
# Inexact Gradient Methods

A STRAND OF EFFICIENT BILEVEL OPTIMISATION FOR IMAGING
IRP INDIVIDUAL REPORT

**Students:** Matthew Pawley

**Supervisors:** Silvia Gazzola
Matthias Ehrhardt

April 26, 2021

# Contents

# 1 Introduction

Suppose we have access to a labelled training set $\{(y_i, u_i^\star)\}_{i=1}^N$ data $y_i$ and the corresponding ground truths $u_i^\star$. The optimal model parameters $\theta \in \Theta$ are obtained by solving the bilevel problem,

$$\min_{\theta\in\Theta} \frac{1}{N}\sum_{i=1}^{N} L_{u_i^\star}(\hat{u}_i(\theta)) \qquad \text{subject to} \qquad \hat{u}_i(\theta) = \arg\min_{u\in\mathcal{U}} E_i(u;\theta), \tag{1}$$

where $E_i(u;\theta)$ is the lower level objective function and the upper level objective comprises cost functions

$$L_{u_i^\star}(\hat{u}_i(\theta)) = \frac{1}{2}\left\|\hat{u}_i(\theta) - u_i^\star\right\|^2, \tag{2}$$

which measure the discrepancy between the reconstruction $\hat{u}_i(\theta)$ and the ground truth $u_i^\star$. The goal is to find the parameter $\theta$ that generates the best reconstructions averaged over the training set. For the sake of notational simplicity, in this report we take $N = 1$ and omit the dependence on $i$.

The number of parameters of interest with the variational model is potentially very large, which renders procedures such as grid search infeasible due to the curse of dimensionality. Instead, the optimisation is typically performed using gradient-based algorithms such as gradient descent (GD) (Chambolle and Pock 2016). However, the nested structure of the bilevel problem creates some challenges in this respect. In particular, the gradient of the upper level objective (2) is given by (Sherry et al. 2020)

$$\nabla_\theta L_{u^\star}(\hat{u}(\theta)) = -D_{\theta,u}E(\hat{u}(\theta))[D_u^2 E(\hat{u}(\theta))]^{-1}\nabla_{\hat{u}(\theta)} L_{u^\star}(\hat{u}(\theta))^\top, \tag{3}$$

which depends on the lower level solution and the inverse of the Hessian of the lower level problem, neither of which are computed exactly in practice. The viability of gradient-based methods in this setting is therefore dubious.

There are two pathways that could be taken at this juncture. First, we could avoid gradient methods altogether. This avenue is explored by Ehrhardt and Roberts (2020), who use derivative-free methods, which do not require access to the gradient, in order to solve the upper level problem. Such methods are efficient in low-dimensional problems ($< 100$ dimensions), but scale poorly to much higher dimensions, where the convergence rate tends to be slow or the per-iteration computational cost prohibitive (Qian, Hu, and Yu 2016). The second approach - to be explored this section - is to use inexact gradient methods, i.e. gradient methods that permit inexact gradient computation. This approach is better suited to large-scale problems, and will improve the efficiency of the bilevel optimisation algorithm by reducing the computational effort expended in solving the lower level problem and the linear system, particularly in the early stages of the algorithm when even an erroneous gradient is likely to be sufficient to

achieve descent.

This section is organised as follows: Section 2 outlines how the upper level gradient is approximated and develops a notational framework for keeping track of the errors that occur at each stage. Section 3 establishes an *a posteriori* bound for the error in the gradient. The proposed algorithm for bilevel learning with inexact gradients is described in Section 4. Empirical experiments based an the example problem (learning the TV regularisation parameter) are performed in Section 5. Finally, Section 6 concludes with suggestions for future research.

The code implementing our approach and performing the experimentation that follows is available at `https://github.com/jip30/Efficient-Bilevel-Optimisation-for-Imaging.git`.

## 2 Approximating the upper level gradient

The lower level problem is concerned with finding the optimal reconstruction of the data $y$ for a variational regularisation model with a given parameter $\theta$:

$$\hat{u}(\theta) = \arg\min_{u \in \mathcal{U}} E(u; \theta). \tag{4}$$

We seek to minimise this function using a gradient-based optimisation technique. We choose a simple first-order, gradient descent method, which is known to perform well on a wide class of high-dimensional optimisation problems (Chambolle and Pock 2016). In order to control the accuracy of the lower level solution, we impose some assumptions on the smoothness and convexity of the lower level objective.

**Definition 2.1** ($M$-smooth)**.** A function $f : \mathbb{R}^n \to \mathbb{R}$ is $M$-smooth if it is differentiable and its derivative is Lipschitz continuous with constant $M \geq 0$, i.e. for all $x, y \in \mathbb{R}^n$,

$$\|\nabla f(x) - \nabla f(y)\| \leq M \|x - y\|.$$

**Definition 2.2** ($\mu$-convex)**.** A function $f : \mathbb{R}^n \to \mathbb{R}$ is $\mu$-convex ($\mu > 0$) if $f - \frac{\mu}{2}\|\cdot\|^2$ is convex.

We assume that $E$ is $M$-smooth and $\mu$-convex. Under these assumptions, the gradient descent (GD) algorithm

$$u^{(k+1)} = u^{(k)} - \tau_k \nabla E(u^{(k)}; \theta)$$

with fixed step-size $\tau_k = 2/(M+\mu)$ converges linearly to the unique minimiser $\hat{u}$ of $E$ (Ehrhardt and Roberts 2020):

$$\left\|u^{(k)} - \hat{u}\right\|^2 \leq \left(1 - \frac{\mu}{M}\right)^{2k} \left\|u^{(0)} - \hat{u}\right\|.$$

By performing sufficiently many GD iterations, the lower level problem can be solved to an

arbitrary level of accuracy. In particular, for any $\delta > 0$, the accuracy $\|u^{(k)} - \hat{u}\|^2 \leq \delta$ is guaranteed by terminating GD when the stopping criterion

$$\left\|\nabla E(u^{(k)})\right\|^2 \leq \delta\mu^2$$

is satisfied (Ehrhardt and Roberts 2020). We will denote by $\tilde{u}_\delta$ the approximate solution to the lower level problem (4) that satisfies

$$\|\tilde{u}_\delta(\theta) - \hat{u}(\theta)\| \leq \delta.$$

For brevity, we rewrite the upper level gradient (3) in a more compact form,

$$\nabla L_{u^\star}(\hat{u}(\theta)) = \Phi(\theta)H^{-1}(\theta)b(\theta),$$

where

$$\Phi(\theta) = -D_{\theta,u}E(\hat{u}(\theta);\theta), \qquad H(\theta) = D_u^2 E(\hat{u}(\theta);\theta), \qquad b(\theta) = \nabla_{\hat{u}(\theta)} L_{u^\star}(\hat{u}(\theta))^\top. \tag{5}$$

Defining

$$w(\theta) = H^{-1}(\theta)b(\theta),$$

as the solution of the linear system, the (exact) gradient is given by $\nabla L_{u^\star}(\hat{u}(\theta)) = \Phi(\theta)w(\theta)$.

Carrying forward this notation, we seek approximations for $\Phi$ and $w$. Let $\boldsymbol{\delta} = (\delta_1, \delta_2)$ for arbitrary $\delta_1, \delta_2 > 0$. Suppose we solve the lower level problem to accuracy $\delta_1$, i.e. find the approximate solution $\tilde{u}_{\delta_1}(\theta)$. Approximations for $\Phi$, $H$ and $b$ (denoted $\tilde{\Phi}$, $\tilde{H}$ and $\tilde{b}$, respectively) are obtained by substituting the approximate lower level solution in place of the exact solution within the expressions in (5):

$$\tilde{\Phi}_{\delta_1}(\theta) = -D_{\theta,u}E(\tilde{u}_{\delta_1}(\theta);\theta), \qquad \tilde{H}_{\delta_1}(\theta) = D_u^2 E(\tilde{u}_{\delta_1}(\theta);\theta), \qquad \tilde{b}_{\delta_1}(\theta) = \nabla_{\tilde{u}_{\delta_1}(\theta)} L_{u^\star}(\tilde{u}_{\delta_1}(\theta))^\top.$$

The approximation of $w$ depends on both $\delta_1$ and $\delta_2$, because it represents the approximate solution of a linear system which itself is constructed from the approximate lower level solution. We define $\tilde{w}_{\boldsymbol{\delta}}(\theta)$ as the vector obtained by solving the lower level problem to accuracy $\delta_1$ and the resulting linear system to accuracy $\delta_2$ (in terms of residual norm), so that $\tilde{w}_{\boldsymbol{\delta}}(\theta)$ satisfies

$$\left\|\tilde{H}_{\delta_1}(\theta)\tilde{w}_{\boldsymbol{\delta}}(\theta) - \tilde{b}_{\delta_1}(\theta)\right\| \leq \delta_2.$$

The approximation of the upper level gradient is denoted

$$\tilde{\nabla}^{(\boldsymbol{\delta})} L_{u^\star}(\hat{u}(\theta)) = \tilde{\Phi}_{\delta_1}(\theta)\tilde{w}_{\boldsymbol{\delta}}(\theta).$$

# 3 Estimating the error in the upper level gradient

When the errors in solving the lower level problem and linear system are accounted for, the error in the upper level gradient can be written

$$e_{\boldsymbol{\delta}}(\theta) = \nabla L_{u^\star}(\hat{u}(\theta)) - \tilde{\nabla}^{(\boldsymbol{\delta})} L_{u^\star}(\hat{u}(\theta)) = \Phi(\theta) w(\theta) - \tilde{\Phi}_{\delta_1}(\theta)\tilde{w}_{\boldsymbol{\delta}}(\theta).$$

Of course, this error is unobservable since the exact gradient is unknown. Instead, we find an estimate (bound) for the error as a function of $\boldsymbol{\delta}$ and $\theta$.

**Lemma 3.1.** *For any $\boldsymbol{\delta} = (\delta_1, \delta_2)$,*

$$\|e_{\boldsymbol{\delta}}(\theta)\| \leq \|H^{-1}\| \left( \|\Phi - \tilde{\Phi}_{\delta_1}\| + \|\tilde{\Phi}_{\delta_1}\| \right) \left( \delta_1 + \delta_2 + \|H - \tilde{H}_{\delta_1}\| \|\tilde{w}_{\boldsymbol{\delta}}\| \right) + \|\tilde{w}_{\boldsymbol{\delta}}\| \|\Phi - \tilde{\Phi}_{\delta_1}\|. \quad (6)$$

*where the dependence on $\theta$ is omitted on the right-hand side for brevity. Assume that $\Phi$ and $H$ (considered as functions of $u$) are Lipschitz continuous with constants $M_\Phi$ and $M_H$ respectively, and that there exists a constant $C_H > 0$ such that $\|H^{-1}\| \leq C_H$. Then*

$$\|e_{\boldsymbol{\delta}}(\theta)\| \leq C_H \left( M_\Phi \delta_1 + \|\tilde{\Phi}_{\delta_1}\| \right) (\delta_1 + \delta_2 + M_H \delta_1 \|\tilde{w}_{\boldsymbol{\delta}}\|) + M_\Phi \delta_1 \|\tilde{w}_{\boldsymbol{\delta}}\|. \quad (7)$$

*Proof.* For brevity, dependencies on $\theta$ and $\boldsymbol{\delta}$ are omitted. By repeated application of the triangle inequality,

$$\begin{aligned}
\|e\| &= \|\Phi w - \tilde{\Phi}\tilde{w}\| \\
&\leq \|\Phi\| \|w - \tilde{w}\| + \|\tilde{w}\| \|\Phi - \tilde{\Phi}\| \\
&\leq \left( \|\Phi - \tilde{\Phi}\| + \|\tilde{\Phi}\| \right) \|w - \tilde{w}\| + \|\tilde{w}\| \|\Phi - \tilde{\Phi}\|.
\end{aligned}$$

The exact solution $w$ of the (exact) linear system is unknown, so further work is required:

$$\begin{aligned}
\|w - \tilde{w}\| &= \|H^{-1}(Hw - H\tilde{w})\| \\
&\leq \|H^{-1}\| \|b - H\tilde{w}\| \\
&\leq \|H^{-1}\| \left( \|b - \tilde{b}\| + \|\tilde{b} - H\tilde{w}\| \right) \\
&\leq \|H^{-1}\| \left( \|b - \tilde{b}\| + \|\tilde{b} - \tilde{H}\tilde{w}\| + \|\tilde{H}\tilde{w} - H\tilde{w}\| \right) \\
&\leq \|H^{-1}\| \left( \|b - \tilde{b}\| + \delta_2 + \|\tilde{H} - H\| \|\tilde{w}\| \right) \\
&\leq \|H^{-1}\| \left( \delta_1 + \delta_2 + \|\tilde{H} - H\| \|\tilde{w}\| \right),
\end{aligned}$$

where the last line follows from the fact that $b = \hat{u} - u^\star$ and so

$$\|b - \tilde{b}\| = \|(\hat{u} - u^\star) - (\tilde{u} - u^\star)\| = \|\hat{u} - \tilde{u}\| \leq \delta_1.$$

Combining these results yields (6). For (7), we simply note that (by assumption)

$$\|\Phi - \tilde{\Phi}\| \leq M_\Phi \|\hat{u} - \tilde{u}\| \leq M_\Phi \delta_1,$$
$$\|H - \tilde{H}\| \leq M_H \|\hat{u} - \tilde{u}\| \leq M_H \delta_1. \qquad \square$$

The estimate (7) provides an *a posteriori* bound for the error in the gradient comprising solely computable terms. That is, for any $\boldsymbol{\delta}$ and $\theta$, an inexact gradient $\tilde{\nabla}^{(\boldsymbol{\delta})} L_{u^\star}(\hat{u}(\theta))$ can be computed and its error estimated. A limitation of the *a posteriori* bound is that it depends on the quantities $\|\tilde{\Phi}_{\delta_1}\|$ and $\|\tilde{w}_{\boldsymbol{\delta}}\|$, which are unknown until we solve the lower level problem and linear system to $\boldsymbol{\delta}$ accuracy. Thus, we can only know that superfluous computations have been performed after the event. Nevertheless, the *a posteriori* bound can be used to improve computational efficiency.

# 4 Algorithm for bilevel optimisation with inexact gradients

The rationale behind our approach is as follows: starting with conservatively chosen (i.e. large) values of $\delta_1$ and $\delta_2$, we can obtain a cheap estimate of the gradient from relatively few iterations of GD and the linear solver; if the error, as estimated by (7), is larger than some specified level $\mathcal{E}$, then reduce $\delta_1$ and/or $\delta_2$ and perform further iterations; repeat this process until the accuracy of the gradient reaches the desired level and perform the upper level GD update in the direction of the inexact gradient. This procedure is formalised in Algorithm 1.

The multi-index of errors $\boldsymbol{\delta}$ is determined by a two-way backtracking scheme. If the desired error $\mathcal{E}_k$ has not been attained, then $\boldsymbol{\delta}^{(k)}$ is rescaled by a factor of $\beta_1 < 1$; this process is repeated until the error becomes sufficiently sufficiently small. Once the error $\mathcal{E}_k$ is reached, the GD update is performed and we take $\beta_2 \boldsymbol{\delta}^{(k)}$ as the starting $\boldsymbol{\delta}$ for the $(k+1)$th iteration, where $\beta_2 > 1$. This approach is likely to lead to efficient choices of $\boldsymbol{\delta}$: the $\boldsymbol{\delta}$ required in one iteration is likely to be close to that which is required in the next iteration (assuming that the error sequence $\{\mathcal{E}_k\}$ does not vary significantly from term to term), but starting the next iteration with a slightly more conservative $\boldsymbol{\delta}$ affords an opportunity to cut superfluous computations where possible. This approach could be extended further by specifying separate $\beta_1$ and $\beta_2$ values for $\delta_1$ and $\delta_2$. In this report, we do not concern ourselves with how best to choose these values, setting $\beta_1 = 0.9$ and $\beta_2 = 10$ throughout. Choosing $\beta_1 \approx 1$ means that the gradient error criterion is checked regularly, and choosing $\beta_2 \gg 1$ means that $\boldsymbol{\delta}$ is chosen conservatively at the start of each upper level iteration. Both of these factors contribute to reducing the number of superfluous computations.

**Algorithm 1:** Parameter selection using bilevel optimisation with inexact gradients.

**Input :** Ground truth signal $u^\star \in \mathbb{R}^n$; data $y \in \mathbb{R}^m$; matrix $B \in \mathbb{R}^{m\times n}$; initial parameter $\theta^{(0)} \in \Theta$; a sequence of gradient errors $\{\mathcal{E}_k\}_{k=0}^{\infty}$; initial accuracies $\boldsymbol{\delta}^{(0)} = (\delta_1^{(0)}, \delta_2^{(0)})$; tolerance for the stopping criterion, $\texttt{tol} > 0$; accuracy line search parameters $0 < \beta_1 < 1 < \beta_2$.

**Output:** The learnt parameter $\theta \in \Theta$.

1 Set $u_0^{(0)} = \mathbf{0}$ ;
2 **for** $k = 0, 1, 2, \ldots$ **do**
3 Solve the lower level problem initialised at $u_0^{(k)}$ to $\delta_1^{(k)}$ accuracy; obtain $\tilde{u} = \tilde{u}_{\delta_1^{(k)}}(\theta^{(k)})$.
4 Compute $\tilde{\Phi}_{\delta_1^{(k)}}(\theta^{(k)})$, $\tilde{H}_{\delta_1^{(k)}}(\theta^{(k)})$ and $\tilde{b}_{\delta_1^{(k)}}(\theta^{(k)})$.
5 Compute $w_{\boldsymbol{\delta}^{(k)}}(\theta^{(k)})$ by solving the linear system to accuracy $\delta_2^{(k)}$.
6 Compute an estimate of the gradient error $\tilde{e}_{\boldsymbol{\delta}^{(k)}}(\theta^{(k)})$ using (7).
7 **if** $\tilde{e}_{\boldsymbol{\delta}^{(k)}}(\theta^{(k)}) \leq \mathcal{E}_k$ **then**
8 Compute the inexact gradient $\tilde{\nabla}^{(\boldsymbol{\delta}^{(k)})} L_{u^\star}(\tilde{u}) = \tilde{\Phi}_{\delta_1^{(k)}}(\theta^{(k)}) w_{\boldsymbol{\delta}^{(k)}}(\theta^{(k)})$.
9 Set $\theta^{(k+1)} = \theta^{(k)} - \tau^{(k)} \tilde{\nabla}^{(\boldsymbol{\delta}^{(k)})} L_{u^\star}(\tilde{u})$, where $\tau^{(k)}$ is chosen by backtracking scheme.
10 **if** $\|\tilde{\nabla}^{(\boldsymbol{\delta}^{(k)})} L_{u^\star}(\tilde{u})\|^2 < \texttt{tol}$ **then**
11 **return** $\theta^{(k+1)}$
12 **end**
13 Set $\boldsymbol{\delta}^{(k+1)} = \beta_2 \boldsymbol{\delta}^{(k)}$.
14 **else**
15 Set $u_0^{(k)} = \tilde{u}$.
16 Set $\boldsymbol{\delta}^{(k)} \leftarrow \beta_1 \boldsymbol{\delta}^{(k)}$.
17 Return to line 3.
18 **end**
19 **end**

Experiments based on the problem of learning the TV regularisation parameter will be used to illustrate and test the algorithm in the following section.

# 5 Experiments: learning the TV regularisation parameter

Unless stated otherwise, the experiments in this section are based on the following: the training set is of size $N = 1$; the ground truth signal $u^\star$ has size $n = 64$; the forward operator $B \in \mathbb{R}^{m\times n}$ is a subsampling matrix with $m = 32$; the observed data is generated by $y = Bu^\star + \xi$ where $\xi$ is Gaussian white noise with standard deviation $\sigma = 0.03$; the TV smoothing parameter is $\gamma = 10^{-3}$ and the convex penalty parameter is $\lambda = 10^{-3}$. The ground truth, observed data and some reconstructions are shown in Figure 1. A large value TV regularisation parameter, say

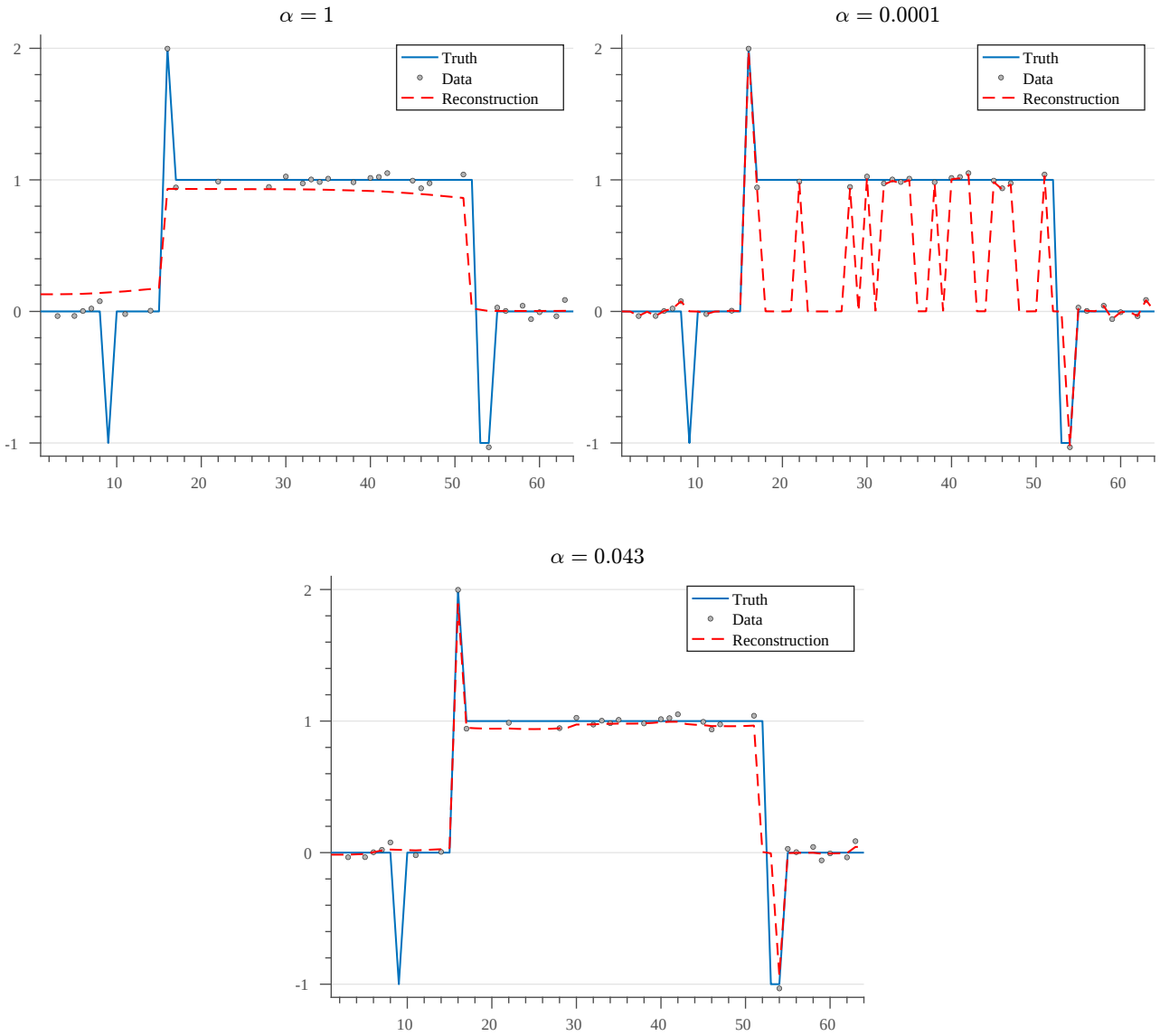


**Figure 1:** The reconstructions corresponding to three values of the TV regularisation parameter $\alpha$.

$\alpha = 1$, leads to poor results because sparse gradients are favoured over data fit. Conversely, when $\alpha = 0.001$ the data fit and convex penalty terms dominate, so the solution fits to the data where it exists and is zero elsewhere. Numerical experiments suggest that $\alpha = 0.043$ gives the best reconstruction. This reconstruction captures sharp discontinuities in the signal but does not fit to the noise in the data. Note that it is impossible to reconstruct the left-most discontinuity due to the absence of data at this point.

## 5.1 Estimating the error in the upper level gradient

For this toy example, the terms in the upper level gradient are given by

$$\begin{aligned} \Phi(\alpha) &= -D_u R_\gamma(\hat{u}(\alpha)) & \tilde{\Phi}_{\delta_1}(\alpha) &= -D_u R_\gamma(\tilde{u}_{\delta_1}(\alpha)) \\ H(\alpha) &= B^\top B + D_u^2 R_\gamma(\hat{u}(\alpha)) + \lambda I & \tilde{H}_{\delta_1}(\alpha) &= B^\top B + D_u^2 R_\gamma(\tilde{u}_{\delta_1}(\alpha)) + \lambda I \\ b(\alpha) &= \hat{u}(\alpha) - u^\star & \tilde{b}_{\delta_1}(\alpha) &= \tilde{u}_{\delta_1}(\alpha) - u^\star. \end{aligned} \tag{8}$$

Considered as functions of $u$, the function $\Phi$ and $H$ are Lipschitz continuous with constants $M_\Phi = 8/\gamma$ and $M_H = 16/\gamma^2$. Since $D_u R_\gamma(u)$ is $(8/\gamma)$-Lipschitz, it follows that

$$\|\Phi - \tilde{\Phi}\| = \|D_u R_\gamma(\hat{u}) - D_u R_\gamma(\tilde{u})\| \leq \frac{8}{\gamma}\|\hat{u} - \tilde{u}\|.$$

Noting that $\rho_\gamma''(x)$ is $(2/\gamma^2)$-Lipschitz, the derivation of $M_H$ proceeds similarly:

$$\begin{aligned} \|H - \tilde{H}\| &= \|(B^\top B + D_u^2 R_\gamma(\hat{u}) + \lambda I) - (B^\top B + D_u^2 R_\gamma(\tilde{u}) + \lambda I)\| \\ &= \|D_u^2 R_\gamma(\hat{u}) - D_u^2 R_\gamma(\tilde{u})\| \\ &= \left\| \mathcal{A}^\top \begin{pmatrix} \rho_\gamma''([\mathcal{A}\hat{u}]_1) & & \\ & \ddots & \\ & & \rho_\gamma''([\mathcal{A}\hat{u}]_n) \end{pmatrix} \mathcal{A} - \mathcal{A}^\top \begin{pmatrix} \rho_\gamma''([\mathcal{A}\tilde{u}]_1) & & \\ & \ddots & \\ & & \rho_\gamma''([\mathcal{A}\tilde{u}]_n) \end{pmatrix} \mathcal{A} \right\| \\ &\leq \|\mathcal{A}^\top\| \left\| \begin{pmatrix} \rho_\gamma''([\mathcal{A}\hat{u}]_1) - \rho_\gamma''([\mathcal{A}\tilde{u}]_1) & & \\ & \ddots & \\ & & \rho_\gamma''([\mathcal{A}\hat{u}]_n) - \rho_\gamma''([\mathcal{A}\tilde{u}]_n) \end{pmatrix} \right\| \|\mathcal{A}\| \\ &= \|\mathcal{A}^\top\| \max_{i=1,\dots,n} \left|\rho_\gamma''([\mathcal{A}\hat{u}]_i) - \rho_\gamma''([\mathcal{A}\tilde{u}]_i)\right| \|\mathcal{A}\| \\ &\leq \|\mathcal{A}^\top\| \max_{i=1,\dots,n} \frac{2}{\gamma^2} \left|[\mathcal{A}(\hat{u} - \tilde{u})]_i\right| \|\mathcal{A}\| \\ &\leq \|\mathcal{A}^\top\| \cdot \frac{2}{\gamma^2} \cdot \|\mathcal{A}(\hat{u} - \tilde{u})\| \cdot \|\mathcal{A}\| \\ &\leq \|\mathcal{A}^\top\| \cdot \frac{2}{\gamma^2} \cdot \|\mathcal{A}\|\|\hat{u} - \tilde{u}\| \cdot \|\mathcal{A}\| \\ &\leq \frac{16}{\gamma^2}\|\hat{u} - \tilde{u}\|. \end{aligned}$$

Finally, note that $\|H^{-1}\| \leq C_H = (\sigma_{\min}(B^\top B) + \lambda)^{-1}$, where $\sigma_{\min}(B^\top B)$ is the smallest singular value of the (known) matrix $B$. Substituting these constants into (7) yields (omitting

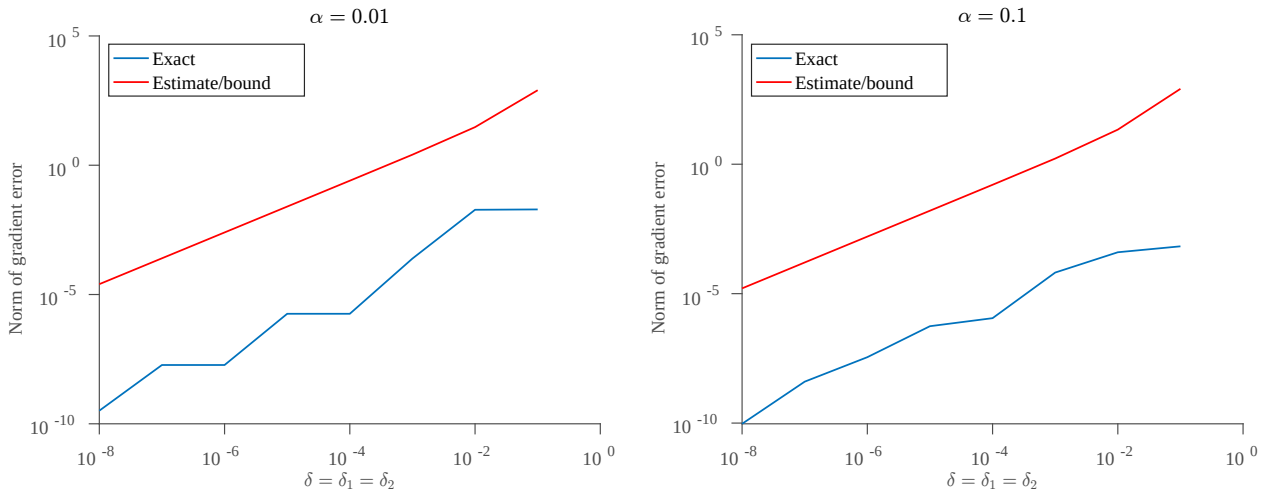


**Figure 2:** Examining the tightness of the bound at $\alpha = 0.01$ (left) and $\alpha = 0.1$ (right) for various values of $\boldsymbol{\delta} = (\delta_1, \delta_2)$ with $\delta_1 = \delta_2$. The 'exact' gradient was computed by computing the lower level solution to accuracy $\delta_1 = 10^{-10}$ and inverting the Hessian exactly. The estimate of the error is computed via (9).

dependencies on $\alpha$)

$$\|e_{\boldsymbol{\delta}}\| \leq (\sigma_{\min}(B^\top B) + \lambda)^{-1} \left( \frac{8\delta_1}{\gamma} + \|\tilde{\Phi}_{\delta_1}\| \right) \left( \delta_1 + \delta_2 + \frac{16\delta_1}{\gamma^2} \|\tilde{w}_{\boldsymbol{\delta}}\| \right) + \frac{8\delta_1}{\gamma} \|\tilde{w}_{\boldsymbol{\delta}}\|. \tag{9}$$

## 5.2 Tightness of the error bound

One of the key premises of this approach is that (7) provides a reasonable estimate for the error in the gradient. If the bound is not tight and the true error is much smaller than our estimate suggests, then excessively many computations will be performed. Therefore, it is useful to compare the estimated error against the true error. Of course, the true error is unknown, since the exact gradient is unknown. As a proxy for the exact gradient, we use the gradient that results from computing the lower level solution to accuracy $\delta_1 = 10^{-10}$ and inverting the Hessian exactly. We examine the tightness of the bound at various points in the parameter space and various values of $\boldsymbol{\delta} = (\delta_1, \delta_2)$. The results are shown in Figure 2. The tightness of the bound is poor: the estimated error and the true error differ by several orders of magnitude. This suggests that further theoretical work to improve the bound is necessary. However, the plot also shows that the estimated error differs from the true error by a constant multiplicative factor that is approximately independent of $\boldsymbol{\delta}$ (since the curves are vertically shifted on the log scale) and also of $\alpha$ (comparing the left- and right-hand plots). We can proceed to test our approach on the proviso that, if a maximum error $\mathcal{E}$ in the gradient is desired, then we should instead input $C\mathcal{E}$ as the required error for the purposes of Algorithm 1, where $C$ is the aforementioned multiplicative constant.

### 5.3 Choosing the error sequence $\{\mathcal{E}_k\}$

The error sequence $\{\mathcal{E}_k\}_{k=0}^{\infty}$ in Algorithm 1 determines how accurately the gradient is computed at each GD iteration for the upper level problem. To be precise, the $\mathcal{E}_k$ should more accurately be interpreted as being proportional to the error in the gradient at each iteration (see Section 5.2). Intuitively, efficiency gains can be achieved by using a decreasing error sequence. In the early stages of the algorithm, even highly erroneous gradients are likely to be sufficient to achieve descent. This is especially true in our one-dimensional problem, where the algorithm only needs to learn the correct sign of the gradient in order to move in the current direction in the parameter space. As the algorithm progresses towards the optimum, more precise gradient computations are needed to facilitate the final fine-tuning. There are two classes of error sequences we can consider: vanishing and non-vanishing. Vanishing error assumptions, e.g.

$$\lim_{k\to\infty} \mathcal{E}_k = 0, \qquad \sum_{k=0}^{\infty} \mathcal{E}_k < \infty,$$

are stipulated by most analyses of inexact gradient methods (Bertsekas 2017). However, Sra (2012) suggests that non-vanishing but uniformly bounded error sequences may lead to improved scalability. Clearly exact stationarity cannot be guaranteed when the errors are non-vanishing, but the iterates may progress towards inexact stationary points, i.e. points where the null gradient condition holds to within the current error level.

We test experimentally the performance of a variety of error sequences: constant (large), constant (small), vanishing but non-summable, and exponentially decreasing,

$$\mathcal{E}_k = C_{\text{large}}, \qquad \mathcal{E}_k = C_{\text{small}}, \qquad \mathcal{E}_k = Ck^{-1}, \qquad \mathcal{E}_k = C^{-\rho k},$$

for constants $C, \rho > 0$. The convergence history for each error sequence when applied to our example problem is shown in Figure 3. The learnt parameter and the number of lower level, linear solver, and upper level iterations required in each case is given in Table 1. When the error is fixed and large, the algorithm performs well initially, but has difficulty locating the precise location of the optimum later on. The errors prevent stationarity from being achieved, so the stopping criterion was not attained and the learnt parameter is not quite the objective minimiser. However, this scheme was very inexpensive computationally, requiring very few

**Table 1:** The computational cost and convergence status associated with each of the error sequences.

| Error sequence, $\mathcal{E}_k$ | Lower level iterations | Linear solver iterations | Upper level iterations | $\alpha$ |
|---|---|---|---|---|
| Constant large | 43,300 | 28,528 | $\geq 50$ | 0.040 |
| Constant small | 416,746 | 22,121 | 25 | 0.043 |
| Vanishing, non-summable | 67,766 | 23,396 | 34 | 0.043 |
| Exponentially decreasing | 295,492 | 23,241 | 23 | 0.043 |

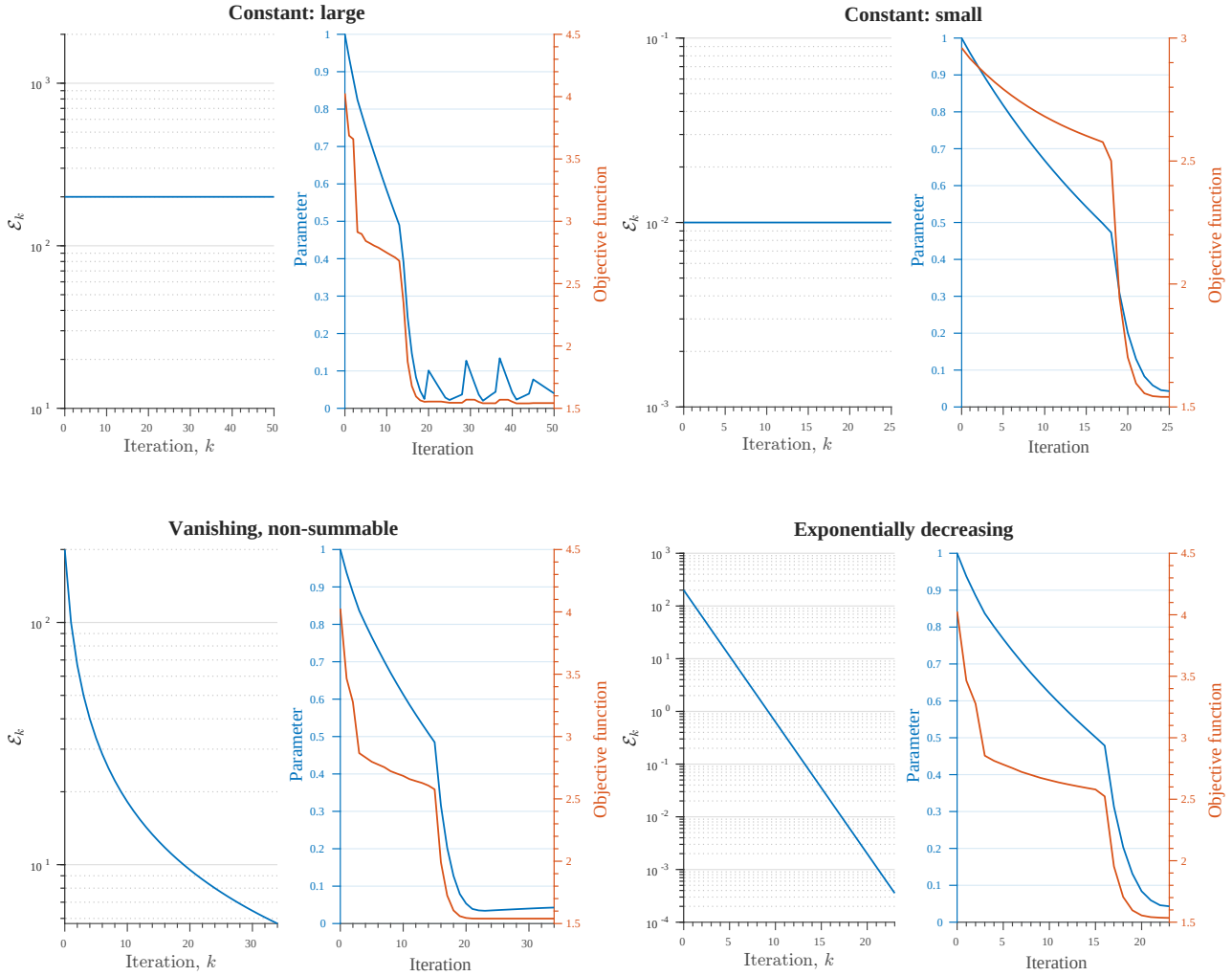


**Figure 3:** The convergence histories based on various error sequences: constant large, $\mathcal{E}_k = 200$; constant small, $\mathcal{E}_k = 10^{-2}$; vanishing, non-summable, $\mathcal{E}_k = 200/(k+1)$; exponentially decreasing, $\mathcal{E}_k = 200 \times 10^{-0.25k}$. Algorithm initialised at $\alpha^{(0)} = 1$ with stopping criterion $\texttt{tol} = 10^{-6}$, which was achieved in all cases except 'constant large'.

lower level iterations. Using a very small error threshold throughout results in no convergence issues, demonstrating a clear benefit in performing more precise computations as the algorithm progresses. However, both the large and small constant error sequences show similar initial performance (descending to an objective value of 0.4 after approximately 20 iterations), so using a small error from the outset involved significantly more computations for no discernible gain. We can combine the advantages of these two cases by specifying a decreasing error schedule, so that the initial gradient computations are cheap and the later estimates are precise. While the exponentially decreasing error sequence converged in fewer upper level iterations than the non-summable sequence, the number of lower level iterations was comparatively high. We conclude that the more gradually decreasing sequence performed best, requiring few lower level iterations and resolving the convergence issues encountered in the first sequence. Comparing this against the performance of the constant small error sequence, the advantage of our inexact gradients approach becomes evident: the number of lower level iterations was reduced by 84%.

# 6 Conclusion

## 6.1 Conclusion

We exploited inexact gradient methods to minimise the computational effort expended in solving the lower level problem and the linear system. An *a posteriori* bound for the error in the gradient was established, from which we proposed an algorithm which computes the gradient to within some specified error threshold at each iteration. A well-chosen sequence of error thresholds is shown to lead to significant computational savings while retaining good convergence properties. In our toy example, we find that a gradually decreasing error sequence performs best reducing the number of lower level iterations by 84% as compared with the naive approach of computing gradients to high accuracy at every iteration.

## 6.2 Future work

We establish an *a posteriori* bound on the error in the gradient, from which we can verify (post hoc) that a specified accuracy in the gradient has been obtained. Numerical experiments suggest the current bound is not very tight; our approach would benefit from further theoretical analysis to improve the tightness. A more significant outcome would be to establish an *a priori* bound, so that a sufficiently small $\boldsymbol{\delta}$ can be found in advance of performing any computations. This would afford two main advantages. First, the number of superfluous computations would be reduced. Using the *a posteriori* bound we can only know that the number of computations was more than required once they have already been performed. Second, it would permit a more systematic approach to choosing $\boldsymbol{\delta}$, instead of the line search procedure adopted presently.

The example considered in this report involved learning only a single parameter. It would be useful to test our approach for more complicated problems where the number of parameters is large. The key motivating reason for choosing a gradient-based approach in the first place was that they perform well on high-dimensional optimisation problems. Our example did not serve to illustrate this. A multi-parameter learning problem, e.g. learning the optimal MRI sampling pattern (Ehrhardt and Roberts 2020; Sherry et al. 2020), would also provide a more interesting test case. In the one parameter problem, errors in the gradient may not have a significant effect: provided the approximation of the gradient has the correct sign, the algorithm will move in the correct direction within the parameter space. However, if the parameter space is high-dimensional, then errors in the gradient are more likely to result in poor updates and convergence issues. As we consider tasks of varying dimensionality, it would be interesting to compare the performance of our methods with the derivative-free approach of Ehrhardt and Roberts (2020). We expect that gradient methods will perform better when the number of parameters is very large.

Our experimentation revealed some interesting conclusions about the best choice for the error sequence $\{\mathcal{E}_k\}$. It is not *a priori* clear how these conclusions would generalise to higher dimensional problems, where precise gradient computations may be necessary at an earlier stage in order to navigate the more complex objective landscape. Another future objective is to ground these conclusions in sound mathematical theory. Sra (2012) provides a useful starting point for deriving convergence results for inexact gradient methods, even under the weak assumption of non-vanishing computational errors.

Additional future work includes: expanding the framework to less tractable bilevel problems, e.g. relaxing the smoothness or convexity assumptions on the lower level objective; performing a thorough convergence analysis to derive convergence guarantees and convergence rates; testing the approach for large training sets and considering stochastic gradient descent with inexact gradient computations; formulating, experimentally or otherwise, some heuristics for choosing $\beta_1$ and $\beta_2$ used in the $\boldsymbol{\delta}$ update step.

# References

Sra, Suvrit (2012). "Scalable nonconvex inexact proximal splitting". en. In: *Advances in Neural Information Processing Systems* 25 (2012), p. 9.

Chambolle, Antonin and Thomas Pock (2016). "An introduction to continuous optimization for imaging". en. In: *Acta Numerica* 25 (May 2016), pp. 161–319. ISSN: 0962-4929, 1474-0508. DOI: 10.1017/S096249291600009X. URL: https://www.cambridge.org/core/product/identifier/S096249291600009X/type/journal_article (visited on 04/17/2021).

Qian, Hong, Yi-Qi Hu, and Yang Yu (2016). "Derivative-Free Optimization of High-Dimensional Non-Convex Functions by Sequential Random Embeddings". en. In: *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence* (2016), pp. 1946–1952.

Bertsekas, Dimitri P. (2017). "Incremental Gradient, Subgradient, and Proximal Methods for Convex Optimization: A Survey". en. In: *arXiv:1507.01030 [cs, math]* (Dec. 2017). arXiv: 1507.01030. URL: http://arxiv.org/abs/1507.01030 (visited on 04/26/2021).

Ehrhardt, Matthias J and Lindon Roberts (2020). "Inexact Derivative-free Optimization for Bilevel Learning". In: *arXiv preprint arXiv:2006.12674* (2020).

Sherry, Ferdia et al. (2020). "Learning the Sampling Pattern for MRI". In: *IEEE Transactions on Medical Imaging* 39.12 (2020), pp. 4310–4321.