solution is then clearly to formulate a model that allows $\mathbf{D}$ to be over-complete. To the author's knowledge, this was first done in [30,31], by formulating the projection learning as a maximum likelihood problem with a key ingredient: enforcing sparsity in $\boldsymbol{\beta}$ by placing a Laplace prior $p(\boldsymbol{\beta})$ inside an iterative EM algorithm. This effectively adds sparse coding to the learning step, thus forcing over-complete solutions to $\mathbf{D}$. Other projection-based dictionary learning strategies follow similar directions. A recent and fairly complete review of dictionary learning that covers the relevant developments up to 2011 can be found in [33].

One of the most general and complete views of dictionary learning formulated so far is the online matrix factorisation algorithm presented in [34]. A wide variety of projection algorithms can be cast as a matrix factorisation problem. A popular example would be the formulation of PCA through a Singular Value Decomposition (SVD). The ideas presented in [34] are shown to be general in the sense that minor modifications of the baseline online dictionary learning algorithm can lead to other well-known models such as sparse PCA and Non-negative Matrix Factorisation (NMF). However, the most important point in [34] is that learning is formulated as an online, or mini-batch problem. This is an important computational aspect; if the learning is sequential, only small batches of data need to be loaded into computer memory at any given time. This in turn

means that learning scales gracefully to arbitrarily large quantities of data, which is crucial in applications where loading the relevant training data into memory would not only be slow, but infeasible.

The following is a summary outline of the dictionary learning algorithm of Mairal et al. [34]. Dictionary learning algorithms use the idea of optimising $\mathbf{D}$ against some empirical cost function that depends on both the data and the dictionary, $l(\mathbf{x}, \mathbf{D})$ [31,30],

$$f(\mathbf{D}) = \sum_{i=1}^{N} l(\mathbf{x}_i, \mathbf{D}). \tag{41}$$

Here, it makes sense to define $l(\mathbf{x}, \mathbf{D})$ as the l_1 regularised cost function to the sparse coding problem [35], which has already been discussed in 3.3,

$$l(\mathbf{x}, \mathbf{D}) \triangleq ||\mathbf{x} - \mathbf{D}\boldsymbol{\beta}||_2^2 + \lambda||\boldsymbol{\beta}||_1. \tag{42}$$

Doing this allows $\mathbf{D}$ to be over-complete, but it also means that the columns $\mathbf{d}_i$, of $\mathbf{D}$ can grow to arbitrarily large values, leading to very small values of $\boldsymbol{\beta}$. To alleviate this, the optimisation of the cost function is defined with an extra constraint, $\mathcal{C}$, on the l_2 norm of each $\mathbf{d}_i$. This allows writing down the dictionary learning as a matrix factorisation problem,

$$\text{minimise} : \frac{1}{2}||\mathbf{X} - \mathbf{D}\boldsymbol{\beta}||_2^2 + \lambda||\boldsymbol{\beta}||_1 \qquad \text{s.t}\mathbf{D} \in \mathcal{C}. \tag{43}$$

Mairal et al. arrive at an efficient iterative sequential algorithm for this optimisation using several techniques. One is the observation that minimising the expected cost, $\mathbb{E}_{\mathbf{x}}[l(\mathbf{x}, \mathbf{D})]$, provides computationally more efficient solutions, as stochastic gradient optimisation schemes have high convergence rates against this expected cost [36].

The actual optimisation used is a sequential stochastic approximation that minimises a quadratic local surrogate of the expected cost. The algorithm thus splits into two steps:

1. Estimate $\boldsymbol{\beta}$ using $\mathbf{D}_{t-1}$ and sparse coding (where $\mathbf{D}_{t-1}$ is the previous estimate of $\mathbf{D}$ during the sequential updates).
2. Update $\mathbf{D}_t$ using $\boldsymbol{\beta}$.

In this case, one assumes that observations of $\mathbf{x}_t$ are given sequentially at discrete time indices t. Also, $\mathbf{D}_0$ can be initialised in a number of ways. However, if one assumes that there truly is no prior information about the process and the data will in fact be presented to the algorithm in an online fashion, initialisation via a random matrix is a good choice for two reasons:

1. No prior information needs to be assumed.
2. Columns that remain as random vectors effectively absorb no information and can be pruned from the final dictionary.

Some readers may also notice that this update scheme is akin to the Expectation–Maximisation updates in an EM algorithm. In fact, there is an online generalisation of the EM algorithms that presents an alternative, but similar formulation to the same problem [37]. The key difference though is that these updates explicitly enforce a sparse solution in the "expectation" step.

This algorithm can be enhanced to represent other classes of models, but for the purposes of this paper, which is to discuss its applicability within a Bayesian CS framework, only the basic online dictionary algorithm has been used. Section 5.3 discusses some practical aspects and results of applying this within a CS framework.

5.2. Clustering interpretation

Clustering is a particular form of unsupervised learning that seeks to summarise the density of a multivariate data-set by finding groups with certain similarities within a training set. One of the most popular clustering algorithms is k-means clustering, owing to its simplicity, yet sound theoretical foundations; k-means can be seen as a special case of Gaussian mixture modelling, which in turn means that there is an efficient EM formulation for a learning algorithm with certain guarantees (an increase in the likelihood of the parameters at every iteration of the algorithm).

The classical, text-book formulation of k-means clustering [38] seeks to find groups within a multivariate data set $\mathbf{X}$ that minimise the Euclidean distance between every cluster centre $\boldsymbol{\mu}_k$ and the (multivariate) data point $\mathbf{x}_i$ that belongs to K. Clustering is referred to as the task of finding an appropriate set of cluster centres that minimise an objective function. The task of assigning a cluster class to a data point is often referred to as *vector quantisation*, and is defined as,

$$\min_j : ||\mathbf{x}_i - \boldsymbol{\mu}_j||_2^2 \tag{44}$$

Vector quantisation seeks to represent an observation $\mathbf{x}$ using the closest cluster available to it. In this sense, it is clearly a sparse coder, albeit an extreme one. The problem can even be formulated in the (now) familiar form,

$$\mathbf{x} = \mathbf{D}\boldsymbol{\beta} + \eta \tag{45}$$

where now the columns of **D** contain the cluster centres, so that for notational convenience these are redefined here as $\mathbf{d}_j = \boldsymbol{\mu}_j$ for $j = [1, \ldots, K]$. The coefficient vector $\boldsymbol{\beta}$ is only allowed one non-zero entry, which is found using Eq. (44).

It is always interesting to relate two seemingly dissimilar models as special cases of a single general model. Such is the case between projection and clustering, as pointed out by Roweis and Ghahramani [39], who explain a wide range of projection, clustering, filtering and smoothing problems under the general framework of linear Gaussian models. Based on this, the obvious similarity between the projection and cluster models should not come as a strong surprise.

Similar observations have also been made regarding the similarity between the clustering and sparse coding problems [32,40]. The specific value of k-means clustering as a dictionary learning strategy was recognised some years after the development of the projection interpretations [41,42]. In particular, Aharon [42] formulated an iterative scheme that alternates between a sparse coding step to solve for $\boldsymbol{\beta}$ (such as matching pursuit or l_1 regularised regression) and an update step of **D** based on SVD. The method is thus called k-SVD.

For the purposes of this paper, the authors have found that the text-book k-means implementation works perfectly well in practice, and can be significantly more efficient than the k-SVD method, since efficient mini-batch and online implementations exist, which are also fairly straightforward to implement.

In classical clustering analysis, one seeks to find a set of clusters that represent the data density well; that split the data into as many segments as possible, but without over-fitting. This is a crucial aspect of cluster learning. In the case of dictionary learning, one wishes to learn a representation that is as over-complete as possible, and over-fitting is not a real risk because the dictionary needs to be redundant. The key here is the redundancy in **D** required for a sparse solver to provide a good approximation.

What this means in practice is that if two dictionary columns, $\mathbf{d}_j$ and $\mathbf{d}_i$ provide equally good representations for the data, a sparse coder will not weight equally between them, but will pick one and attempt to shrink the other as much as possible towards zero.

Even though projection and clustering can have the same formulations, the constraints applied to **D** are still different and thus yield different solutions. In clustering, there is no specific requirement for a decomposition of the data, whereas this may be enforced in a projection setting. In certain cases, some projections have been found to be equivalent to clustering models. For example, there is an equivalence between k-means and non-negative matrix factorisation [43].

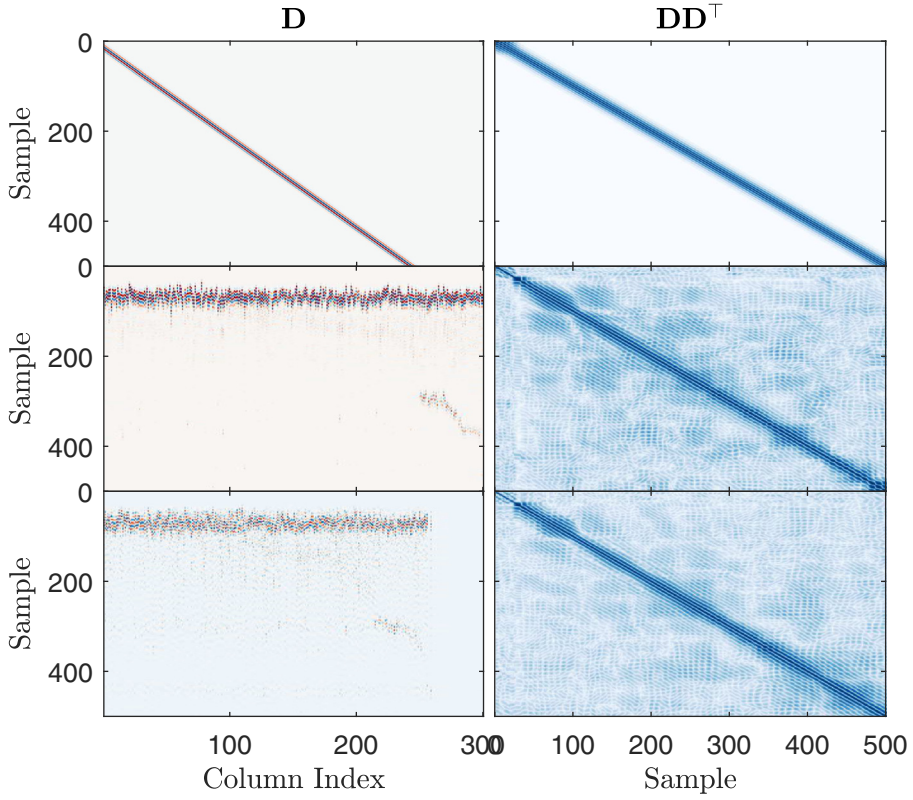

Fig. 5. Comparison of three different dictionaries for the ultrasound data-set a) model-based tone-burst, b) k-means clustering and c) online matrix factorisation. The first column shows **D** while the second column shows $\mathbf{DD}^\top$.

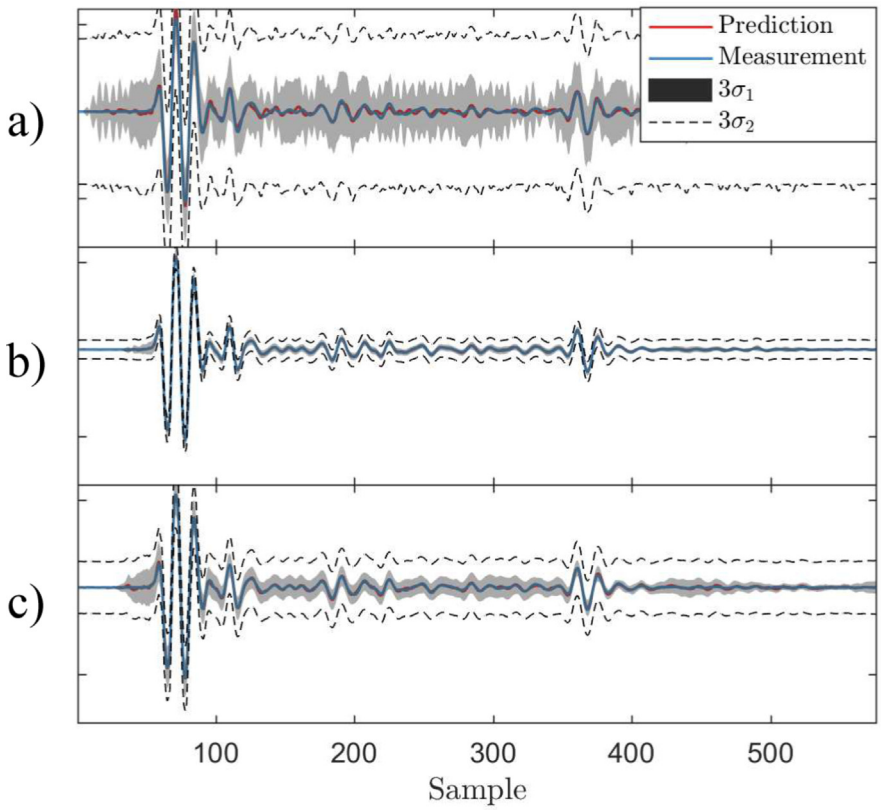

Fig. 6. Comparison of predictions on ultrasound data using three different dictionaries: a) model-based tone-burst, b) k-means clustering and c) online matrix factorisation. Two different uncertainty bounds are shown: σ_1 shows predictive variance without the noise term, whilst σ_2 shows the result of the additive noise term.

5.3. Effect of dictionary choice on predictive uncertainty

As already discussed, a major feature of any Bayesian formulation of a statistical model is that the predictive distribution encodes information about the uncertainty of the predictions. This section discusses the effects that the choice of dictionary form can have on the predictive distribution.

The role that the dictionary plays in quantifying the uncertainty of the predictions is evident by revisiting Eqs. (22) and (23), which define the predictive mean and covariance over the signal reconstruction, respectively. Note that the predictive variance is a sum of two terms: $\mathbf{V}_{\star} = \sigma^2 + \mathbf{D}\boldsymbol{\Sigma}\mathbf{D}^{\top}$; σ^2 encodes the measurement uncertainty while $\mathbf{D}\boldsymbol{\Sigma}\mathbf{D}^{\top}$ encodes the uncertainty in the process (how well the data relates to the dictionary).

Fig. 5 illustrates three different dictionaries, derived for the ultrasound pulses discussed in Section 1:

1. Model based dictionary, consisting of shifted tone-bursts. As discussed in Section 2.2 this leads to a deconvolution problem for this specific application.
2. Clustering-based dictionary, assembled using a simple k-means algorithm.
3. Online matrix factorisation-based dictionary, derived using the algorithm in [34], and summarised above.

As in Fig. 4, on the left is the matrix $\mathbf{D}$, sorted by increasing TOF index, while on the right the product $\mathbf{D}\mathbf{D}^{\top}$ is shown to illustrate the covariance structure.[3] Fig. 6 illustrates the difference in predictions using three different types of dictionaries. The greyed-out area in Fig. 6 shows the process uncertainty, while the dashed line denotes the confidence intervals due to full terms in Eq. (23) including measurement noise. The model-based dictionary is fairly bad at estimating the uncertainty both in terms of measurement and process noise, but has a tendency to explain most of the uncertainty as measurement noise. The simple k-means dictionary performs best in this particular example, having a tight confidence interval relating to the process noise and a small additive measurement noise term. One reason for this is that the EM algorithm, used to train EM, runs with the

[3] For visualisation purposes, the covariances have been scaled according to their diagonal variance and square-rooted. This better highlights the cross-variance terms of each individual dimension and the square root removes the effect of the original squaring that happens in $\mathbf{X}\mathbf{X}^{\top}$

full batch of data. The number of iterations over which the algorithm can "improve" is given by a tolerance on an increase in its objective function. On the other hand, the number of iterations of the online matrix factorisation is given by the number of training points, and it cannot improve beyond that. There is, however, nothing stopping one from iterating several times over a training set, in order to achieve a better estimate.

One more point to highlight is that the online matrix factorisation algorithm has a tendency to leave unused columns as they are found during initialisation. Therefore, if it is initialised with a random matrix with a low variance (relative to the data, which should be scaled anyway), then it is easy to identify them and remove them in order to enhance signal reconstruction performance with the RVM. In this example, these unused columns are evident on the right-hand side of the dictionary in Fig. 5c. Note that this is advantageous given the computational burden of having unnecessary columns in a dictionary.

6. Performance on ultrasound C-scan data

In this section, the performance of the Bayesian Compressive Sensing framework presented above is evaluated for the particular application of estimating ultrasound waveforms on the type of C-scan data that was introduced in Section 1. The interest here is focused on the reconstruction performance of the algorithm. Classically, this can be assessed simply by analysing the reconstruction errors, using a measure such as a Mean Squared Error (MSE). However, because this is a Bayesian CS scheme, reconstruction performance has to involve an assessment of the prediction uncertainty (confidence intervals).

6.1. Experimental set-up and data acquisition on composite wing panel

A 1.2 m $\times$ 3 m composite panel was scanned using a six-axis robotic head, with a water coupled ultrasound probe. The probe consists of 64 transducers, each of which fire a 5 MHz tone burst, and also act as receivers. The resolution of the scan can be adjusted, but for these results, the speed of the probe was adjusted to yield a spatial resolution of 0.8 mm in the direction of the probe travel. The C-scan shown in Fig. 1c was generated using this data set, using a maximum autocorrelation to estimate the TOF. Further details of this experimental technique have been published in [44], where the interested reader is referred to for further details.

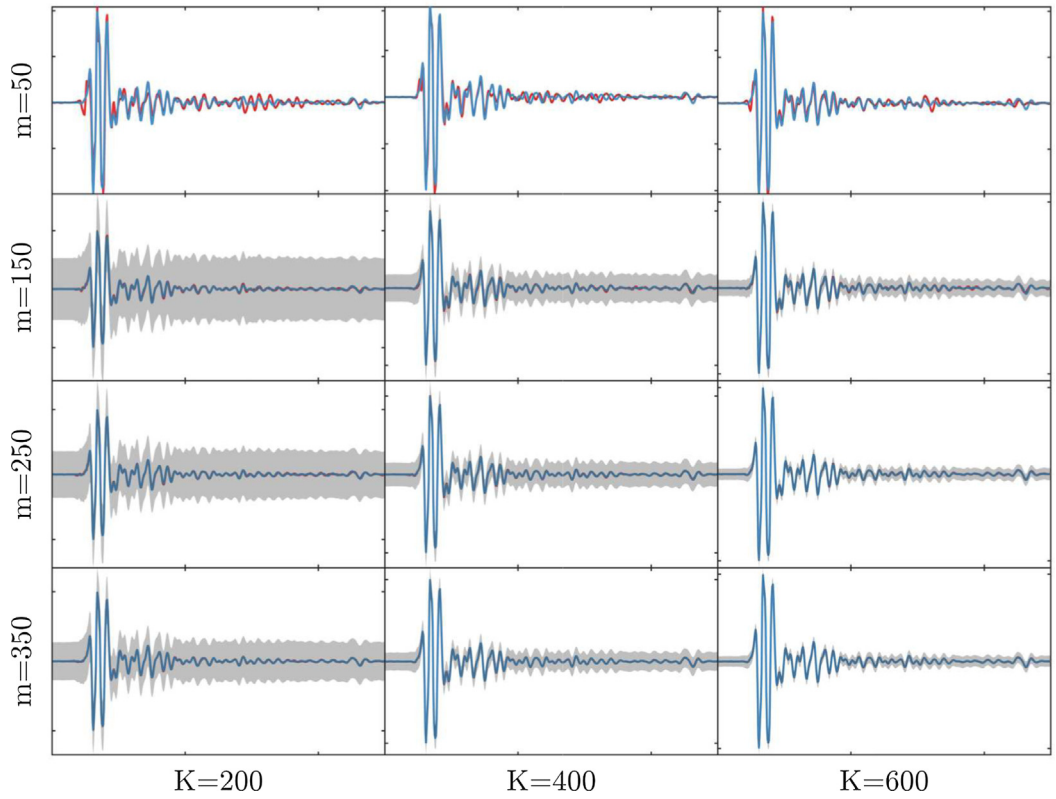

Fig. 7. Effect of compressed signal size and dictionary size on confidence intervals of an illustrative ultrasound pulse. This prediction uses a k-means dictionary. *m* denotes the compressed dimension while *K* denotes the dictionary size. The original dimension was 1232 data points.

From a signal processing point of view, the important points are that all of these signals contain information at a narrow band around centred around 5 MHz, and the Nyquist frequency lies at just 25 MHz, so the problem is not oversampled. This is an important point to make, given the problem of compression. It would be trivial to demonstrate a CS algorithm on a problem that is significantly over-sampled, where the same (or a better) level of compression could be achieved by a simple decimation without any loss of information. Sadly, this has been the case in some recent applications, for example [45].

In order to capture the range of different depths of this composite sample, an acquisition time of 24.64μs was used. This equates to 1232 samples at a sample rate of 50 MHz. The original ultrasound pulses collected through the water-coupled probe are susceptible to misalignment with respect to the arrival time of the first burst (the front wall reflection). This is evident, for example, in the illustrative B-scan shown in 1b.

It would help both the dictionary learning and signal reconstruction steps if this misalignment could be removed as a pre-processing step. This could be done with a simple threshold or via a more accurate and advanced onset estimation method such as those presented in [8]. However, the results presented here have been derived without this correction, as it makes matters a little bit more interesting given the variability in the alignment of the pulses.

Although this particular data-set is fairly large (32 million ultrasound pulses), it has been truncated for the purposes of analysing the performance of the Bayesian CS scheme. More specifically, 5000 of these pulses have been selected at random from the entire plate. From this set, half was used for training of dictionaries and half for testing. This yields two independent training and testing data sets, consisting of 2500 ultrasound pulses each.

6.2. Results

It has already been illustrated through the examples in Section 5 that a signal can be reconstructed fairly accurately with tight confidence intervals if one uses an appropriate data-driven dictionary learning strategy. It has been shown through illustration how lower compression ratios lead to wider uncertainty bounds, as well as how a dictionary with more columns will lead to a reduction of both reconstruction error and uncertainty. This was shown in Fig. 6.

This point is generalised in here, using the 2500 training and testing pulses as discussed above. Several metrics can be extracted to show the overall reconstruction performance. Four different metrics are examined here:

1. Number of relevant vectors: this shows the level of sparsity selected by the algorithm.

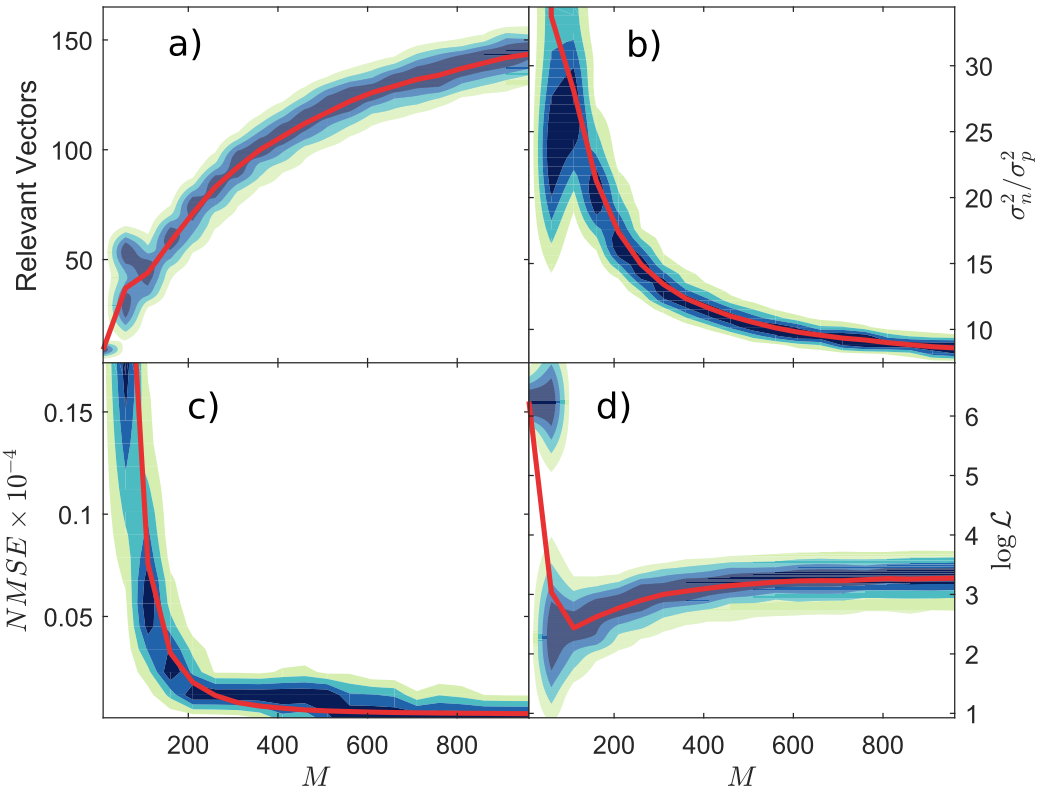

Fig. 8. Compressed dimension, M against a) number of selected relevant vectors, b) ratio of measurement noise to process variance, c) normalised mean squared error and d) log-likelihood. These metrics come from 500 randomly selected ultrasound pulses from the C-scan data of Fig. 1. The red line denotes the median, while the colour-map represents the density for the 500 draws.

2. Ratio of measurement noise to process variance: this outlines how much of the uncertainty is explained as simple measurement noise relative to process noise.
3. Normalised mean squared error (NMSE): the NMSE gives an indication of the performance of the mean estimate, against the true signal.
4. Compressed data log-likelihood, $\log \mathcal{L}$: this gives a measure of the *predicted* uncertainty.

Dictionary training was carried out using k-means clustering with 200 cluster centres on the training set of pulses. Fig. 8 shows a summary of the four different metrics evaluated on the testing set, for varying levels of compression, M. The plots show a contour of the normalised density of that metric, with four levels, so that each colour represents a different quantile. The red line that goes through the centre of the density represents the median of the metric. Note that values of M were used in the range between 50 and 900.

There are many interesting points to observe from Fig. 8. The number of relevant vectors is a good place to start, as these largely explain the rest of the results. As discussed in Section 4.2, high compression ratios (low M) lead to solutions in terms of fewer relevant vectors, in order for the product $\mathbf{\Phi R}$ (which defines the randomly projected set of active dictionary columns) to be invertible. Using fewer dictionary columns to explain the signal leads to higher error and this is evident by examination of the NMSE metric. The high error rates at high compression levels are a direct result of the low number of relevant vectors. The average number of relevant vectors, using a solution without applying compression, is 150. Clearly, Fig. 8a shows that the number of relevant vectors gets asymptotically closer to this value as M increases, which is a fairly intuitive result.

The ratio of measurement noise to process variance, shown in Fig. 8b, tells an interesting story. The ratio is very high at low values of M. In fact, under around $M = 100$, it says that most of the uncertainty in signals are explained as noise. This is a direct result of the collapsing of the posterior covariance, $\mathbf{\Sigma}$ at low compression levels, which leads to a very small process variance.

Finally, the data likelihood, shown in Fig. 8d shows very clearly how the problem becomes ill-posed under approximately $M = 100$, as it becomes unstable and multi-modal; otherwise it is nicely asymptotic as M increases.

7. Conclusions

This paper has presented a framework for performing Bayesian compressive sensing. The key point is that the sparse coding step can be solved as a Bayesian linear regression problem with a sparse prior, and the Relevance Vector Machine is well suited to solve this problem efficiently. The main advantage of going through the trouble of the Bayesian formulation is that unlike the traditional methods used in the sparse coding step of compressive sensing, a Bayesian approach offers automatic tuning of the sparsity level, as well as an estimate of the uncertainty bounds of the signal reconstruction. It has been emphasised and demonstrated that using data-driven dictionaries, which capture the original variability in the data, is key to the successful application of this method. The Bayesian CS scheme has been demonstrated in ultrasound signal processing applications, where data compression is a key to widespread application, but an appropriate understanding of the quality of the reconstructed signals is also important, and this is given by the uncertainty quantification.

Acknowledgements

This work has been funded by the UK Engineering and Physical Sciences Research Council (EPSRC), through the Autonomous Inspection in Manufacturing and Re-manufacturing (AIMaReM) grant EP/N018427/1. Support for K. Worden from the EPSRC through grant reference number EP/J016942/1 is also gratefully acknowledged.

References

[1] D.L. Donoho, De-noising by soft-thresholding, IEEE Trans. Inf. Theory 41 (3) (1995) 613–627.
[2] E. Candès, Compressive sampling, in: Proceedings of the International Congress of Mathematicians Madrid, August 2230, 2006, European Mathematical Society Publishing House, Zuerich, Switzerland, 2006, pp. 1433–1452.
[3] E.J. Candes, T. Tao, Near-optimal signal recovery from random projections: universal encoding strategies?, IEEE Trans Inf. Theory 52 (Dec 2006) 5406–5425.
[4] G. Cardoso, J. Saniie, Data compression and noise suppression of ultrasonic NDE signals using wavelets, in: IEEE Symposium on Ultrasonics, 2003, IEEE, 2015, pp. 250–253, vol. 16.
[5] S. Legendre, J. Goyette, D. Massicotte, Ultrasonic NDE of composite material structures using wavelet coefficients, NDT&E Int. 34 (2001) 31–37.
[6] M. Tipping, Sparse Bayesian learning and the relevance vector machine, J. Mach. Learn. Res. 1 (2001) 211–244.
[7] M. Parrilla, J. Anaya, C. Fritsch, Digital signal processing techniques for high accuracy ultrasonic range measurements, IEEE Trans. Instrum. Meas. 40 (4) (1991) 759–763.
[8] J.H. Kurz, C.U. Grosse, H.W. Reinhardt, Strategies for reliable automatic onset time picking of acoustic emissions and of ultrasound signals in concrete, Ultrasonics 43 (7) (2005) 538–546.
[9] F. Bomann, G. Plonka, T. Peter, O. Nemitz, T. Schmitte, Sparse deconvolution methods for ultrasonic NDT: application on TOFD and wall thickness measurements, J. Nondestr. Eval. 31 (3) (2012) 225–244.
[10] G.-M. Zhang, D.M. Harvey, Contemporary ultrasonic signal processing approaches for nondestructive evaluation of multilayered structures, Nondestr. Testing Eval. 27 (1) (2012) 1–27.
[11] R. Demirli, J. Saniie, Model-based estimation of ultrasonic echoes part I: Analysis and algorithms, IEEE Trans. Ultrason. Ferroelectr. Freq. Control 48 (3) (2001) 787–802.

[12] R. Fuentes, K. Worden, I. Antoniadou, C. Mineo, S.G. Pierce, E.J. Cross, Compressive sensing for direct time of flight estimation in ultrasound-based NDT, in: Proceedings of the 11th International Workshop in Structural Health Monitoring, (Palo Alto, CA), 2017.
[13] W.B. Johnson, J. Lindenstrauss, Extensions of Lipschitz mappings into a Hilbert space, Contemp. Math. 26 (1984) 189–206.
[14] S.G. Mallat, Z. Zhang, Matching pursuits with time-frequency dictionaries, IEEE Trans. Signal Process. 41 (12) (1993) 3397–3415.
[15] R.R. Coifman, M.V. Wickerhauser, Entropy-based algorithms for best basis selection, IEEE Trans. Inf. Theory 38 (2) (1992) 713–718.
[16] S. Chen Shaobing, D.L. Donoho, M.A. Saunders, Atomic decomposition by basis pursuit, Soc. Ind. Appl. Math. 43 (1) (2001) 129–159.
[17] C.M. Bishop, Latent Variable Models in Graphical Models, Published in Learning, 1999, pp. 371–403.
[18] R. Tibshirani, Regression Selection and Shrinkage via the Lasso, 1996.
[19] B. Efron, T. Hastie, Least angle regression, Ann. Stat. 32 (2) (2004) 407–499.
[20] J. Friedman, T. Hastie, H. Höfling, R. Tibshirani, Pathwise coordinate optimization, Ann. Appl. Stat. 1 (2) (2007) 302–332.
[21] M.E. Tipping, A.C. Faul, Fast marginal likelihood maximisation for sparse Bayesian models, in: Proceedings of the ninth, 2003, no. x, pp. 1–13.
[22] A. Dempster, N. Laird, D.B. Rubin, Maximum likelihood from incomplete data via the EM algorithm, J. R. Stat. Soc. Ser. B Methodol. 39 (1) (1977) 1–38.
[23] A. Faul, A. Tipping, Analysis of sparse Bayesian learning, Advances in Neural Information Processing Systems, 2002, pp. 383–389.
[24] K.P. Murphy, Conjugate Bayesian analysis of the Gaussian distribution, tech. rep., 2007.
[25] T. Hastie, R. Tibshirani, M. Wainwright, Statistical Learning with Sparsity: The Lasso and Generalizations, CRC, 2015, p. 362.
[26] T. Park, G. Casella, The bayesian lasso, J. Am. Stat. Assoc. 103 (482) (2008) 681–686.
[27] L. He, L. Carin, Exploiting structure in wavelet-based bayesian compressive sensing, IEEE Trans. Signal Process. 57 (9) (2009) 3488–3497.
[28] S. Ji, Y. Xue, L. Carin, Bayesian compressive sensing, IEEE Trans. Signal Process. 56 (6) (2008) 2346–2356.
[29] J.B. Harley, J.M.F. Moura, Sparse recovery of the multimodal and dispersive characteristics of Lamb waves, J. Acoust. Soc. Am. 133 (5) (2013) 2732–2745.
[30] B.A. Olshausen, D.J. Field, Sparse coding with an overcomplete basis set: a strategy employed by V1?, Vision Res. 37 (23) (1997) 3311–3325.
[31] M.S. Lewicki, T.J. Sejnowski, Learning overcomplete representations, Neural Comput. 12 (2) (2000) 337–365.
[32] K. Kreutz-Delgado, J.F. Murray, B.D. Rao, K. Engan, T.-W. Lee, T.J. Sejnowski, Dictionary learning algorithms for sparse representation, Neural Comput. 15 (2) (2003) 349–396.
[33] I. Tosic, P. Frossard, Dictionary learning, what is the right representation for my signal?, Signal Process Mag. IEEE 28 (2) (2011) 27–38.
[34] J. Mairal, F. Bach, J. Ponce, G. Sapiro, Online learning for matrix factorization and sparse coding, J. Mach. Learn. Res. 11 (2010) 19–60.
[35] H. Lee, A. Battle, R. Raina, A.Y. Ng, Efficient sparse coding algorithms, Adv. Neural Inf. Process. Syst. 19 (2007) 801–808.
[36] L. Bottou, O. Bousquet, The Tradeoffs of Large Scale Learning, Nips, vol. 20, 2007, pp. 161–168.
[37] O. Cappe, K. Mengersen, M. Titterington, C.P. Robert, Online Expectation-Maximisation, in: Mixtures: Estimation and Applications, Wiley, 2011, pp. 1–53.
[38] C.M. Bishop, Pattern Recognition and Machine Learning, Springer-Verlag, New York, 2006.
[39] S. Roweis, Z. Ghahramani, A unifying review of linear gaussian models, Neural Comput. 11 (2) (1999) 305–345.
[40] J.A. Tropp, Topics in sparse approximation (Ph.D. thesis), University of Texas at Austin, 2004.
[41] P. Schmid-Saugeon, A. Zakhor, Dictionary design for matching pursuit and application to motion compensated video coding, IEEE T. Circ. Syst. Vid. 14 (6) (2004) 880–886.
[42] M. Aharon, M. Elad, A. Bruckstein, K-SVD: an algorithm for designing overcomplete dictionaries for sparse representation, IEEE Trans. Signal Process. 54 (11) (2006) 4311–4322.
[43] H. Park, Nonnegative Matrix Factorization for Clustering, Modern Massive Data Sets, 2012, pp. 1–26.
[44] C. Mineo, C. MacLeod, M. Morozov, S.G. Pierce, R. Summan, T. Rodden, D. Kahani, J. Powell, P. McCubbin, C. McCubbin, G. Munro, S. Paton, D. Watson, Flexible integration of robotics, ultrasonics and metrology for the inspection of aerospace components, 2017.
[45] Y. Bao, Z. Shi, X. Wang, H. Li, Compressive sensing of wireless sensors based on group sparse optimization for structural health monitoring, Struct. Health Monit. Int. J. (2017).