ABSTRACT

Nonlinear inverse problems are often hampered by non-uniqueness and local minima because of missing low frequencies and far offsets in the data, lack of access to good starting models, noise, and modeling errors. A well-known approach to counter these deficiencies is to include prior information on the unknown model, which regularizes the inverse problem. While conventional regularization methods have resulted in enormous progress in ill-posed (geophysical) inverse problems, challenges remain when the prior information consists of multiple pieces. To handle this situation, we propose an optimization framework that allows us to add multiple pieces of prior information in the form of constraints. Compared to additive regularization penalties, constraints have a number of advantages making them more suitable for inverse problems such as full-waveform inversion. The proposed framework is rigorous because it offers assurances that multiple constraints are imposed uniquely at each iteration, irrespective of the order in which they are invoked. To project onto the intersection of multiple sets uniquely, we employ Dykstra's algorithm that scales to large problems and does not rely on trade-off parameters. In that sense, our approach differs substantially from approaches such as Tikhonov regularization, penalty methods, and gradient filtering. None of these offer assurances, which makes them less suitable to full-waveform inversion where unrealistic intermediate results effectively derail the iterative inversion process. By working with intersections of sets, we keep expensive objective and gradient calculations unaltered, separate from projections, and we also avoid trade-off parameters. These features allow for easy integration into existing code bases. In addition to more predictable behavior, working with constraints also allows for heuristics where we built up the complexity of the model gradually by relaxing the constraints. This strategy helps to avoid convergence to local minima that represent unrealistic models. We illustrate this unique feature with examples of

varying complexity.

INTRODUCTION

We propose an optimization framework to include prior knowledge in the form of constraints into nonlinear inverse problems that are typically hampered by the presence of parasitic local minima. We favor this approach over more commonly known regularization via (quadratic) penalties because including constraints does not alter the objective, and therefore first- and second-order derivative information. Moreover, constraints do not need to be differentiable, and most importantly, they offer guarantees that the updated models meet the constraints at each iteration of the inversion. While we focus on seismic full-waveform inversion (FWI), our approach is more general and applies in principle to any linear or nonlinear geophysical inverse problem as long as its objective is differentiable so it can be minimized with local derivative information to calculate descent directions that reduce the objective.

In addition to the above important features, working with constraints offers several additional advantages. For instance, because models always remain within the constraint set, inversion with constraints mitigates the adverse effects of local minima which we encounter in situations where the starting model is not accurate enough or where low-frequency and long-offset data are missing or too noisy. In these situations, derivative-based methods are likely to end up in a local minimum mainly because of the oscillatory nature of the data and the non-convexity of the objective. Moreover, costs of data acquisition and limitations on available compute also often force us to work with only small subsets of data. As a result, the inversions may suffer from artifacts. Finally, noise in the data and modeling errors can also give rise to artifacts. We will demonstrate that by adding constraints, which prevent these artifacts from occurring in the estimated models, our inversion results can be greatly

improved and make more geophysical and geological sense.

To deal with each of the challenging situations described above, geophysicists traditionally often rely on Tikhonov regularization, which corresponds to adding differentiable quadratic penalties that are connected to Gaussian Bayesian statistics on the prior. While these penalty methods are responsible for substantial progress in working with geophysical ill-posed and ill-conditioned problems, quadratic penalties face some significant shortcomings. Chiefly amongst these is the need to select a penalty parameter, which weights the trade-off between data misfit and prior information on the model. While there exists an extensive literature on how to choose this parameter in the case of a single penalty term (e.g., Vogel, 2002; Zhdanov, 2002; Sen and Roy, 2003; Farquharson and Oldenburg, 2004; Mueller and Siltanen, 2012), these approaches do not easily translate to situations where we want to add more than one type of prior information. There is also no simple prior distribution to bound pointwise values on the model without making assumptions on the underlying and often unknown statistical distribution (see Backus, 1988; Scales and Snieder, 1997; Stark, 2015). By working with constraints, we avoid making these type of assumptions.

Outline

Our primary goal is to develop a comprehensive optimization framework that allows us to directly incorporate multiple pieces of prior information in the form of multiple constraints. The main task of the optimization is to ensure that the inverted models meet all constraints during each iteration. To avoid certain ambiguities, we will do this with projections so that the updated models are unique, lie in the intersection of all constraints and remain as close as possible to the model updates provided by FWI without constraints.

There is an emerging literature on working with constrained optimization, see Lelièvre and Oldenburg (2009); Zeev et al. (2006); Bello and Raydan (2007); Baumstein (2013); Smithyman et al. (2015); Esser et al. (2015); Esser et al. (2016); Esser et al. (2018) and Peters and Herrmann (2017). Because this is relatively new to the geophysical community, we first start with a discussion on related work and what the limitations are of unconstrained regularization methods. Next, we discuss how to include (multiple pieces of) prior information with constraints. This discussion includes projections onto convex sets and how to project onto intersections of convex sets. After describing these important concepts, we combine them with nonlinear optimization and describe concrete algorithmic instances based on spectral-projected gradients and Dykstra's algorithm. We conclude by demonstrating our approach on an FWI problem

Notation

Before we discuss the advantages of constrained optimization for FWI, let us first establish some mathematical notation. Our discretized unknown models live on regular grids with N grid points represented by the model vector $\mathbf{m} \in \mathbb{R}^N$, which is the result of lexicographically ordering the 2 or 3D models. In Table 1 we list a few other definitions we will use.

[Table 1 about here.]

Related work

A number of authors use constraints to include prior knowledge in nonlinear geophysical inverse problems. Most of these works focus on only one or maximally two constraints. For instance; Zeev et al. (2006); Bello and Raydan (2007) and Métivier and Brossier (2016)

consider nonlinear geophysical problems with only bound constraints, which they solve with projection methods. Because projections implement these bounds exactly, these methods avoid complications that may arise if we attempt to approximate bound constraints by differentiable penalty functions. While standard differentiable optimization can minimize the resulting objective with quadratic penalties, there is no guarantee the inverted parameters remain within the specified range at every grid point during each iteration of the inversion. Moreover, there is also no consistent and easy way to add multiple constraints reflecting complementary aspects (e.g., bounds and smoothness) of the underlying geology. Bound constraints in a transformed-domain are discussed by Lelièvre and Oldenburg (2009).

Close in spirit to the approach we propose is recent work by Becker et al. (2015), who introduces a quasi-Newton method with projections and proximal operators (see e.g., Parikh and Boyd, 2014) to add a single ℓ_1 norm constraint or penalty on the model in FWI. These authors include this non-differentiable norm to induce sparsity on the model by constraining the ℓ_1 norm in some transformed domain or on the gradient as in total-variation minimization. While their method uses the fact that it is relatively easy to project on the ℓ_1-ball, they have to work on the coefficients rather than on the physical model parameters themselves, and this makes it difficult to combine this transform-domain sparsity with say bound constraints that live in another transform-domain. As we will demonstrate, we overcome this problem by allowing for multiple constraints in multiple transform-domains simultaneously.

Several authors present algorithms that can incorporate multiple constraints simultaneously, but there are some subtle, but important algorithmic details that we discuss in this paper. For instance, the work by Baumstein (2013) employs the well-known projection-onto-convex-sets (POCS) algorithm, which can only be shown to converge to the projection of a point in special cases, see, e.g., work by Escalante and Raydan (2011) and Bauschke and

Combettes (2011). Projecting the updated model parameters onto the intersection of multiple constraints solves this problem and offers guarantees that each model iterate (model after each iteration) remains after projection the closest in Euclidean distance to the unconstrained model and at the same time satisfies all the constraints. Different methods exist to ensure that the model iterates remain within the non-empty intersection of multiple constraint sets. Most notably, we would like to mention the work by the late Ernie Esser (Esser et al., 2018), who developed a scaled gradient projection method for this purpose involving box constraints, total-variation, and hinge-loss constraints. Esser et al. (2018) arrived at this result by using a primal-dual hybrid gradient (PDHG) method, which derives from Lagrangians associated with total-variation and hinge-loss minimization. To allow for more flexibility in the number and type of constraints, we propose the use of Dykstra's algorithm (Dykstra, 1983; Boyle and Dykstra, 1986) instead. As we will show, this approach offers maximal flexibility in the number and complexity—i.e., we do not need closed-form expressions for the projections, of the constraints we would like to impose. We refer to Smithyman et al. (2015) and Peters and Herrmann (2017) for examples of successful geophysical applications of multiple constraints to FWI and its distinct advantages over adding constraints as weighted penalties.

LIMITATIONS OF UNCONSTRAINED REGULARIZATION METHODS

In the introduction, we stated our requirements on a regularization framework for nonlinear inverse problems. While there are a number of successful regularization approaches such as Tikhonov regularization, a change of variables, gradient filtering and modified Gauss-Newton, these methods miss one or more of our desired properties listed in the introduction. Below we will show why the above methods do not generalize to multiple constraints or do so at

the cost of introducing additional manual tuning parameters.

Tikhonov and quadratic regularization

Perhaps the most well known and widely used regularization technique in geophysics is the addition of quadratic penalties to a data-misfit function. Let us denote the model vector with medium parameters by $\mathbf{m} \in \mathbb{R}^N$ (for example velocity) where the number of grid points is N. The total objective with quadratic regularization $\phi(\mathbf{m}) : \mathbb{R}^N \to \mathbb{R}$ is given by

$$\phi(\mathbf{m}) = f(\mathbf{m}) + \frac{\alpha_1}{2}\|\mathbf{R}_1\mathbf{m}\|_2^2 + \cdots + \frac{\alpha_p}{2}\|\mathbf{R}_p\mathbf{m}\|_2^2. \tag{1}$$

In this expression, the data misfit function $f(\mathbf{m}) : \mathbb{R}^N \to \mathbb{R}$ measures the difference between predicted and observed data. A common choice for the data-misfit is

$$f(\mathbf{m}) = \frac{1}{2}\|\mathbf{d}^{\text{pred}}(\mathbf{m}) - \mathbf{d}^{\text{obs}}\|_2^2, \tag{2}$$

where $\mathbf{d}^{\text{obs}}$ and $\mathbf{d}^{\text{pred}}(\mathbf{m})$ are observed and predicted (from the current model $\mathbf{m}$) data, respectively. The predicted data may depend on the model parameters in a nonlinear way.

There are p regularization terms in equation 1, all of which describe different pieces of prior information in the form of differentiable quadratic penalties weighted by scalar penalty parameters $\alpha_1, \alpha_2, \ldots, \alpha_p$. The operators $R_i \in \mathbb{C}^{M_i \times N}$ are selected to penalize unwanted properties in $\mathbf{m}$—i.e., we select each R such that the penalty terms become large if the model estimate does not lie in the desired class of models. For example, we will promote smoothness of the model estimate $\mathbf{m}$ if we add horizontal or vertical discrete derivatives as $\mathbf{R}_1$ and $\mathbf{R}_2$.

Aside from promoting certain properties on the model, adding penalty terms also changes the gradient and Hessian—i.e., we have

$$\nabla_{\mathbf{m}}\phi(\mathbf{m}) = \nabla_{\mathbf{m}} f(\mathbf{m}) + \alpha_1\mathbf{R}_1^*\mathbf{R}_1\mathbf{m} + \alpha_2\mathbf{R}_2^*\mathbf{R}_2\mathbf{m} \tag{3}$$

and

$$\nabla^2_{\mathbf{m}}\phi(\mathbf{m}) = \nabla^2_{\mathbf{m}} f(\mathbf{m}) + \alpha_1 \mathbf{R}_1^* \mathbf{R}_1 + \alpha_2 \mathbf{R}_2^* \mathbf{R}_2. \tag{4}$$

Both expressions, where* refers to the complex conjugate transpose, contain contributions from the penalty terms designed to add certain features to the gradient and to improve the spectral properties of the Hessian by applying a shift to the eigenvalues of $\nabla^2_{\mathbf{m}}\phi(\mathbf{m})$.

While regularization of the above type has been applied successfully, it has two important disadvantages. First, it is not straightforward to encode one's confidence in a starting model other than including a reference model ($\mathbf{m}_{\text{ref}}$) in the quadratic penalty term—i.e., $\alpha/2\|\mathbf{m}_{\text{ref}} - \mathbf{m}\|_2^2$ (see, e.g., Farquharson and Oldenburg (2004) and Asnaashari et al. (2013)). Unfortunately, this type of penalty tends to spread deviations with respect to this reference model evenly so we do not have easy control over its local values (cf. box constraints) unless we provide detailed prior information on the covariance. Secondly, quadratic penalties are antagonistic to models that exhibit sparse structure—i.e., models that can be well approximated by models with a small total-variation or by transform-domain coefficient (e.g. Fourier, wavelet, or curvelet) vectors with a small ℓ_1-norm or cardinality ($\|\cdot\|_0$ "norm"). Regrettably, these sparsifying norms are non-differentiable, which often leads to problems when they are added to the objective by smoothing or reweighting the norms. In either case, this can lead to slower convergence, to unpredictable behavior in nonlinear inverse problems (Anagaw, 2014,page 110; Lin and Huang, 2015,and Peters and Herrmann (2017)) or to a worsening of the conditioning of the Hessian (Akcelik et al., 2002). Even without smoothing non-differential penalties, there are still penalty parameters to select (Farquharson and Oldenburg, 1998; Becker et al., 2015; Lin and Huang, 2015; Xue and Zhu, 2015; Qiu et al., 2016). Finally, these issues with quadratic penalties are not purely theoretical. For instance, when working with a land dataset, (Smithyman et al., 2015) found that the above limitations

of penalty terms hold in practice and found that constraint optimization overcomes these limitations, an observation motivating this work.

Gradient filtering

Aside from adding penalties to the data-misfit, we can also remove undesired model artifacts by filtering the gradients of $f(\mathbf{m})$. When we minimize the data objective (cf. Equation 1) with standard gradient descent, this amounts to applying a filter to the gradient when we update the model—i.e., we have

$$\mathbf{m}^{k+1} = \mathbf{m}^k - \gamma s(\nabla_{\mathbf{m}} f(\mathbf{m})), \tag{5}$$

where γ is the step-length and $s(\cdot)$ a nonlinear or linear filter. For instance; Brenders and Pratt (2007) apply a 2D spatial low-pass filter to prevent unwanted high-wavenumber updates to the model when inverting low-frequency seismic data. The idea behind this approach is that noise-free low-frequency data should give rise to smooth model updates. While these filters can remove unwanted high-frequency components of the gradient, this method has some serious drawbacks.

First, the gradient is no longer necessarily a gradient of the objective function (Equation 1) after applying the filter. Although the filtered gradient may under certain technical conditions remain a decent direction, optimization algorithms, such as spectral projected gradient (SPG) (Birgin et al., 1999) or quasi-Newton methods (Nocedal and Wright, 2000), expect true gradients when minimizing (constrained) objectives. Therefore gradient filtering can generally not be used in combination with these optimization algorithms, without giving up their expected behavior. Second, it is not straightforward to enforce more than one property on the model in this way. Consider, for instance, a two-filter case where $s_1(\cdot)$ is a smoother and

$s_2(\cdot)$ enforces upper and lower bounds on the model. In this case, we face the unfortunate ambiguity $s_2(s_1(\nabla_{\mathbf{m}} f(\mathbf{m}))) \neq s_1(s_2(\nabla_{\mathbf{m}} f(\mathbf{m})))$. Moreover, this gradient will have non-smooth clipping artifacts if we smooth first and then apply the bounds. Anagaw and Sacchi (2017) present a method that filters the updated model instead of a gradient, but it is also not clear how to extend this filtering technique to more than one model property.

Change of variables / subspaces

Another commonly employed method to regularize nonlinear inverse problems involves certain (possibly orthogonal) transformations of the original model vector. While somewhat reminiscent of gradient filtering, this approach entails a change of variables, see, e.g., Jervis et al. (1996); Shen et al. (2005); Shen and Symes (2008) for examples in migration velocity analysis and Li et al. (2013); Kleinman and den Berg (1992); Guitton et al. (2012); Guitton and Díaz (2012) for examples the context of waveform inversion. This approach is also known as a subspace method (Kennett and Williamson, 1988; Oldenburg et al., 1993). We can invoke this change of variables by transforming the model into $\mathbf{p} = \mathbf{T}\mathbf{m}$, where $\mathbf{T}$ is a (not necessarily invertible) linear operator. This changes the unconstrained optimization problem $\min_{\mathbf{m}} f(\mathbf{m})$ into another unconstrained problem $\min_{\mathbf{p}} f(\mathbf{p})$. To see why this might be helpful, we observe that the gradient becomes $\nabla_{\mathbf{p}} f(\mathbf{p}) = \mathbf{T}^* \nabla_{\mathbf{m}} f(\mathbf{m})$, which shows how $\mathbf{T}$ can be designed to 'filter' the gradient. The matrix $\mathbf{T}$ can also represent a subspace (limited number of basis vectors such as splines, wavelets). Just as with gradient filtering, a change of variables does not easily lend itself to multiple transforms aimed at incorporating complementary pieces of prior information. However, subspace information fits directly into the constrained optimization approach if we constrain our models to be elements of the subspace. The constrained approach has the advantage that we can straightforwardly combine it with other

constraints in multiple transform-domains; all constraints in the proposed framework act on the variables $\mathbf{m}$ in the physical space since we do not minimize subspace/transform-domain coefficients $\mathbf{p}$.

Modified Gauss-Newton

A more recent successful attempt to improve model estimation for certain nonlinear inverse problems concerns imposing curvelet domain ℓ_1-norm based sparsity constraints on the model updates (Herrmann et al., 2011; Li et al., 2012, 2016). This approach converges to local minimizers of $f(\mathbf{m})$ (and hopefully a global one) because sparsity constrained updates provably remain descent directions (Burke (1990), chapter 2; Herrmann et al. (2011)). However, there are no guarantees that the curvelet coefficients of the model itself will remain sparse unless the support (= locations of the non-zero coefficients) is more or less the same for each Gauss-Newton update (Li et al., 2016). Zhu et al. (2017) use a similar approach, but they update the transform (dictionary) at every FWI iteration.

In summary, while regularizing the gradients or model updates leads to encouraging results for some applications, the constrained optimization approach proposed in this work enforces constraints on the model estimate itself, without modifying the gradient. More importantly, while imposing constraints via projections may superficially look similar to the above methods, our proposed approach differs fundamentally in two main respects. Firstly, it projects uniquely on the intersection of arbitrarily many constraint sets — effectively removing the ambiguity of order in which constraints are applied. Secondly, it does not alter the gradients because it imposes the projections on the proposed model updates, i.e., we will project $\mathbf{m}^{k+1} = \mathbf{m}^k - \nabla_{\mathbf{m}} f(\mathbf{m})$ onto the constraint set.

INCLUDING PRIOR INFORMATION VIA CONSTRAINTS

Before we introduce constrained formulations of nonlinear inverse problems with multiple convex and non-convex constraint sets, we first discuss some important core properties of convex sets, of projections onto convex sets, and of projections onto intersections of convex sets. These properties provide guarantees that our approach generalizes to arbitrarily many constraint sets, i.e., one constraint set is mathematically the same as many constraint sets. The presented convex set properties also show that there is no need to somehow order of the sets to avoid ambiguity, as was the case for gradient filtering and of naive implementations of constrained optimization. The constrained formulation also stays away from penalty parameters, yet still offers guarantees all constraints are satisfied at every iteration of the inversion.

Constrained formulation

To circumvent problems related to incorporating multiple sources of possibly non-differentiable prior information, we propose techniques from constrained optimization (Boyd and Vandenberghe, 2004; Boyd et al., 2011; Parikh and Boyd, 2014; Beck, 2015; Bertsekas, 2015). The key idea of this approach is to minimize the data-misfit objective while at the same time making sure that the estimated model parameters satisfy constraints. These constraints are mathematical descriptors of prior information on certain physical (e.g., maximal and minimal values for the wavespeed) and geological properties (e.g., velocity models with unconformities that lead to discontinuities in the wavespeed) on the model. We build our formulation on earlier work on constrained optimization with up to three constraint sets as presented by Lelièvre and Oldenburg (2009); Smithyman et al. (2015); Esser et al. (2015); Esser et al.

(2016); Esser et al. (2018); Peters and Herrmann (2017).

Given an arbitrary but finite number of constraint sets (p), we formulate our constrained optimization problem as follows:

$$\min_{\mathbf{m}} f(\mathbf{m}) \text{ subject to } \mathbf{m} \in \bigcap_{i=1}^{p} \mathcal{C}_i. \tag{6}$$

As before, $f(\mathbf{m}) : \mathbb{R}^N \to \mathbb{R}$ is the data-misfit objective, which we minimize over the discretized medium parameters represented by the vector $\mathbf{m} \in \mathbb{R}^N$. Prior knowledge on this model vector resides in the indexed constraint sets $\mathcal{C}_i, i = 1 \cdots p$, each of which captures a known aspect of the Earth's subsurface. These constraints may include bounds on permissible parameter values, desired smoothness or complexity, or limits on the number of layers in sedimentary environments and many others.

In cases where more than one piece of prior information is available, we want the model vector to satisfy these constraints simultaneously, such that we keep control over the model properties as is required for strategies that relax constraints gradually (Esser et al., 2016; Peters and Herrmann, 2017). Because it is difficult to think of a nontrivial example where the intersection of these sets is empty, it is safe to assume that there is at least one model that satisfies all constraints simultaneously. For instance, a homogeneous medium will satisfy many constraints, because its total-variation is zero, it has a rank of 1 and has parameter values between minimum and maximum values. We denote the mathematical requirement that the estimated model vector satisfies p constraints simultaneously by $\mathbf{m} \in \bigcap_{i=1}^{p} \mathcal{C}_i$. The symbol $\bigcap_{i=1}^{p}$ indicates the intersection of p items. Before we discuss how to solve constrained nonlinear geophysical inverse problems, let us first discuss projections and examples of projections onto convex and non-convex sets.

Convex sets

A projection of $\mathbf{m}$ onto a set $\mathcal{C}$ corresponds to solving

$$\mathcal{P}_{\mathcal{C}}(\mathbf{m}) = \arg\min_{\mathbf{x}} \frac{1}{2}\|\mathbf{x}-\mathbf{m}\|_2^2 \text{ subject to } \mathbf{x} \in \mathcal{C}. \tag{7}$$

Amongst all possible model vectors $\mathbf{x}$, the above optimization problem finds the vector $\mathbf{x}$ that is closest in Euclidean distance to the input vector $\mathbf{m}$ while it lies in the constraint set. For a given model vector $\mathbf{x}$, the solution of this optimization problem depends on the constraint set $\mathcal{C}$ and its properties. For instance, the above projection is unique for a convex $\mathcal{C}$.

To better understand how to incorporate prior information in the form of one or more constraint sets, let us first list some important properties of constraint sets and their intersection. These properties allow us to use relatively simple algorithms to solve Problem 6 by using projections of the above type. First of all, most optimization algorithms require the constraint sets to be convex. Intuitively, a set is convex if any point on the line segment connecting any two points in a set is also in the set—i.e., for all $\mathbf{x} \in \mathcal{C}$ and $\mathbf{y} \in \mathcal{C}$. In that case, the following relation holds:

$$c\mathbf{x} + (1-c)\mathbf{y} \in \mathcal{C} \quad \text{for } 0 \leq c \leq 1. \tag{8}$$

There are a number of advantages when working with convex sets, namely

i. The intersection of convex sets is also a convex set. This property implies that the properties of a convex set also hold for the intersection of arbitrarily many convex sets. Practically, if an optimization algorithm is defined for a single convex set, the algorithm also works in case of arbitrarily many convex sets, as the intersection is still a single convex set.

ii. The projection onto a convex set is unique (Boyd and Vandenberghe, 2004,section 8.1). When combined with property *(i)*, this implies that the projection onto the intersection of multiple convex sets is also unique. In this context, a unique projection means that given any point outside a convex set, there exists one point in the set which is closest (in a Euclidean sense) to the given point than any other point in the set.

iii. Projections onto convex sets are non-expansive, see e.g., (Bauschke and Combettes, 2011,section 4.1-4.2, or Dattorro (2010), E.9.3). If we define the projection operator as $\mathcal{P}_{\mathcal{C}}(\mathbf{x})$ and take any two points $\mathbf{x}$ and $\mathbf{y}$, the non-expansive property is stated as: $\|\mathcal{P}_{\mathcal{C}}(\mathbf{x}) - \mathcal{P}_{\mathcal{C}}(\mathbf{y})\| \leq \|\mathbf{x} - \mathbf{y}\|$. This property guarantees that projections of estimated models on a convex set are 'stable'. In this context, stability implies that any pair of models moves closer or remain equally distant to each other after projection. This prevents increased separation after projection of pairs of models.

While these properties make convex sets favorites amongst practitioners of (convex) optimization, restricting ourselves to convexity is sometimes too limiting for our application. In the following sections, we may use non-convex sets in the same way as a convex set, but in that case, the above properties generally do not hold. Performance of the algorithms then needs empirical verification.

Actual projections onto a single set themselves are either available in closed-form (e.g., for bounds and certain norms) or are computed iteratively (with the alternating direction method of multipliers, ADMM, see e.g., Boyd et al. (2011) and Appendix A) when closed form-expressions for the projections are not available.

COMPUTING PROJECTIONS ONTO INTERSECTIONS OF CONVEX SETS

Our problem formulation, equation 6, concerns multiple constraints, so we need to be able to work with multiple constraint sets simultaneously to make sure the model iterates satisfy all prior knowledge. To avoid intermediate model iterates to become physically and geologically unfeasible, we want our model iterates to satisfy a predetermined set of constraints at every iteration of the inversion process. Because of property *(i)* (listed above), we can treat the projection onto the intersection of multiple constraints as the projection onto a single set. This implies that we can use relatively standard (convex) optimization algorithm to solve Problem 6 as long as the intersection of the different convex sets is not empty. We define the projection on the intersection of multiple sets as

$$\mathcal{P}_{\mathcal{C}}(\mathbf{m}) = \arg\min_{\mathbf{x}} \|\mathbf{x} - \mathbf{m}\|_2^2 \quad \text{s.t.} \quad \mathbf{x} \in \bigcap_{i=1}^{p} \mathcal{C}_i. \tag{9}$$

The projection of $\mathbf{m}$ onto the intersection of the sets, $\bigcap_{i=1}^{p} \mathcal{C}_i$, means that we find the unique vector $\mathbf{x}$, in the intersection of all sets, that is closest to $\mathbf{m}$ in the Euclidean sense. To find this vector, we compute the projection onto this intersection via Dykstra's alternating projection algorithm (Dykstra, 1983; Boyle and Dykstra, 1986; Bauschke and Koch, 2015). We made this choice because this algorithm is relatively simple to implement (we only need projections on each set individually) and contains no manual tuning parameters. By virtue of property *(ii)*, projecting onto each set separately and cyclically, Dykstra's algorithm finds the unique projection on the intersection as long as all sets are convex (Boyle and Dykstra, 1986,Theorem 2).

To illustrate how Dykstra's algorithm works, let us consider the following toy example where we project the point (2.5, 3.0) onto the intersection of two constraint sets, namely a

halfspace ($\|y\|_2 \leq 2$, this corresponds to bound constraints in two dimensions) and a disk ($x^2 + y^2 \leq 3$, this corresponds to a $\|\cdot\|_2$-norm ball), see Figure 1. If we are just interested in finding a feasible point in the set that is not necessarily the closest, we can use the projection onto convex sets (POCS) algorithm (also known as von Neumann's alternating projection algorithm) whose steps are depicted by the solid black line in Figure 1. The POCS algorithm iterates $\mathcal{P}_{\mathcal{C}_2}(\ldots(\mathcal{P}_{\mathcal{C}_1}(\mathcal{P}_{\mathcal{C}_2}(\mathcal{P}_{\mathcal{C}_1}(\mathbf{m})))))$, so depending on whether we first project on the rectangle or disk, POCS finds two different feasible points. Like POCS, Dykstra projects onto each set in an alternating fashion, but unlike POCS, the solution path that is denoted by the red dashed line provably ends up at a single unique feasible point that is closest to the starting point. The solution found by Dykstra is independent of the order with which the constraints are imposed. POCS does not project onto the intersection of the two convex sets; it just solves the convex feasibility problem

$$\textbf{find } \mathbf{x} \in \bigcap_{i=1}^{p} \mathcal{C}_i \tag{10}$$

instead. POCS finds a model that satisfies all constraints but which is non-unique (solution is either (1.92, 2.0) or (2.34, 1.87) situated at Euclidean distances 1.16 and 1.14) and not the projection at $(2.0, \sqrt{2^2 + 3^2} \approx 2.24)$ at a minimum distance of 1.03. This lack of uniqueness and vicinity to the true solution of the projection problem leads to solutions that satisfy the constraints, but that may be too far away from the initial point and this may adversely affect the inversion. See also (Escalante and Raydan, 2011,Example 5.1; Dattorro, 2010,Figure 177 & Figure 182, and Bauschke and Combettes (2011), Figure 29.1) for further details on this important point.

The geophysical implication of this difference between Dykstra and POCS is that the latter may end up solving a problem with unnecessarily tight constraints, moving the model

too far away from the descent direction informed by the data misfit objective. We observe this phenomenon of being too constrained in Figure 1 where the two solutions from POCS are not on the boundary of both sets, but instead relatively 'deep' inside one of them. Aside from potential "over constraining", the results from POCS may also differ depending which of the individual constraints is activated first leading to undesirable side effects. The issue of "over constraining" does not just occur in geometrical two-dimensional examples and it is not specific to the constraints from the previous example. Figure 2 shows what happens if we project a velocity model (with Dykstra) or find two feasible models with POCS, just as we show in Figure 1. The constraint is the intersection of bounds ($\{\mathbf{m} \mid \mathbf{l}_i \leq \mathbf{m}_i \leq \mathbf{u}_i\}$) and total-variation ($\{\mathbf{m} \mid \|\mathbf{A}\mathbf{m}\|_1 \leq \sigma\}$ with scalar $\sigma > 0$ and $A = (\mathbf{D}_x^T \ \mathbf{D}_z^T)^T$). While one of the POCS results is similar to the projection, the other POCS result has much smaller total-variation than the constraint enforces, i.e., the result of POCS is not the projection but a feasible point in the interior of the intersection. To avoid these issues, Dykstra is our method of choice to include two or more constraints into nonlinear inverse problems. Algorithm 1 summarizes the main steps of Dykstra's approach, which aside from stopping conditions, is parameter free. In Figure 3 we show what happens if we replace the projection (with Dykstra's algorithm) in projected gradient descent with POCS. Projected gradient descent solves an FWI problem with bounds and total-variation constraints while using a small number of sources and receivers and an incorrectly estimated source function. The results of Dykstra and POCS are different, while the results using POCS depend on the ordering of the sets. Dykstra's algorithm is independent of the ordering because it finds the projection onto an intersection of convex sets, which is unique.

[Figure 1 about here.]

Algorithm 1 Dykstra's algorithm, following the notation of Birgin and Raydan (2005), to compute the projection of $\mathbf{m}$ onto the intersection of p convex sets: $\mathcal{P}_{\mathcal{C}}(\mathbf{m}) = \arg\min_{\mathbf{x}} \|\mathbf{x} - \mathbf{m}\|_2^2$ s.t. $\mathbf{x} \in \bigcap_{i=1}^{p} \mathcal{C}_i$. $\mathbf{y}_i$ are auxiliary vectors.

Algorithm DYKSTRA($\mathbf{m}, \mathcal{P}_{\mathcal{C}_1}, \mathcal{P}_{\mathcal{C}_2}, \ldots, \mathcal{P}_{\mathcal{C}_p}$)

0a. $\mathbf{x}_p^0 = \mathbf{m}$, $k = 1$ `//initialize`

0b. $\mathbf{y}_i^0 = 0$ for $i = 1, 2, \ldots, p$ `//initialize`

WHILE stopping conditions not satisfied **DO**

1. $\mathbf{x}_0^k = x_p^{k-1}$

FOR $i = 1, 2, \ldots, p$

2. $\mathbf{x}_i^k = \mathcal{P}_{\mathcal{C}_i}(\mathbf{x}_{i-1}^k - \mathbf{y}_i^{k-1})$

END

FOR $i = 1, 2, \ldots, p$

3. $\mathbf{y}_i^k = \mathbf{x}_i^k - (\mathbf{x}_{i-1}^k - \mathbf{y}_i^{k-1})$

END

4. $k = k + 1$

END

output: $\mathbf{x}_p^k$

[Figure 2 about here.]

[Figure 3 about here.]

NONLINEAR OPTIMIZATION WITH PROJECTIONS

So far, we discussed a method to project models onto the intersection of multiple constraint sets. How these projections interact with nonlinear optimization problems with expensive

to evaluate (data-misfit) objectives is not yet clear and also delicate given the devastating effects parasitic local minima may have on the progression of the inversion. Moreover, aside from our design criteria (multiple constraints instead of competing penalties; guarantees that model iterations remain in constraint set), we need to include a clean separation of misfit/gradient calculations and projections so that we avoid additional expensive PDE solves at all times. This separation also allows us to use different codes bases for each task (objective/gradient calculations versus projections).

Projected-gradient descent

The simplest first-order algorithm that minimizes a differentiable objective function subject to constraints is the projected gradient method (e.g., Beck (2014), section 9.4). This algorithm is a straightforward extension of the well-known gradient-descent method (e.g., Bertsekas, 2015, section 2.1) involving the following updates on the model:

$$\mathbf{m}^{k+1} = \mathcal{P}_{\mathcal{C}}\big(\mathbf{m}^k - \gamma \nabla_{\mathbf{m}} f(\mathbf{m}^k)\big). \tag{11}$$

A line search determines the scalar step length $\gamma > 0$. This algorithm first takes a gradient-descent step, involving an expensive gradient calculation, followed by the often much cheaper projection of the updated model back onto the intersection of the constraint sets. By construction, the computationally expensive gradient computations (and data-misfit for the line search) are separate from projections onto constraints. The projection step itself guarantees that the model estimate $\mathbf{m}^k$ satisfies all constraints at every k^{th} iteration.

Figure 4 illustrates the difference between gradient descent to minimize a a two variable non-convex objective $\min_{\mathbf{m}} f(\mathbf{m})$, and projected gradient descent to minimize $\min_{\mathbf{m}} f(\mathbf{m})$ s.t. $\mathbf{m} \in \mathcal{C}$. If we compare the solution paths for gradient and projected gradient

descent, we see that the latter explores the boundary as well as the interior of the constraint set $\mathcal{C} = \{\mathbf{m}| \, \|\mathbf{m}\|_2 \leq \sigma\}$ to find a minimizer. This toy example highlights how constraints pose upper limits (the set boundary) on certain model properties but do not force solutions to stay on the constraint set boundary. Because one of the local minima lies outside the constraint set, this example also shows that adding constraints may guide the solution to a different (correct) local minimizer. This is exactly what we want to accomplish with constraints for FWI: prevent the model estimate $\mathbf{m}^k$ to convergence to local minimizers that represent unrealistic models.

[Figure 4 about here.]

Spectral projected gradient

Standard projected gradient has two important drawbacks. First, we need to project onto the constraint set after each line search step, which involves multiple expensive objective calculations. To be more specific, we need to calculate the step-length parameter $\gamma \in (0, 1]$ if the objective of the projected model iterate is larger than the current model iterate—i.e., $f\left(\mathcal{P}_{\mathcal{C}}(\mathbf{m}^k - \gamma\nabla_{\mathbf{m}} f(\mathbf{m}^k))\right) > f(\mathbf{m}^k)$. In that case, we need to reduce γ and test again whether the data-misfit is reduced. For every reduction of γ, we need to recompute the projection and evaluate the objective, which is too expensive. Second, first-order methods do not use curvature information, which involves the Hessian of $f(\mathbf{m})$ or access to previous gradient and model iterates. Projected gradient algorithms are therefore often slower than Newton, Gauss-Newton, or quasi-Newton algorithms for FWI without constraints.

To avoid these two drawbacks and possible complications arising from the interplay of imposing constraints and correcting for Hessians, we use the spectral projected gradient

method (SPG; Birgin et al. (1999); Birgin et al. (2003)); an extension of the standard projected gradient algorithm (Equation 11), which corresponds to a simple scalar scaling (related to the eigenvalues of the Hessian, see Birgin et al. (1999) and Dai and Liao (2002)). At model iterate k, the SPG iterations involve the step

$$\mathbf{m}^{k+1} = \mathbf{m}^k + \gamma \mathbf{p}^k, \tag{12}$$

with update direction

$$\mathbf{p}^k = \mathcal{P}_{\mathcal{C}}\big(\mathbf{m}^k - \alpha \nabla_{\mathbf{m}} f(\mathbf{m}^k)\big) - \mathbf{m}^k. \tag{13}$$

These two equations define the core of SPG, which differs from standard projected gradient descent in three different ways:

i. The spectral stepsize α (Barzilai and Borwein, 1988; Raydan, 1993; Dai and Liao, 2002) is calculated by from the secant equation (Nocedal and Wright, 2000, section 6.1) to approximate the Hessian, leading to accelerated convergence. An interpretation of the secant equation is to mimic the action of the Hessian by scalar α and use finite-difference approximations for the second derivative of $f(\mathbf{m})$. This approach is closely related to the idea behind quasi-Newton methods. We compute α as the solution of

$$\mathbf{D}^k = \underset{\mathbf{D}=\alpha I}{\arg\min} \|\mathbf{D}\mathbf{s}^k - \mathbf{y}^k\|_2, \tag{14}$$

where $\mathbf{y}^k = \nabla_{\mathbf{m}} f(\mathbf{m}^{k+1}) - \nabla_{\mathbf{m}} f(\mathbf{m}^k)$ and $\mathbf{s}^k = \mathbf{m}^{k+1} - \mathbf{m}^k$, and $\mathbf{I}$ the identity matrix. This results in scaling by $\alpha = \mathbf{s}_k^* \mathbf{s}_k / \mathbf{s}_k^* \mathbf{y}_k$ derived from gradient and model iterates from the current and previous SPG iterations. Clearly, this is computationally cheap because α is not computed by a separate line search.

ii. Spectral-projected gradient employs non-monotone (Grippo and Sciandrone, 2002) inexact line searches to calculate the γ in equation 12. As for all FWI problems, $f(\mathbf{m})$

is not convex so we cannot use an exact line-search. Non-monotone means that the objective function value is allowed to increase temporarily, which often results in faster convergence and fewer line search steps, see, e.g., Birgin et al. (1999) for numerical experiments. Our intuition behind this is as follows: gradient descent iterations often exhibit a 'zig-zag' pattern when the objective function behaves like a 'long valley' in a certain direction. When the line searches are non-monotone, the objective does not always have to go down so we can take relatively larger steps along the valley in the direction of the minimizer that are slightly 'uphill', increasing the objective temporarily.

iii. Each SPG iteration requires only one projection onto the intersection of constraint sets to compute the update direction (Equation 13) and does not need additional projections for line search steps. This is a significant computational advantage over standard projected gradient descent, which computes one projection per line search step, see equation 11. From equations 12 and 13, we observe that $\gamma \mathbf{p}^k$ lies on the line between the previous model estimate ($\mathbf{m}^k$) and the proposed update, projected back onto the feasible set—i.e., $\mathcal{P}_{\mathcal{C}}(\mathbf{m}^k - \alpha \nabla_{\mathbf{m}} f(\mathbf{m}^k))$. Therefore, $\mathbf{m}^{k+1}$ is on the line segment between these two points in a convex set and the new model will satisfy all constraints simultaneously at every iteration (see equation 8). For this reason, any line search step that reduces γ will also be an element of the convex set. Works by Zeev et al. (2006) and Bello and Raydan (2007) confirm that SPG with non-monotone line searches can lead to significant acceleration on FWI and seismic reflection tomography problems with bound constraints compared to projected gradient descent.

In summary, each SPG iteration in Algorithm 2 requires at the k^{th} iteration a single evaluation of the objective $f(\mathbf{m}^k)$ and gradient $\nabla_{\mathbf{m}} f(\mathbf{m}^k)$. In fact, SPG combines data-misfit

minimization (our objective) with imposing constraints, while keeping the data-misfit/gradient and projection computations separate. When we impose the constraints, the objective and gradient do not change. Aside from computational advantages, this separation allows us to use different code bases for the objective $f(\mathbf{m})$ and its gradient $\nabla_{\mathbf{m}} f(\mathbf{m})$ and the imposition of the constraints. The above separation of responsibilities also leads to a modular software design, which applies to different inverse problems that require (costly) objective and gradient calculations.

Spectral projected gradient with multiple constraints

We now arrive at our main contribution where we combine projections onto multiple constraints with nonlinear optimization with costly objective and gradient calculations using a spectral projected gradient (SPG) method. Recall from the previous section that the projection onto the intersection of convex sets in SPG is equivalent to running Dykstra's algorithm (Algorithm 1) —i.e., we

$$\begin{aligned} &\mathcal{P}_{\mathcal{C}}(\mathbf{m}^k - \alpha \nabla_{\mathbf{m}} f(\mathbf{m}^k)) \\ &= \mathcal{P}_{\mathcal{C}_1 \bigcap \mathcal{C}_2 \bigcap \cdots \bigcap \mathcal{C}_p}(\mathbf{m}^k - \alpha \nabla_{\mathbf{m}} f(\mathbf{m}^k)) \\ &\Leftrightarrow \text{DYKSTRA}(\mathbf{m}^k - \alpha \nabla_{\mathbf{m}} f(\mathbf{m}^k), \mathcal{P}_{\mathcal{C}_1}, \ldots, \mathcal{P}_{\mathcal{C}_p}). \end{aligned} \tag{15}$$

With this equivalence established, we arrive at our version of SPG presented in Algorithm 2, which has appeared in some form in the non-geophysical literature Birgin et al. (2003) and Schmidt and Murphy (2010).

The proposed optimization algorithm for nonlinear inverse problems with multiple constraints (Eq. 6) has the following three-level nested structure:

1. At the top level, we have a possibly non-convex optimization problem with a differen-

Algorithm 2 $\min_{\mathbf{m}} f(\mathbf{m})$ s.t. $\mathbf{m} \in \bigcap_{i=1}^{p} \mathcal{C}_i$ with spectral-projected gradient, non-monotone line searches and combined with Dykstra's algorithm.

input:

`// one projector per constraint set`

$\mathcal{P}_{\mathcal{C}_1}, \mathcal{P}_{\mathcal{C}_2}, \ldots$

$\mathbf{m}^0$ `//starting model`

Initialization

0. $M =$ integer, $\eta \in (0, 1)$, select initial α

0. $k = 1$, select sufficient descent parameter ϵ

WHILE stopping conditions not satisfied **DO**

1. $f(\mathbf{m}^k)$, $\nabla_{\mathbf{m}} f(\mathbf{m}^k)$ `//objective & gradient`

`// project onto intersection of sets:`

2. $\mathbf{r}^k = \text{DYKSTRA}\big(\mathbf{m}^k - \alpha \nabla_{\mathbf{m}} f(\mathbf{m}^k), \mathcal{P}_{\mathcal{C}_1}, \mathcal{P}_{\mathcal{C}_2}, \ldots\big)$

3. $\mathbf{p}^k = \mathbf{r}^k - \mathbf{m}^k$ `// update direction`

`//save previous M objective:`

4a. $f_{\text{ref}} = \{f_k, f_{k-1}, \ldots, f_{k-M}\}$

4b. $\gamma = 1$

4c. **IF** $f(\mathbf{m}^k + \gamma \mathbf{p}^k) < \max(f_{\text{ref}}) + \epsilon \gamma \nabla_{\mathbf{m}} f(\mathbf{m}^k)^* \mathbf{p}^k$

$\mathbf{m}^{k+1} = \mathbf{m}^k + \gamma \mathbf{p}$ `// update model iterate`

$\mathbf{y}_k = \nabla_{\mathbf{m}} f(\mathbf{m}^{k+1}) - \nabla_{\mathbf{m}} f(\mathbf{m}^k)$

$\mathbf{s}_k = \mathbf{m}^{k+1} - \mathbf{m}^k$

$\alpha = \frac{\mathbf{s}_k^* \mathbf{s}_k}{\mathbf{s}_k^* \mathbf{y}_k}$ `// spectral steplength`

$k = k + 1$

ELSE

$\gamma = \eta \gamma$ `//step size reduction,`

go back to 4c

END

tiable objective and multiple possibly non-differentiable constraints:

$$\min_{\mathbf{m}} f(\mathbf{m}) \text{ subject to } \mathbf{m} \in \bigcap_{i=1}^{p} \mathcal{C}_i,$$

which we solve with the spectral projected gradient method;

2. At the next level, we project onto the intersection of multiple (convex) sets:

$$\mathcal{P}_{\mathcal{C}}(\mathbf{m}) = \arg\min_{\mathbf{x}} \|\mathbf{x} - \mathbf{m}\|_2 \text{ subject to } \mathbf{x} \in \bigcap_{i=1}^{p} \mathcal{C}_i$$

implemented via Algorithm 1 (Dykstra's algorithm);

3. At the lowest level, we project onto individual sets:

$$\mathcal{P}_{\mathcal{C}_i}(\mathbf{m}) = \arg\min_{\mathbf{x}} \|\mathbf{x} - \mathbf{m}\|_2 \text{ subject to } \mathbf{x} \in \mathcal{C}_i$$

for which we use ADMM (see Appendix B) if there is no closed form solution available.

While there are many choices for the algorithms at each level, we base our selection of any particular algorithm on their ability to solve each level without relying on additional manual tuning parameters. We summarized our choices in Figure 5, which illustrates the *three*-level nested optimization structure.

[Figure 5 about here.]

Numerical example

As we mentioned earlier, full-waveform inversion (FWI) faces problems with parasitic local minima when the starting model is not sufficiently accurate, and the data are cycle skipped.

FWI also suffers when no reliable data are available at the low end of the spectrum (typically less than 3 Hertz) or at offsets larger than about two times the depth of the model. Amongst the myriad of recent, sometimes somewhat ad hoc proposals to reduce the adverse effects of these local minima, we show how the proposed constrained optimization framework allows us to include prior knowledge on the unknown model parameters with guarantees that our inverted models indeed meet these constraints for each updated model.

Let us consider the situation where we may not have precise prior knowledge on the actual model parameters itself, but where we may still be in a position to mathematically describe some characteristics of a good starting model. With a good starting model, we mean a model that leads to significant progress towards the true model during nonlinear inversion. So our strategy is to first improve our starting model — by constraining the inversion such that the model satisfies our expectation of what a starting model looks like — followed by a second cycle of regular FWI. We relax constraints for the second cycle to allow for model updates that further improve the data fit. Figure 6 shows the actual and initial starting models for this 2D FWI experiment. For this purpose, we take a 2D slice from the BG Compass velocity and density model. We choose this model because it contains realistic velocity "kick back", which is known to challenge FWI. The original model is sampled at 6 m, and we generate "observed data" by running a time-domain (Louboutin et al., 2017) simulation code with the velocity and density models given in Figure 6. The sources and receivers (56 each) are located near the surface, with 100 m spacing. A coarse source and receiver spacing of a 100 m amounts to about one spatial wavelength at the highest frequency in the water; well below the spatial Nyquist sampling rate.

To mimic realistic situations where the simulation kernels of the inversion miss important aspects of the wave physics, we invert for velocity only while fixing the density to be equal

to one everywhere. While there are better approximations to the density model than the one we use, we intentionally use a rough approximation of the physics to show that constraints are also beneficial in that situation. To add another layer of complexity, we solve the inverse problem in the frequency domain (Da Silva and Herrmann, 2017) following the well-known multiscale frequency continuation strategy of Bunks (1995). To deal with the situation where ocean bottom marine data is often severely contaminated with noise at the low-end of the spectrum, we start the inversion on the frequency interval $3-4$ Hertz. We define this interval as a frequency batch. We subsequently use the result of the inversion with this first frequency batch as the starting model for the next frequency batch, inverting data from the $4-5$ Hertz interval. We repeat this process up to frequencies on the interval $14-15$ Hertz. We also estimate the unknown frequency spectrum of the source on the fly during each iteration, using the variable projection method by (Pratt, 1999; Aravkin and van Leeuwen, 2012). To avoid additional complications, we assume the sources and receivers to be omnidirectional with a flat spatial frequency spectrum.

[Figure 6 about here.]

While frequency continuation and on-the-fly source estimation are both well-established techniques by now, the combination of velocity-only inversion for a poor starting model remains challenging because we *(i)* ignore density variations in the inversion, which means we can never hope to fit the observed data fully; *(ii)* we miss the velocity kick back at roughly $300-500\,m$ in the starting model; and *(iii)* we invert on an up to roughly $10\times$ coarser grid compared to the fine $6\,m$ grid on which the "observed" time-domain data was generated. Because of these challenges, battle-tested multiscale workflows for FWI, where we start at the low frequencies and gradually work our way up to higher frequencies, fail even if we

impose bound constraints (minimum of 1425 (m/s) and maximum 5000 (m/s)) values for the estimated velocities) on the model. See Figure 7. Only the top 700 m of the velocity model is inverted reasonably well. The bottom part, on the other hand, is far from the true model almost everywhere. The main discontinuity into the ≥ 4000 (m/s) rock is not at the correct depth and does not have the right shape.

[Figure 7 about here.]

To illustrate the potential of adding more constraints on the velocity model, we follow a heuristic that combines multiple warm-started multiscale FWI cycles with relaxing the constraints. This approach was successfully employed in earlier work by Esser et al. (2016); Esser et al. (2018) and Peters and Herrmann (2017). Since we are dealing with a relatively undistorted sedimentary basin (see Figure 6), we impose constraints that limit lateral variations and force the inverted velocities to increase monotonically with depth during the first inversion cycle. In the second cycle, we relax this condition. We accomplish this by combining box constraints with slope constraints in the vertical direction (described in detail in Appendix B). To enforce continuity in the lateral direction, we work with tighter slope constraints in that direction. Specifically, we limit the variation of the velocity per meter in the depth direction (z-coordinate) of the discretized model $m[i,j] = m(i\Delta z, j\Delta x)$. Mathematically, we enforce $0 \leq (m[i+1,j] - m[i,j])/\Delta z \leq +\infty$ for $i = 1 \cdots n_z$ and $j = 1 \cdots n_x$, where n_z, n_x are the number of grid points in the vertical and lateral direction, and Δz the grid size in depth. With this slope constraint, the inverted velocities are only allowed to increase monotonically with depth, but there is no limit on how fast the velocity can increase in that direction. We impose lateral continuity by selecting the lateral slope constraint as $-\varepsilon \leq (m[i,j+1] - m[i,j])/\Delta x \leq +\varepsilon \quad$ for all $i = 1 \cdots n_z, j = 1 \cdots n_x$. The

scalar ε is a small number set in the physical units of velocity [meter/second] change per meter and Δx is the grid size in the lateral direction. We select $\varepsilon = 1.0$ for this example.

Compared to other methods that enforce continuity, e.g., via a sharpening operator in a quadratic penalty term, these slope constraints have several advantages. First, they have a natural interpretable physical parameter (ε). Second, they are met at each point in the model—i.e., they are applied and enforced pointwise; and most importantly these slope constraints do not impose more structure than needed. For instance, the vertical slope constraint only enforces monotonic increases and nothing else. We do not claim that other methods, such as Tikhonov regularization, cannot accomplish these features. We claim that we do this without nebulous parameter tuning and with guarantees that our constraints are satisfied at each iteration of FWI.

The FWI results with slope constraints for $3-4$ Hertz data are shown in Figure 8(a). This result from the first FWI cycle improves the starting model significantly without introducing geologically unrealistic artifacts. This partially inverted model can now serve as input for the second FWI cycle where we invert data over a broader frequency range between $3-15$ Herz (cf. Figure 8(b)) using box constraints only. Apparently, adding slope constraints during the first cycle is enough to prevent the velocity model from moving in the wrong direction while allowing for enough freedom to get closer to the true model underlying the success of the second cycle without slope constraints. This example demonstrates that keeping the recovered velocity model after the first FWI cycle in check — via not too constrained constraints — can be a successful strategy even though final velocity model does not lie in the constraint set imposed during the first FWI cycle where velocity kick back was not allowed. We kept the computational overhead of this multi-cycle FWI method to a minimum by working with low-frequency data only during the first cycle, which reduces the size of the

computational grid by a factor of about fourteen.

This example was designed to illustrate how our framework for constrained FWI can be of great practical use for FWI problems where good starting models are missing or where low-frequencies and long offsets are absent. Our proposed method is not tied to a specific constraint. For different geological settings, we can use the same approach, but with different constraints.

[Figure 8 about here.]

DISCUSSION

Our main contribution in solving optimization problems with multiple constraints is that we employ a hierarchical divide and conquer approach to handle problems with expensive to evaluate objectives and gradients. We arrive at this result by splitting each problem into simpler and therefore computationally more manageable subproblems. We start from the top with spectral-projected gradient (SPG), which splits the constrained optimization problem into an optimality (decreasing the objective) and feasibility (satisfying all constraints) problem, and continue downwards by satisfying the individual constraints using Dykstra's algorithm. Even at the lowest level, we employ this strategy when there is no closed form projection available for the constraints. We use the alternating direction method of multipliers (ADMM) for the examples. As a result, we end up with an algorithm that remains computationally feasible for large-scale problems with expensive to evaluate objectives and gradients.

So far, the minimization of our optimality problem relied on first-order derivative information only and what is essentially a scalar approximation of Hessian via SPG. Theoretically, we

can also incorporate Dykstra's algorithm into projected quasi-Newton (Schmidt et al., 2009) or (Gauss-) Newton methods (Schmidt et al., 2012; Lee et al., 2014). However, unlike SPG, these approaches usually require more than one projection computation per FWI iteration to solve quadratic sub-problems with constraints. We would require a more careful evaluation to see if second-order methods in this case indeed provide advantages compared to projected first-order methods such as SPG.

We also would like to note that there exist parallel versions of Dykstra and similar algorithms (Censor, 2006; Combettes and Pesquet, 2011; Bauschke and Koch, 2015). These algorithms compute all projections in parallel, so each Dykstra iteration takes as much time as the most expensive projection computations. As a result, the time per Dykstra iteration does not necessarily increase if there are more constraint sets.

CONCLUSIONS

Because of its computational complexity and notorious local minima, full-waveform inversion easily ranks amongst one of the most challenging nonlinear inverse problems. To meet this challenge, we introduced a versatile optimization framework for (non)linear inverse problems with the following key features: *(i)* it invokes prior information via projections onto the intersection of multiple (convex) constraint sets and thereby avoids reliance on cumbersome trade-off parameters; *(ii)* it allows for imposing arbitrarily many constraints simultaneously as long as their intersection is non-empty; *(iii)* it projects the updated models uniquely on the intersection at every iteration and as such stays away from ambiguities related to the order in which the constraints are invoked; *(iv)* it guarantees that model updates satisfy all constraints simultaneously at each iteration and *(v)* it is built on top of existing code bases that only need to compute data-misfit objective values and gradients. These features

in combination with our ability to relax and add other constraints that have appeared in the geophysical literature offer a powerful optimization framework to mitigate some of the adverse effects of local minima.

Aside from promoting certain to-be-expected model properties, our examples also confirmed that invoking multiple constraints as part of a multi-cycle inversion heuristic can lead to better results. These improvements occur as long as the constraint set is small enough to prevent adverse model updates caused by local minima to enter into the model estimate, during the first full-waveform inversion cycle(s). Provided the inversions make some progress to the solution, later inversion cycles will benefit if the tight constraints are subsequently relaxed either by dropping them or by increasing the size of the constraint set. Our examples confirm this important aspect and clearly demonstrate the advantages of working with constraints that are satisfied at each iteration of the inversion.

Compared to many other regularization methods, our approach is easily extendable to other convex or non-convex constraints. However, for non-convex constraints, we can no longer offer certain guarantees, except that all sub-problems in the alternating direction method of multipliers remain solvable without the need to tune trade-off parameters manually. We can do this because we work with projections onto the intersection of multiple sets and we split the computations into multiple pieces that have closed-form solutions.

APPENDIX A

Alternating Direction Method of Multipliers (ADMM) for the projection problem.

We show how to use Alternating Direction Method of Multipliers (ADMM) to solve projection problems. Iterative optimization algorithms are necessary in case there is no closed-form solution available. The basic idea is to split a 'complicated' problem into several 'simple' pieces. Consider a function that is the sum of two terms and where one of the terms contains a transform-domain operator: $\min_{\mathbf{x}} h(\mathbf{x}) + g(\mathbf{A}\mathbf{x})$. We proceed by renaming one of the variables, $\mathbf{A}\mathbf{x} \rightarrow \mathbf{z}$ and we also add the constraint $\mathbf{A}\mathbf{x} = \mathbf{z}$. This new problem is $\min_{\mathbf{x},\mathbf{z}} h(\mathbf{x}) + g(\mathbf{z})$ s.t. $A\mathbf{x} = \mathbf{z}$. The solution of both problems is the same, but algorithms to solve the new formulation are typically simpler. This formulation leads to an algorithm that can solve all projection problems discussed in this paper. Different projections only need different inputs but require no algorithmic changes.

As an example, consider the projection problem for ℓ_1 constraints in a transform-domain (e.g., total-variation, sparsity in the curvelet domain). The corresponding set is $\mathcal{C} \equiv \{\mathbf{m} \mid \|\mathbf{A}\mathbf{m}\|_1 \leq \sigma\}$ and the associated projection problem is

$$\mathcal{P}_{\mathcal{C}}(\mathbf{m}) = \arg\min_{\mathbf{x}} \frac{1}{2}\|\mathbf{x} - \mathbf{m}\|_2^2 \quad \text{s.t} \quad \|\mathbf{A}\mathbf{x}\|_1 \leq \sigma. \tag{16}$$

ADMM solves problems with the structure: $\min_{\mathbf{m},\mathbf{z}} h(\mathbf{m}) + g(\mathbf{z})$ s.t. $\mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{z} = \mathbf{c}$. The projection problem is of the same form as the ADMM problem. To see this, we use the indicator function on a set $\mathcal{C}$ as

$$\iota_{\mathcal{C}}(\mathbf{m}) = \begin{cases} 0 & \text{if } \mathbf{m} \in \mathcal{C}, \\ +\infty & \text{if } \mathbf{m} \notin \mathcal{C}. \end{cases} \tag{17}$$

The indicator function $\iota_{\ell_1}(\mathbf{Am})$ corresponds to the set $\mathcal{C}$ that we introduced above. We use the indicator function and variable splitting to rewrite the projection problem as

$$\begin{aligned}\mathcal{P}_{\mathcal{C}}(\mathbf{m}) &= \arg\min_{\mathbf{x}} \frac{1}{2}\|\mathbf{x}-\mathbf{m}\|_2^2 \quad \text{s.t} \quad \|\mathbf{Ax}\|_1 \leq \sigma \\ &= \arg\min_{\mathbf{x}} \frac{1}{2}\|\mathbf{x}-\mathbf{m}\|_2^2 + \iota_{\ell_1}(\mathbf{Am}) \\ &= \arg\min_{\mathbf{x},\mathbf{z}} \frac{1}{2}\|\mathbf{x}-\mathbf{m}\|_2^2 + \iota_{\ell_1}(\mathbf{z}) \quad \text{s.t} \quad \mathbf{Ax}=\mathbf{z}.\end{aligned} \tag{18}$$

We have $\mathbf{c}=0$ and $\mathbf{B}=-\mathbf{I}$ for all projection problems in this paper. The problem stated in the last line is the sum of two functions acting on different variables with additional equality constraints. This is exactly what ADMM solves. The following derivation is mainly based on Boyd et al. (2011). Identify $h(\mathbf{x})=\frac{1}{2}\|\mathbf{x}-\mathbf{m}\|_2^2$ and $g(\mathbf{z})=\iota_{\mathcal{C}}(\mathbf{z})$. ADMM uses the augmented-Lagrangian (Nocedal and Wright, 2000,chapter 17) to include the equality constraints $\mathbf{Ax}-\mathbf{z}=0$ as

$$L_\rho(\mathbf{x},\mathbf{z},\mathbf{v}) = h(\mathbf{x}) + g(\mathbf{z}) + \mathbf{v}^*(\mathbf{Ax}-\mathbf{z}) + \frac{\rho}{2}\|\mathbf{Ax}-\mathbf{z}\|_2^2. \tag{19}$$

The scalar ρ is a positive penalty parameter and $\mathbf{v}$ is the vector of Lagrangian multipliers. The derivation of the ADMM algorithm is non-trivial, see e.g., Ryu and Boyd (2016) for a derivation. Each ADMM iteration (k) has three main steps:

$$\begin{aligned}\mathbf{x}^{k+1} &= \arg\min_{\mathbf{x}} L_\rho(\mathbf{x},\mathbf{z}^k,\mathbf{v}^k) \\ \mathbf{z}^{k+1} &= \arg\min_{\mathbf{z}} L_\rho(\mathbf{x}^{k+1},\mathbf{z},\mathbf{v}^k) \\ \mathbf{v}^{k+1} &= \mathbf{v}^k + \rho(\mathbf{Ax}^{k+1}-\mathbf{z}^{k+1}).\end{aligned}$$

ADMM will converge to the solution as long as ρ is positive and reaches a stable value eventually. The choice of ρ does influence the number of iterations that are required (Nishihara et al., 2015; Xu et al., 2017, 2016) and the performance on non-convex problems (Xu et al., 2016). We use an adaptive strategy to adjust ρ at every iteration, see He et al. (2000). The

derivation proceeds in the scaled form with $\mathbf{u} = \mathbf{v}/\rho$. Reorganizing the equations leads to

$$\begin{aligned}
\mathbf{x}^{k+1} &= \arg\min_{\mathbf{x}} \Big(h(\mathbf{x}) + \frac{\rho}{2} \|\mathbf{A}\mathbf{x} - \mathbf{z}^k + \mathbf{u}^k\|_2^2 \Big) \\
\mathbf{z}^{k+1} &= \arg\min_{\mathbf{z}} \Big(g(\mathbf{z}) + \frac{\rho}{2} \|\mathbf{A}\mathbf{x}^{k+1} - \mathbf{z} + \mathbf{u}^k\|_2^2 \Big) \\
\mathbf{u}^{k+1} &= \mathbf{u}^k + \mathbf{A}\mathbf{x}^{k+1} - \mathbf{z}^{k+1}.
\end{aligned}$$

Now insert the expressions for $h(\mathbf{x})$ and $g(\mathbf{z})$ to obtain the more explicitly defined iterations

$$\begin{aligned}
\mathbf{x}^{k+1} &= \arg\min_{\mathbf{x}} \left(\frac{1}{2} \|\mathbf{x} - \mathbf{m}\|_2^2 + \frac{\rho}{2} \|\mathbf{A}\mathbf{x} - \mathbf{z}^k + \mathbf{u}^k\|_2^2 \right) \\
\mathbf{z}^{k+1} &= \arg\min_{\mathbf{z}} \Big(\iota_{\mathcal{C}}(\mathbf{z}) + \frac{\rho}{2} \|\mathbf{A}\mathbf{x}^{k+1} - \mathbf{z} + \mathbf{u}^k\|_2^2 \Big) \\
\mathbf{u}^{k+1} &= \mathbf{u}^k + \mathbf{A}\mathbf{x}^{k+1} - \mathbf{z}^{k+1}.
\end{aligned}$$

If we replace the minimization steps with their respective close-form solutions, we have the following pseudo-algorithm:

$$\begin{aligned}
\mathbf{x}^{k+1} &= (\rho \mathbf{A}^*\mathbf{A} + I)^{-1} \big(\rho \mathbf{A}^* (\mathbf{z}^k - \mathbf{u}^k) + \mathbf{m} \big) \\
\mathbf{z}^{k+1} &= \mathcal{P}_{\mathcal{C}}(\mathbf{A}\mathbf{x}^{k+1} + \mathbf{u}^k) \\
\mathbf{u}^{k+1} &= \mathbf{u}^k + \mathbf{A}\mathbf{x}^{k+1} - \mathbf{z}^{k+1}.
\end{aligned}$$

This shows that the second minimization step in the ADMM algorithm to compute a projection is a different projection. The projection part of ADMM for the transform-domain ℓ_1 constraint ($\mathbf{z}^{k+1} = \mathcal{P}_{\mathcal{C}}(\mathbf{A}\mathbf{x}^{k+1} + \mathbf{u}^k) = \arg\min_{\mathbf{z}} 1/2 \|\mathbf{z} - \mathbf{v}\|_2^2$ s.t $\|\mathbf{z}\|_1 \leq \sigma$, with $\mathbf{v} = \mathbf{A}\mathbf{x}^{k+1} + \mathbf{u}^k$) is a much simpler problem than the original projection problem (Equation 16) because we do not have the transform-domain operator multiplied with the optimization variable. The $\mathbf{x}$-minimization step is equivalent to the least-squares problem

$$\mathbf{x}^{k+1} = \arg\min_{\mathbf{x}} \left\| \begin{pmatrix} \sqrt{\rho}\mathbf{A} \\ \mathbf{I} \end{pmatrix} \mathbf{x} - \begin{pmatrix} \sqrt{\rho}(\mathbf{z}^k - \mathbf{u}^k) \\ \mathbf{m} \end{pmatrix} \right\|_2 \tag{20}$$

We can solve the $\mathbf{x}$-minimization problem using direct (QR-factorization) or iterative methods (LSQR (Paige and Saunders, 1982) on the least-squares problem or conjugate-gradient on the

normal equations). We adjust the penalty parameter ρ every ADMM cycle. We recommend iterative algorithms for this situation, to avoid recomputing the QR factorization every ADMM iteration. Iterative methods allow for the current estimate of $\mathbf{x}$ as the initial guess. Moreover, $\mathbf{z}$ and $\mathbf{u}$ change less as the ADMM iterations progress, meaning that the previous $\mathbf{x}$ is a better and better initial guess. Therefore, the number of LSQR iterations typically decreases as the number of ADMM iterations increases. Algorithm 3 shows the ADMM algorithm to compute projections, including automatic adaptive penalty parameter adjustment. For numerical experiments in this paper, we use $\mu = 10$, $\tau = 2$ as suggested by Boyd et al. (2011).

If we have a different constraint set, but same transform domain operator, we only change the projector that we pass to ADMM. If the constraint set is the same, but the transform-domain operator is different, we provide a different $\mathbf{A}$ to ADMM. Therefore, the various types of transform-domain ℓ_1, cardinality or bound constraints all use ADMM to compute the projection, but with (partially) different inputs.

APPENDIX B

Transform-domain bounds / slope constraints

Our main interest in transform-domain bound constraints originates from the special case of slope constraints, see e.g., Petersson and Sigmund (1998) and Bauschke and Koch (2015) for examples from computational design. Lelièvre and Oldenburg (2009) propose a transform-domain bound constraint in a geophysical context, but use interior point algorithms for implementation. In our context, slope means the model parameter variation per distance unit, over a predefined path in the model. For example, the slope of the 2D model parameters

Algorithm 3 ADMM to compute the projection, including automatic (heuristic) penalty parameter adjustment.

input: $\mathbf{m}$, transform-domain $\mathbf{A}$,

norm/bound/cardinality projector $\mathcal{P}_{\mathcal{C}}$

$x_0 = \mathbf{m}$, $\mathbf{z}_0 = 0$, $\mathbf{u}_0 = 0$, $k = 1$,

select $\tau > 1$, $\mu > 1$

WHILE not converged

$\mathbf{x}^{k+1} = (\rho\mathbf{A}^*\mathbf{A} + \mathbf{I})^{-1}\big(\rho\mathbf{A}^*(\mathbf{z}^k - \mathbf{u}^k) + \mathbf{m}\big)$

$\mathbf{z}^{k+1} = \mathcal{P}_{\mathcal{C}}(\mathbf{A}\mathbf{x}^{k+1} + \mathbf{u}^k)$

$\mathbf{u}^{k+1} = \mathbf{u}^k + \mathbf{A}\mathbf{x}^{k+1} - \mathbf{z}^{k+1}$

$\mathbf{r} = \mathbf{A}\mathbf{x}^{k+1} - \mathbf{z}^{k+1}$

$\mathbf{s} = \rho\mathbf{A}^*(\mathbf{z}^{k+1} - \mathbf{z}^k)$

IF $\|\mathbf{r}\| > \mu\|\mathbf{s}\|$ `//increase penalty`

$\rho = \rho\tau$

$\mathbf{u} = \mathbf{u}/\tau$

IF $\|\mathbf{s}\| > \mu\|\mathbf{r}\|$ `//decrease penalty`

$\rho = \rho/\tau$

$\mathbf{u} = \mathbf{u}\tau$

ELSE

ρ `//do nothing`

END

END

output: $\mathbf{x}$

in the vertical direction (z-direction) form the constraint set

$$\mathcal{C} \equiv \{\mathbf{m} \mid \mathbf{b}_j^l \leq ((\mathbf{D}_z \otimes \mathbf{I}_x)\mathbf{m})_j \leq \mathbf{b}_j^u\}, \tag{21}$$

with Kronecker product $\otimes$, identity matrix with dimension equal to the x-direction $\mathbf{I}_x$, and $\mathbf{D}_z$ is a 1D finite-difference matrix corresponding to the z-direction. $\mathbf{b}_j^l$ is element j of the lower bound vector. An appealing property of this constraint is the physical meaning in a pointwise sense. If the model parameters are acoustic velocity in meters per second and the grid is also in units of meters, the constraint then defines the maximum velocity increment/decrement per meter in a direction. This type of direct physical meaning of a constraint is not available for ℓ_1, rank or Nuclear norm constraints; those constraints assign a single scalar value to a property of the entire model.

There are different modes of operation of the slope constraint:

Approximate monotonicity. The acoustic velocity generally increases with depth inside the Earth. This means the parameter values increase (approximately) monotonically with dept (positivity of the vertical discrete gradient). The set $\mathcal{C} \equiv \{\mathbf{m} \mid -\varepsilon \leq ((\mathbf{D}_z \otimes \mathbf{I}_x)\mathbf{m})_j \leq +\infty\}$ describes this situation, where $\varepsilon > 0$ is a small number. Exact monotonicity corresponds to $\varepsilon = 0$, which means we allow the model parameter values to increase arbitrarily fast with increasing depth, but enforce a slow decrease of parameter values when looking into the depth direction.

Smoothness. We obtain a type of smoothness by setting both bounds to small numbers: $\mathcal{C} \equiv \{\mathbf{m} \mid -\varepsilon_1 \leq ((\mathbf{D}_z \otimes \mathbf{I}_x)\mathbf{m})_j \leq +\varepsilon_2\}$, where $\varepsilon_1 > 0$, $\varepsilon_2 > 0$ are small numbers. This type of smoothness results in a different projection problem than if smoothness is obtained using constraints based on norms or subspaces. Another difference is that the slope constraint is inherently locally defined.

The slope constraint may be defined along any path using any discrete derivative matrix. Higher order derivatives lead to bounds on different properties. Approximate monotonicity of parameter values can also be obtained using other constraints. Esser et al. (2016) use the norm based hinge-loss constraint. However, we prefer to work with linear inequalities because norm based constraints are not defined pointwise and do not have the direct physical interpretation as described above. Figure 9 shows what happens when we project a velocity model onto the different slope constraint sets.

[Figure 9 about here.]

REFERENCES

Akcelik, V., G. Biros, and O. Ghattas, 2002, Parallel multiscale gauss-newton-krylov methods for inverse wave propagation: Supercomputing, ACM/IEEE 2002 Conference, 41–41.

Anagaw, A. Y., 2014, Full waveform inversion using simultaneous encoded sources based on first-and second-order optimization methods: PhD thesis, University of Alberta.

Anagaw, A. Y., and M. D. Sacchi, 2017, Edge-preserving smoothing for simultaneous-source fwi model updates in high-contrast velocity models: GEOPHYSICS, **0**, 1–18.

Aravkin, A. Y., and T. van Leeuwen, 2012, Estimating nuisance parameters in inverse problems: Inverse Problems, **28**, 115016.

Asnaashari, A., R. Brossier, S. Garambois, F. Audebert, P. Thore, and J. Virieux, 2013, Regularized seismic full waveform inversion with prior model information: GEOPHYSICS, **78**, R25–R36.

Backus, G. E., 1988, Comparing hard and soft prior bounds in geophysical inverse problems: Geophysical Journal International, **94**, 249.

Barzilai, J., and J. M. Borwein, 1988, Two-point step size gradient methods: IMA Journal of Numerical Analysis, **8**, 141–148.

Baumstein, A., 2013, Pocs-based geophysical constraints in multi-parameter full wavefield inversion: Presented at the 75th EAGE Conference and Exhibition 2013, EAGE.

Bauschke, H. H., and P. L. Combettes, 2011, Convex analysis and monotone operator theory in hilbert spaces, 1st ed.: Springer Publishing Company, Incorporated.

Bauschke, H. H., and V. R. Koch, 2015, Projection methods: Swiss army knives for solving feasibility and best approximation problems with halfspaces: Contemporary Mathematics, **636**, 1–40.

Beck, A., 2014, Introduction to nonlinear optimization: Society for Industrial and Applied

Mathematics.

———, 2015, On the convergence of alternating minimization for convex programming with applications to iteratively reweighted least squares and decomposition schemes: SIAM Journal on Optimization, **25**, 185–209.

Becker, S., L. Horesh, A. Aravkin, E. van den Berg, and S. Zhuk, 2015, General optimization framework for robust and regularized 3d fwi: Presented at the 77th EAGE Conference and Exhibition 2015.

Bello, L., and M. Raydan, 2007, Convex constrained optimization for the seismic reflection tomography problem: Journal of Applied Geophysics, **62**, 158 – 166.

Bertsekas, D. P., 2015, Convex optimization algorithms: Athena Scientific.

Birgin, E. G., J. M. Martínez, and M. Raydan, 1999, Nonmonotone spectral projected gradient methods on convex sets: SIAM J. on Optimization, **10**, 1196–1211.

Birgin, E. G., J. M. Martínez, and M. Raydan, 2003, Inexact spectral projected gradient methods on convex sets: IMA Journal of Numerical Analysis, **23**, 539.

Birgin, E. G., and M. Raydan, 2005, Robust stopping criteria for dykstra's algorithm: SIAM Journal on Scientific Computing, **26**, 1405–1414.

Boyd, S., N. Parikh, E. Chu, B. Peleato, and J. Eckstein, 2011, Distributed optimization and statistical learning via the alternating direction method of multipliers: Found. Trends Mach. Learn., **3**, 1–122.

Boyd, S., and L. Vandenberghe, 2004, Convex optimization: Cambridge University Press.

Boyle, J. P., and R. L. Dykstra, 1986, *in* A Method for Finding Projections onto the Intersection of Convex Sets in Hilbert Spaces: Springer New York, 28–47.

Brenders, A. J., and R. G. Pratt, 2007, Full waveform tomography for lithospheric imaging: results from a blind test in a realistic crustal model: Geophysical Journal International,

168, 133–151.

Bunks, C., 1995, Multiscale seismic waveform inversion: Geophysics, **60**, 1457.

Burke, J., 1990, Basic convergence theory: Technical report, University of Washington.

Censor, Y., 2006, Computational acceleration of projection algorithms for the linear best approximation problem: Linear Algebra and its Applications, **416**, 111 – 123.

Combettes, P. L., and J.-C. Pesquet, 2011, Proximal splitting methods in signal processing, *in* Fixed-Point Algorithms for Inverse Problems in Science and Engineering: Springer New York, volume **49** *of* Springer Optimization and Its Applications, 185–212.

Da Silva, C., and F. J. Herrmann, 2017, A Unified 2D/3D Large Scale Software Environment for Nonlinear Inverse Problems: ArXiv e-prints.

Dai, Y., and L. Liao, 2002, R-linear convergence of the barzilai and borwein gradient method: IMA Journal of Numerical Analysis, **22**, 1.

Dattorro, J., 2010, Convex optimization & euclidean distance geometry: Meboo Publishing USA.

Dykstra, R. L., 1983, An algorithm for restricted least squares regression: Journal of the American Statistical Association, **78**, 837–842.

Escalante, R., and M. Raydan, 2011, Alternating projection methods: Society for Industrial and Applied Mathematics.

Esser, E., L. Guasch, F. J. Herrmann, and M. Warner, 2016, Constrained waveform inversion for automatic salt flooding: The Leading Edge, **35**, 235–239.

Esser, E., L. Guasch, T. van Leeuwen, A. Y. Aravkin, and F. J. Herrmann, 2015, Automatic salt delineation — wavefield reconstruction inversion with convex constraints: SEG Technical Program Expanded Abstracts 2015, 1337–1343.

———, 2018, Total variation regularization strategies in full-waveform inversion: SIAM

Journal on Imaging Sciences, **11**, 376–406.

Farquharson, C. G., and D. W. Oldenburg, 1998, Non-linear inversion using general measures of data misfit and model structure: Geophysical Journal International, **134**, 213.

———, 2004, A comparison of automatic techniques for estimating the regularization parameter in non-linear inverse problems: Geophysical Journal International, **156**, 411–425.

Grippo, L., and M. Sciandrone, 2002, Nonmonotone globalization techniques for the barzilai-borwein gradient method: Computational Optimization and Applications, **23**, 143–169.

Guitton, A., G. Ayeni, and E. Díaz, 2012, Constrained full-waveform inversion by model reparameterization: GEOPHYSICS, **77**, R117–R127.

Guitton, A., and E. Díaz, 2012, Attenuating crosstalk noise with simultaneous source full waveform inversion: Geophysical Prospecting, **60**, 759–768.

He, B. S., H. Yang, and S. L. Wang, 2000, Alternating direction method with self-adaptive penalty parameters for monotone variational inequalities: Journal of Optimization Theory and Applications, **106**, 337–356.

Herrmann, F., X. Li, A. Y. Aravkin, and T. Van Leeuwen, 2011, A modified, sparsity-promoting, gauss-newton algorithm for seismic waveform inversion: SPIE Optical Engineering+ Applications, International Society for Optics and Photonics, 81380V–81380V.

Jervis, M., M. K. Sen, and P. L. Stoffa, 1996, Prestack migration velocity estimation using nonlinear methods: GEOPHYSICS, **61**, 138–150.

Kennett, B. L. N., and P. R. Williamson, 1988, *in* Subspace methods for large-scale nonlinear inversion: Springer Netherlands, 139–154.

Kleinman, R., and P. den Berg, 1992, A modified gradient method for two- dimensional problems in tomography: Journal of Computational and Applied Mathematics, **42**, 17 – 35.

Lee, J. D., Y. Sun, and M. A. Saunders, 2014, Proximal newton-type methods for minimizing composite functions: SIAM Journal on Optimization, **24**, 1420–1443.

Lelièvre, P. G., and D. W. Oldenburg, 2009, A comprehensive study of including structural orientation information in geophysical inversions: Geophysical Journal International, **178**, 623.

Li, M., O. Semerci, and A. Abubakar, 2013, A contrast source inversion method in the wavelet domain: Inverse Problems, **29**, 025015.

Li, X., A. Y. Aravkin, T. van Leeuwen, and F. J. Herrmann, 2012, Fast randomized full-waveform inversion with compressive sensing: GEOPHYSICS, **77**, A13–A17.

Li, X., E. Esser, and F. J. Herrmann, 2016, Modified Gauss-Newton full-waveform inversion explained–-why sparsity-promoting updates do matter: Geophysics, **81**, R125–R138.

Lin, Y., and L. Huang, 2015, Acoustic- and elastic-waveform inversion using a modified total-variation regularization scheme: Geophysical Journal International, **200**, 489–502.

Louboutin, M., P. Witte, M. Lange, N. Kukreja, F. Luporini, G. Gorman, and F. J. Herrmann, 2017, Full-waveform inversion, part 1: Forward modeling: The Leading Edge, **36**, 1033–1036.

Mueller, J., and S. Siltanen, 2012, Linear and nonlinear inverse problems with practical applications: Society for Industrial and Applied Mathematics.

Métivier, L., and R. Brossier, 2016, The seiscope optimization toolbox: A large-scale nonlinear optimization library based on reverse communication: GEOPHYSICS, **81**, F1–F15.

Nishihara, R., L. Lessard, B. Recht, A. Packard, and M. I. Jordan, 2015, A general analysis of the convergence of admm.: Int. Conf. Mach. Learn., 343–352.

Nocedal, J., and S. J. Wright, 2000, Numerical optimization: Springer.

Oldenburg, D. W., P. R. McGillivray, and R. G. Ellis, 1993, Generalized subspace methods

for large-scale inverse problems: Geophysical Journal International, **114**, 12.

Paige, C. C., and M. A. Saunders, 1982, Lsqr: An algorithm for sparse linear equations and sparse least squares: ACM Trans. Math. Softw., **8**, 43–71.

Parikh, N., and S. Boyd, 2014, Proximal algorithms: Foundations and Trends® in Optimization, **1**, 127–239.

Peters, B., and F. J. Herrmann, 2017, Constraints versus penalties for edge-preserving full-waveform inversion: The Leading Edge, **36**, 94–100.

Petersson, J., and O. Sigmund, 1998, Slope constrained topology optimization: International Journal for Numerical Methods in Engineering, **41**, 1417–1434.

Pratt, R. G., 1999, Seismic waveform inversion in the frequency domain, part 1: Theory and verification in a physical scale model: GEOPHYSICS, **64**, 888–901.

Qiu, L., N. Chemingui, Z. Zou, and A. Valenciano, 2016, Full-waveform inversion with steerable variation regularization: SEG Technical Program Expanded Abstracts 2016, 1174–1178.

Raydan, M., 1993, On the barzilai and borwein choice of steplength for the gradient method: IMA Journal of Numerical Analysis, **13**, 321–326.

Ryu, E. K., and S. Boyd, 2016, Primer on monotone operator methods: Appl. Comput. Math, **15**, 3–43.

Scales, J. A., and R. Snieder, 1997, To bayes or not to bayes?: GEOPHYSICS, **62**, 1045–1046.

Schmidt, M., D. Kim, and S. Sra, 2012, Projected newton-type methods in machine learning, *in* Optimization for Machine Learning: MIT Press, **35**, 11, 305–327.

Schmidt, M., and K. Murphy, 2010, Convex structure learning in log-linear models: Beyond pairwise potentials: Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics, PMLR, 709–716.

Schmidt, M., E. Van Den Berg, M. P. Friedlander, and K. Murphy, 2009, Optimizing costly functions with simple constraints: A limited-memory projected quasi-newton algorithm: Presented at the Proc. of Conf. on Artificial Intelligence and Statistics.

Sen, M., and I. Roy, 2003, Computation of differential seismograms and iteration adaptive regularization in prestack waveform inversion: GEOPHYSICS, **68**, 2026–2039.

Shen, P., and W. W. Symes, 2008, Automatic velocity analysis via shot profile migration: GEOPHYSICS, **73**, VE49–VE59.

Shen, P., W. W. Symes, and C. C. Stolk, 2005, Differential semblance velocity analysis by wave-equation migration: SEG Technical Program Expanded Abstracts 2003, 2132–2135.

Smithyman, B., B. Peters, and F. Herrmann, 2015, Constrained waveform inversion of colocated vsp and surface seismic data: Presented at the 77th EAGE Conference and Exhibition 2015.

Stark, P. B., 2015, Constraints versus priors: SIAM/ASA Journal on Uncertainty Quantification, **3**, 586–598.

Vogel, C., 2002, Computational methods for inverse problems: Society for Industrial and Applied Mathematics.

Xu, Z., S. De, M. Figueiredo, C. Studer, and T. Goldstein, 2016, An empirical study of admm for nonconvex problems: Presented at the NIPS workshop on nonconvex optimization.

Xu, Z., M. Figueiredo, and T. Goldstein, 2017, Adaptive ADMM with Spectral Penalty Parameter Selection: Proceedings of the 20th International Conference on Artificial Intelligence and Statistics, PMLR, 718–727.

Xue, Z., and H. Zhu, 2015, Full waveform inversion with sparsity constraint in seislet domain: SEG Technical Program Expanded Abstracts 2015, 1382–1387.

Zeev, N., O. Savasta, and D. Cores, 2006, Non-monotone spectral projected gradient method

applied to full waveform inversion: Geophysical Prospecting, **54**, 525–534.

Zhdanov, M. S., 2002, Geophysical inverse theory and regularization problems: Elsevier, **36**.

Zhu, L., E. Liu, and J. H. McClellan, 2017, Sparse-promoting full-waveform inversion based on online orthonormal dictionary learning: GEOPHYSICS, **82**, R87–R107.

LIST OF FIGURES

1 The trajectory of Dykstra's algorithm for a toy example with two constraints: a maximum 2-norm constraint (disk) and bound constraints. The feasible set is the intersection of a halfspace and a disk. The circle and horizontal lines are the boundaries of the sets. The difference between the two figures is the ordering of the two sets. The projection onto convex sets (POCS) algorithm converges to different points, depending onto which set we project first. In both cases, the points found by POCS are not the projection onto the intersection. Dykstra's algorithm converges to the projection of the initial point onto the intersection in both cases, as expected. 49

2 The Marmousi model, the projection onto an intersection of bound constraints and total-variation constraints found with Dykstra's algorithm and two feasible models found by the POCS algorithm. We observe that one of the POCS results is very similar to the projection, but the other result is very different. The different model (POCS 2) has a total-variation much smaller than requested. This situation is analogous to Figure 1. 50

3 FWI with an incorrect source function with projections (with Dykstra) and FWI with two feasible points (with POCS) for various TV-balls (as a percentage of the TV of the true model) and bound constraints. Also shows differences (rightmost two columns) between results. The results show that using POCS inside a projected gradient algorithm instead of the projection leads to different results that also depend on the order in which we provide the sets to POCS. This example illustrates the differences between the methods and it is not the intention to obtain excellent FWI results. 51

4 Example of the iteration trajectory when using gradient descent to minimize a non-convex function and projected gradient descent to minimize a non-convex function subject to a constraint. The constraint requires the model estimate to inside the circular area. The semi-transparent area outside the circle is not accessible by projected gradient descent. There are two important observations: 1) The constrained minimization converges to a different (local) minimizer. 2) The intermediate projected gradient parameter estimates can be in the interior of the set or on the boundary. Black represents low values of the function. 52

5 The 3-level nested constrained optimization workflow. 53

6 True and initial velocity models for the example. 54

7 Model estimate obtained by FWI with bound constraints only. 55

8 a) Model estimate obtained by FWI with bound constraints, a vertical slope constraint and a constraint on the velocity variation per meter in the horizontal direction. b) Model estimate by FWI with bound constraints and using the result from a) as starting model. 56

9 Different uses of transform domain bound constraints. Figures show the projection of a velocity model onto two different slope constraint sets. The middle panel shows the effect of allowing arbitrary velocity increase with depth, but only slow velocity decrease with depth. The bottom panel shows lateral smoothness, by bounding the upper and lower limit on the velocity change per distance interval in the lateral direction. 57

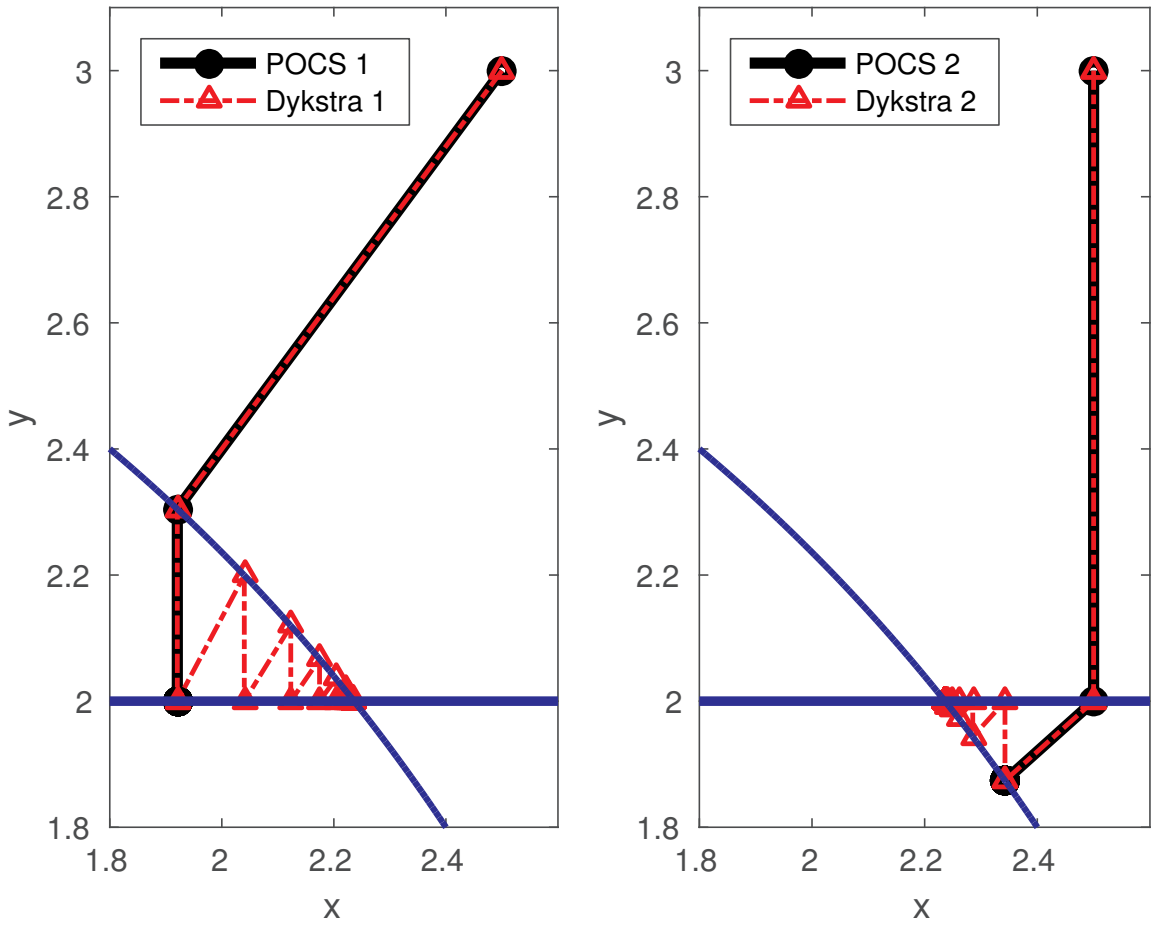


Figure 1: The trajectory of Dykstra's algorithm for a toy example with two constraints: a maximum 2-norm constraint (disk) and bound constraints. The feasible set is the intersection of a halfspace and a disk. The circle and horizontal lines are the boundaries of the sets. The difference between the two figures is the ordering of the two sets. The projection onto convex sets (POCS) algorithm converges to different points, depending onto which set we project first. In both cases, the points found by POCS are not the projection onto the intersection. Dykstra's algorithm converges to the projection of the initial point onto the intersection in both cases, as expected.

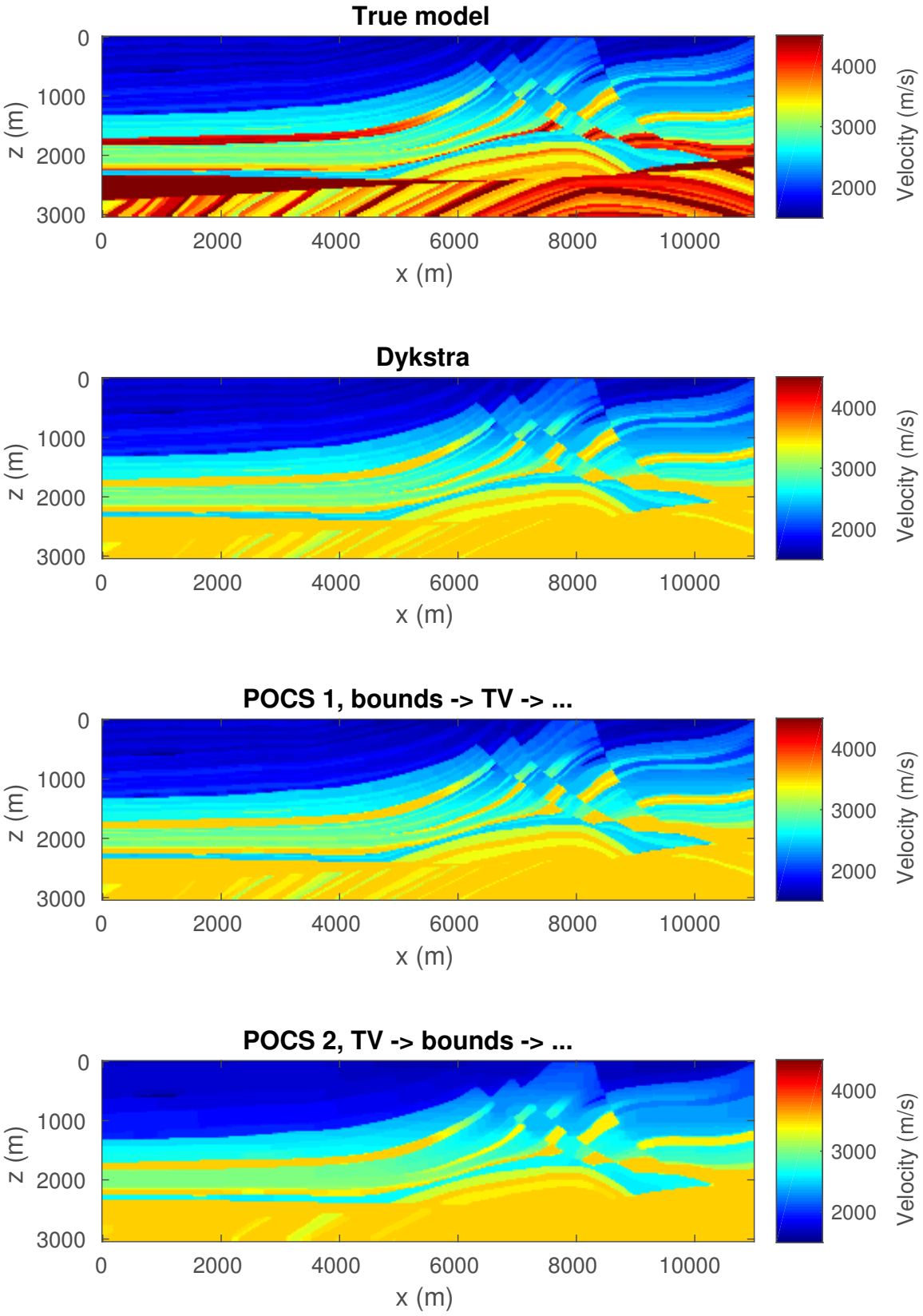


Figure 2: The Marmousi model, the projection onto an intersection of bound constraints and total-variation constraints found with Dykstra's algorithm and two feasible models found by the POCS algorithm. We observe that one of the POCS results is very similar to the projection, but the other result is very different. The different model (POCS 2) has a total-variation much smaller than requested. This situation is analogous to Figure 1.

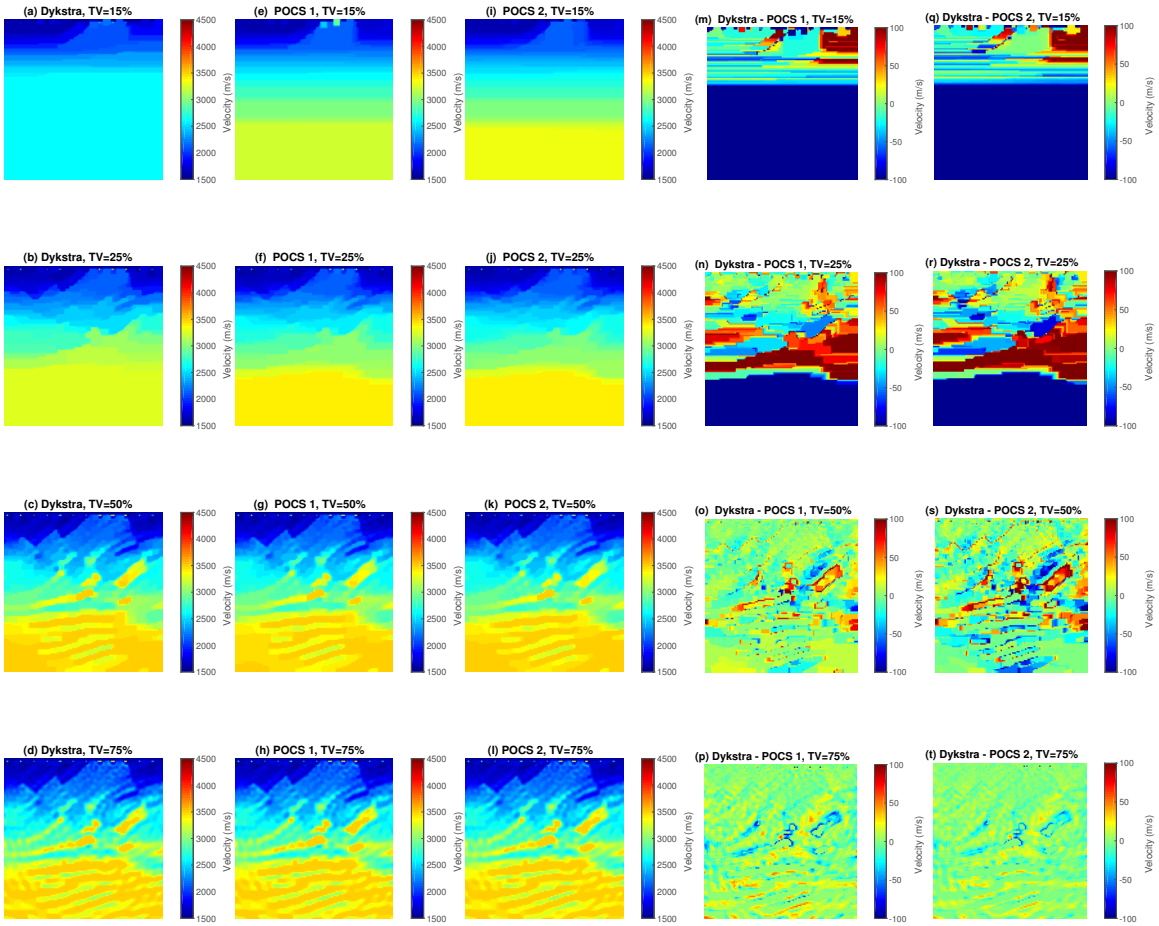
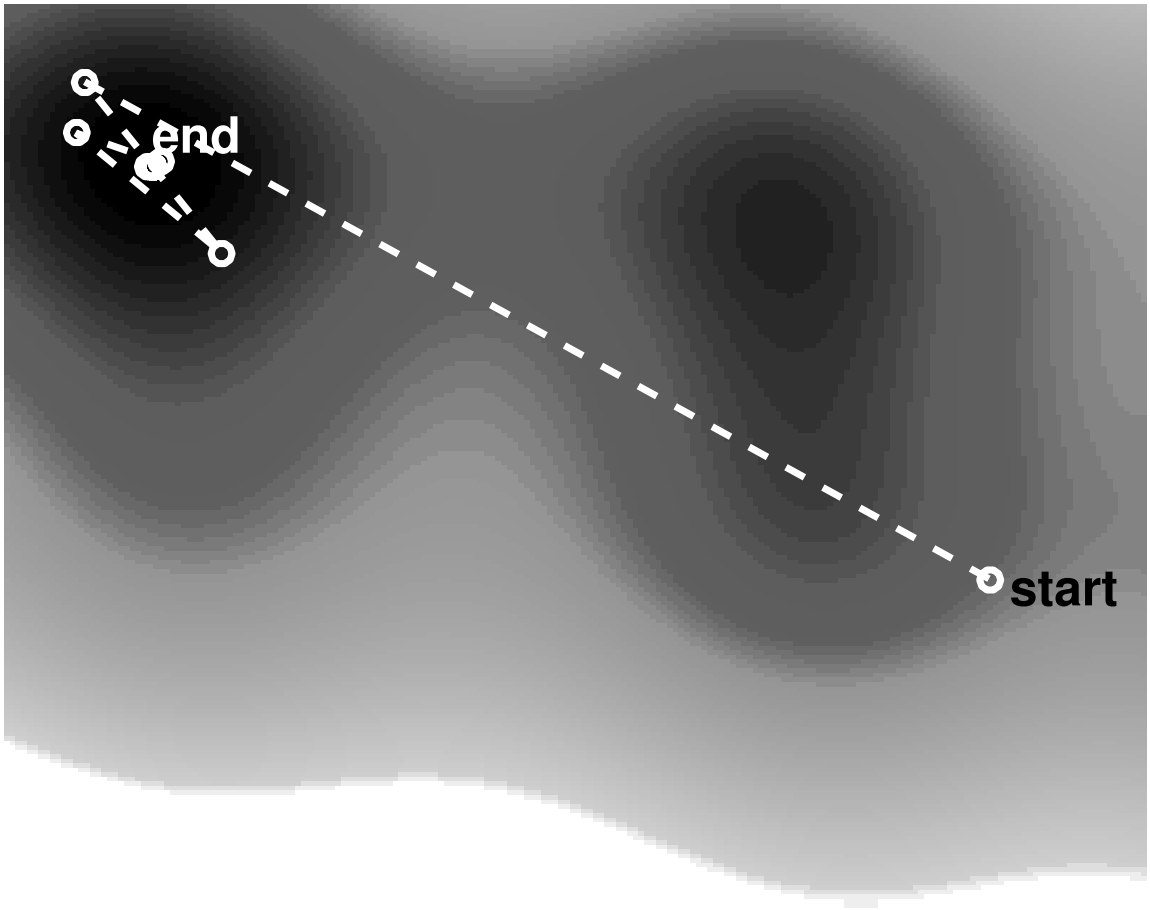
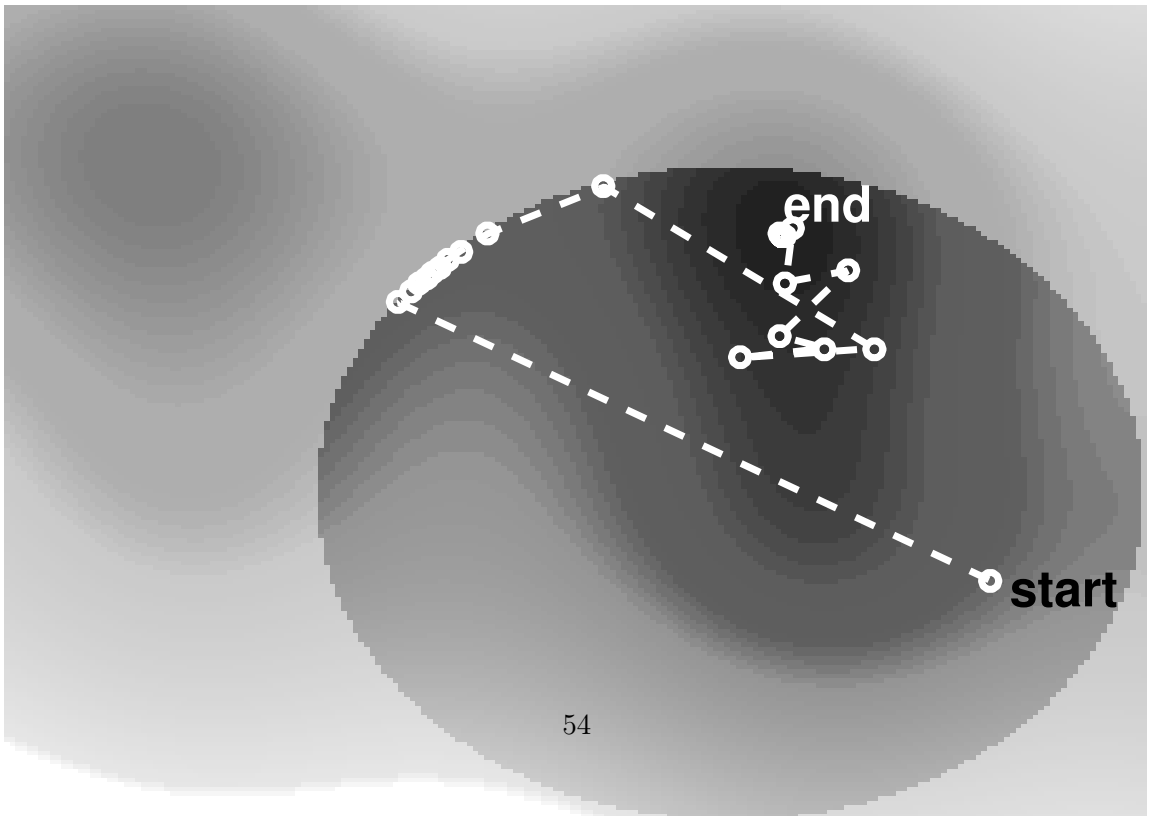


Figure 3: FWI with an incorrect source function with projections (with Dykstra) and FWI with two feasible points (with POCS) for various TV-balls (as a percentage of the TV of the true model) and bound constraints. Also shows differences (rightmost two columns) between results. The results show that using POCS inside a projected gradient algorithm instead of the projection leads to different results that also depend on the order in which we provide the sets to POCS. This example illustrates the differences between the methods and it is not the intention to obtain excellent FWI results.

Gradient decent



Projected gradient decent



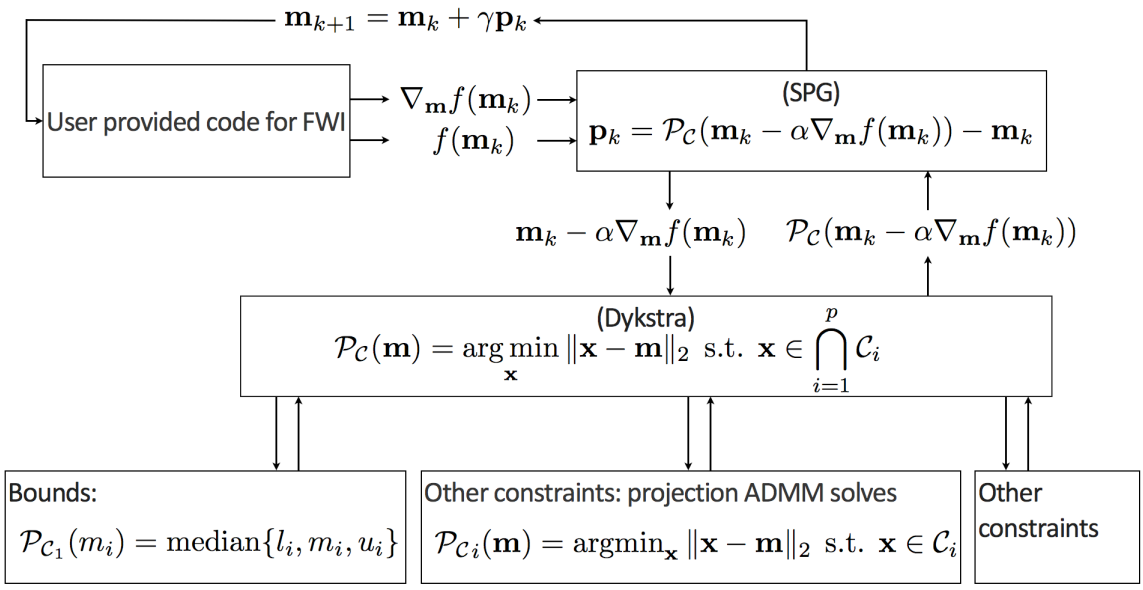


Figure 5: The 3-level nested constrained optimization workflow.

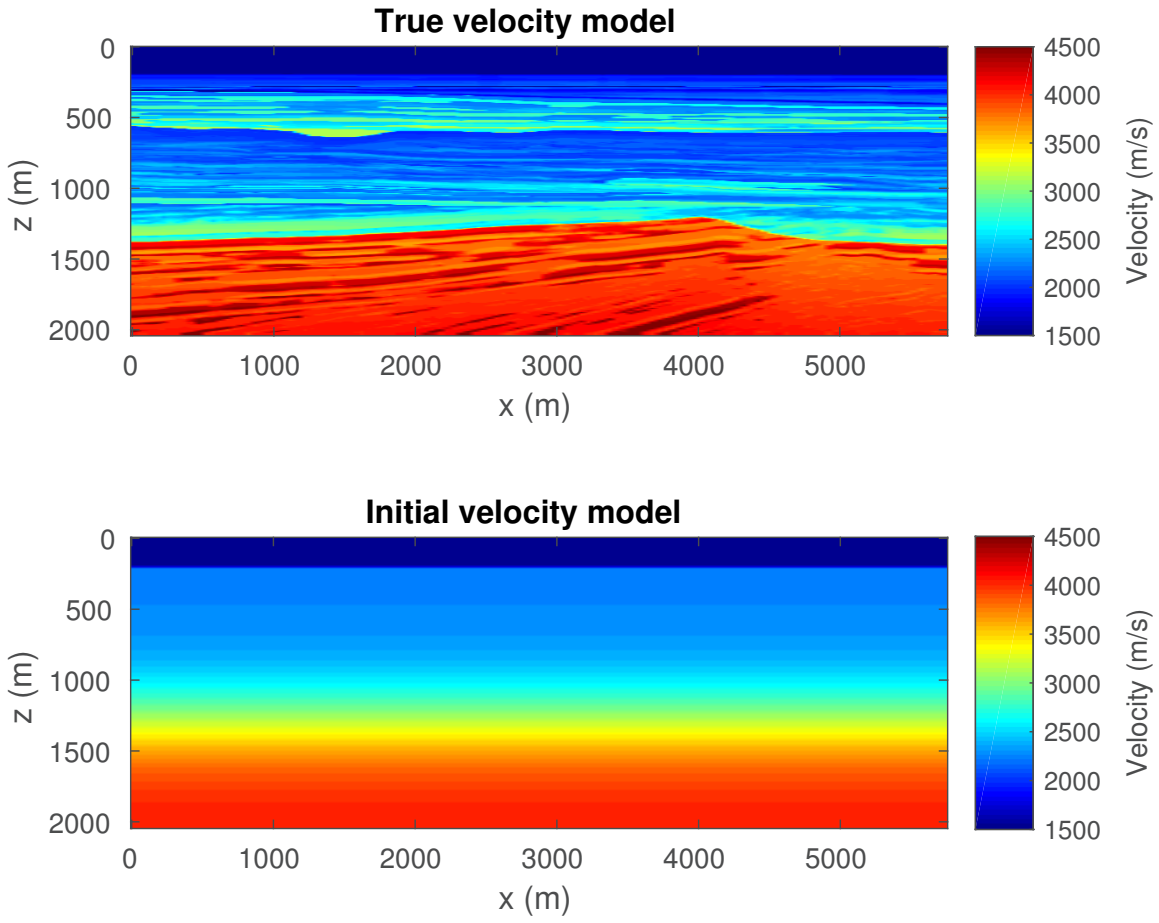


Figure 6: True and initial velocity models for the example.

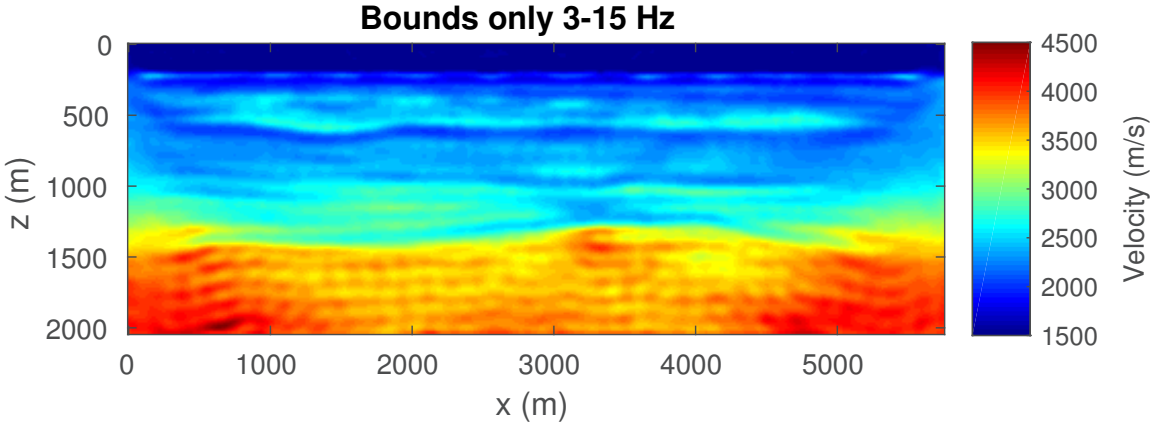


Figure 7: Model estimate obtained by FWI with bound constraints only.

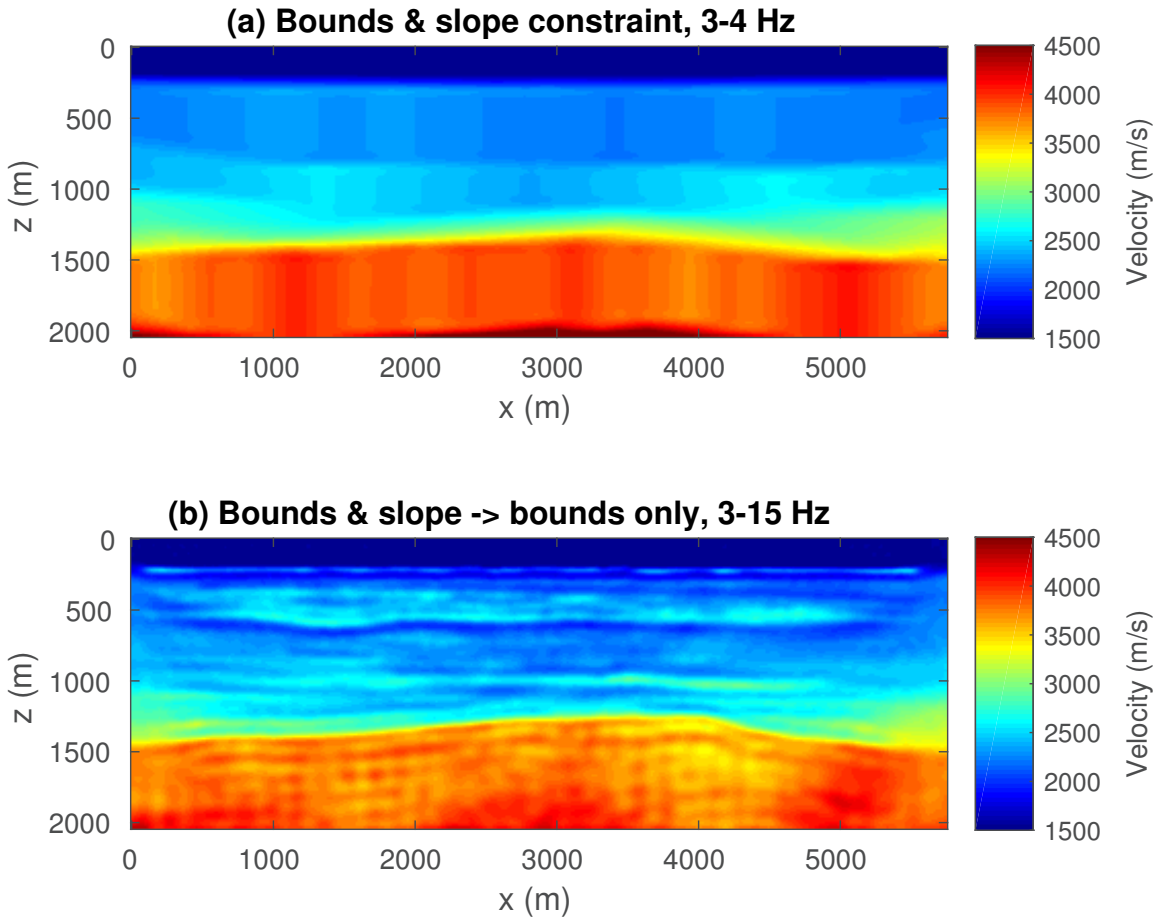


Figure 8: a) Model estimate obtained by FWI with bound constraints, a vertical slope constraint and a constraint on the velocity variation per meter in the horizontal direction. b) Model estimate by FWI with bound constraints and using the result from a) as starting model.

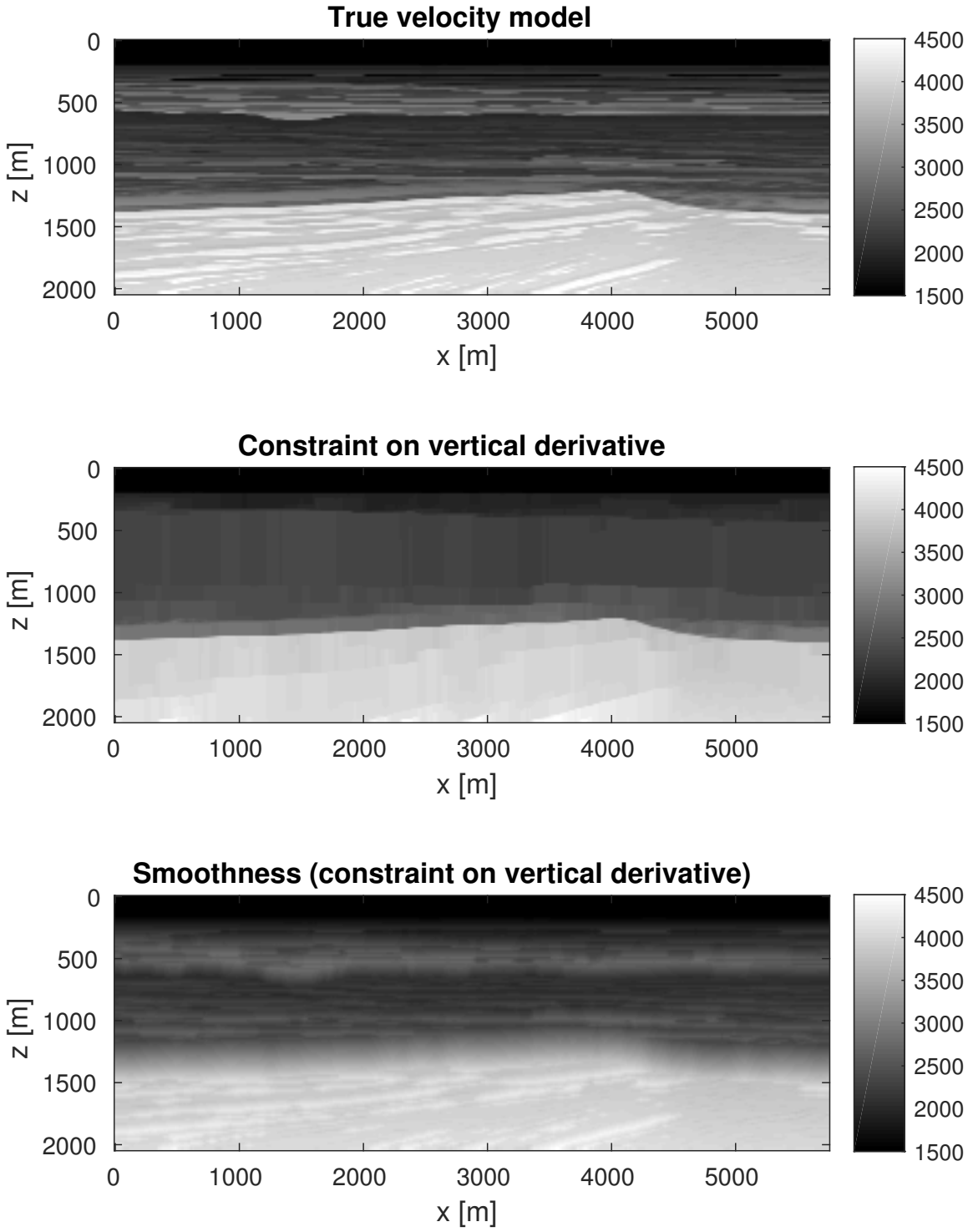


Figure 9: Different uses of transform domain bound constraints. Figures show the projection of a velocity model onto two different slope constraint sets. The middle panel shows the effect of allowing arbitrary velocity increase with depth, but only slow velocity decrease with depth. The bottom panel shows lateral smoothness, by bounding the upper and lower limit on the velocity change per distance interval in the lateral direction.

LIST OF TABLES

1 Notation used in this paper. 59

description	symbol
data-misfit	$f(\mathbf{m})$
gradient w.r.t. medium parameters	$\nabla_{\mathbf{m}} f(\mathbf{m})$
set (convex or non-convex)	$\mathcal{C}$
intersection of sets	$\bigcap_{i=1}^{p} \mathcal{C}_i$
any transform-domain operator	$\mathbf{A} \in \mathbb{C}^{M \times N}$
discrete derivative matrix in 1D	$\mathbf{D}_z$ or $\mathbf{D}_x$
cardinality (number of nonzero entries) or ℓ_0 'norm'	$\mathbf{card}(\cdot) \Leftrightarrow \|\cdot\|_0$
ℓ_1 norm (one-norm)	$\|\cdot\|_1$

Table 1: Notation used in this paper.