AN ITERATION COUNT ESTIMATE FOR A MESH-DEPENDENT STEEPEST DESCENT METHOD BASED ON FINITE ELEMENTS AND RIESZ INNER PRODUCT REPRESENTATION

TOBIAS SCHWEDES*, SIMON W. FUNKE†, AND DAVID A. HAM‡

Abstract. Existing implementations of gradient-based optimisation methods typically assume that the problem is posed in Euclidean space. When solving optimality problems on function spaces, the functional derivative is then inaccurately represented with respect to $\ell^2$ instead of the inner product induced by the function space. This error manifests as a mesh dependence in the number of iterations required to solve the optimisation problem. In this paper, an analytic estimate is derived for this iteration count in the case of a simple and generic discretised optimisation problem. The system analysed is the steepest descent method applied to a finite element problem. The estimate is based on Kantorovich’s inequality and on an upper bound for the condition number of Galerkin mass matrices. Computer simulations validate the iteration number estimate. Similar numerical results are found for a more complex optimisation problem constrained by a partial differential equation. Representing the functional derivative with respect to the inner product induced by the continuous control space leads to mesh independent convergence.

Key words. Mesh dependent optimisation, continuous optimisation, gradient based optimisation, steepest descent method, finite element method, inner product representation, Riesz theorem

AMS subject classifications. 46E30, 46N10, 90C25, 65K10, 35Q93, 76M10

1. Introduction. Computationally solving an optimisation problem on Hilbert spaces requires discretisation and the application of an optimisation method to the resulting finite dimensional problem. In this context, mesh independent convergence of the optimisation algorithm means that, for a discretisation given by a sufficiently fine mesh, the number of iterations required to solve the optimisation problem to a given tolerance is bounded. Conversely, for a mesh dependent algorithm there exist arbitrarily fine meshes which will require an arbitrarily large number of iterations to achieve convergence.

As we will discuss in section 2, employing a “suitable” inner product in the optimisation algorithm is a necessary condition for achieving mesh independent convergence. However, many well-established continuous optimisation packages are not designed for function based optimisation and are hard-coded to apply the Euclidian ($\ell^2$) inner product (for example the MATLAB Optimization Toolbox [1], TAO [2], SNOPT [9], scipy.optimize [11, 4] and IPOPT [12, 20]). One can intuitively comprehend the inaccuracy in this practise for function based optimisation: since the inner product defines angles and distances, drawbacks in the convergence of optimisation methods using an inner product associated with a space different from the control space are expected. For example, this error manifests in suboptimal search directions. For gradient based optimisation methods, drawbacks can similarly be expected from the fact that the gradient representation of the derivative associated with the functional of interest is inner product dependent. As a result, one obtains mesh dependent convergence rates when these packages are applied to problems on Hilbert spaces. Only recently inner product aware optimisation packages have appeared [21, 18]. Thus, given the prevalence of this practise, it is germane to attempt to quantify the extent of mesh dependence in optimisation problems. In other words, to what extent is mesh independence a mathematical nicety, and to what extent an essential component of an acceptable optimisation algorithm? To the best of our knowledge, there are no analytical results on the iteration count of mesh dependent iterative methods in existing literature.

*Department of Mathematics, Imperial College London, London, UK and Grantham Institute, Imperial College London, London, UK (t.schwedes14@imperial.ac.uk).
†Center for Biomedical Computing, Simula Research Laboratory, Oslo, Norway (simon@simula.no).
‡Department of Mathematics, Imperial College London, London, UK (david.ham@imperial.ac.uk).
In this paper, we study the finite element discretisation of a generic optimisation problem formulated in a Hilbert function space. If we solve this problem using the steepest descent method with respect to the $\ell^2$ inner product, the convergence rate is mesh dependent. For this situation we analytically derive an iteration number estimate that reveals that the number of iterations is polynomial in the ratio between largest and smallest directional element size of the mesh. Conversely, if the optimisation is conducted in the inner product induced by the Hilbert space, the optimisation converges in exactly one iteration, independently of the computational mesh employed. Numerical experiments confirm the analytical estimate in the special case of the Hilbert space $L^2$.

The estimate is derived for a particularly simple case in the expectation that the impact of an “incorrect” inner product on more complex problems will be at least as severe. We illustrate this using the example of the optimal control of the Poisson equation, a partial differential equation (PDE) constrained optimisation problem. When the $\ell^2$ inner product is employed in the optimisation, the iteration count is mesh dependent, similar to that observed in the simple case. Conversely, by using an implementation that employs the $L^2$ inner product, mesh independent convergence is achieved.

This work is organised as follows: In section 2, we examine the existing work on mesh dependence. Section 3 restates the Riesz representation theorem from functional analysis. In section 4, the continuous optimisation problem considered throughout this paper is formulated, and section 5 is dedicated to the finite element discretisation of the problem. In section 6, the iteration count for the steepest descent making use of the $L^2$ represented Riesz gradient is derived, while section 7 develops an estimate for the iteration number when a gradient is used that is represented with respect to the Euclidean scalar product. Numerical experiments investigating the validity in applications for the stated iteration counts are presented in section 8, as well as a simulation study considering mesh dependence for a PDE constrained problem.

2. Related work. Recent work on mesh independence in finite element methods focuses on achieving this characteristic in the solution of the primary finite element problem \[13, 10, 16\]. This is, in fact, very closely related to the problem considered here: solving the finite element problem amounts to minimising the residual, and the task of mapping the residual gradient from the dual space back into the primal function space is similar to employing the correct inner products in calculating updates for the optimisation problem. \[13\] derives bounds for the condition number of finite element operator matrices based on the structure that the discretisation inherits from the underlying continuous Hilbert space. The condition number depends on the underlying discretisation due to a non-isometric embedding of the operator into Euclidean space. Spectral bounds that are independent of the finite element subspace (that is in particular of the mesh) are achieved by applying the Riesz map as preconditioner. This abstract preconditioning principle transfers directly to the problem considered here: mesh independent convergence requires that the Fréchet derivative of the objective functional, defining the search direction in the optimisation, is represented with respect to the correct Hilbert space. This is equivalent to employing an $\ell^2$ representation and preconditioning with the Riesz map. In the case of the Hilbert space $L^2$, the Riesz map of the discretised problem is the inverse of the Galerkin mass matrix.

More generally, \[10\] shows that the choice of a preconditioner is equivalent to the choice of an inner product on the underlying space. That work investigates the dependence of conjugate gradient and MINRES methods in Hilbert spaces on the underlying scalar product. In finite dimensions, the naturally induced scalar product is related to every other inner product by the application of a positive definite matrix. The Riesz map preconditioner corresponds to an application of that matrix’s inverse. For $L^2$, this is once again inverse of the Galerkin mass matrix.
\[ f'(u)v = \langle \hat{u}, v \rangle_H \]

for all \( v \in H \). The map \( R_H : H^* \to H \) that sends \( f'(u) \) to its unique representative \( \hat{u} \in H \) is called the Riesz map. Further, \( R_H(f'(u)) = \hat{u} \) is called the Riesz representor of \( f'(u) \) in \( H \), or the gradient of \( f \) in \( u \) represented with respect to \( H \).

4. Formulation. In order to discuss how the choice of the inner product for the representation of the functional derivative impacts upon the convergence of the optimisation with respect to the underlying discretisation, a problem as simple as possible, at the same time being generic in this context, is considered. Such an optimisation problem is then exemplary while its simplicity suggests dependence on the inner product and the discretisation of at least similar order for more complex optimisation problems, including problems with PDE constraints. Consider the minimisation problem,

\[ \min_{u \in H} \alpha(u, u)_H + \beta(u, v)_H + \gamma, \]

where \( \alpha > 0, \beta, \gamma \in \mathbb{R} \) and \( v \in H \). Further, \( H \subset \{ u : \Omega \to \mathbb{R} \} \), with bounded domain \( \Omega \subset \mathbb{R}^n \) and \( n \in \mathbb{N} \), is a Hilbert function space with inner product \( \langle \cdot, \cdot \rangle_H \) that is of some integral type. For instance, \( H \) could be a Hilbert Sobolev space, i.e. \( H = W^{k,2} \) with \( k \in \mathbb{N} \). The derivation of the steepest descent iteration count estimate for (2) can be reduced to the consideration of the simple optimisation problem

\[ \min_{u \in L^2(\Omega)} \left\{ f(u) = \frac{1}{2} \langle 1 - u, 1 - u \rangle_{L^2} \right\}, \]

where \( \langle \cdot, \cdot \rangle_{L^2} \) denotes the inner product of the space \( L^2 := L^2(\Omega) \). This simplification is justified by the common finite element approach and the quadratic structure of both discretisations. Details are found in section 5. It is easy to see that (3) is well posed with \( u = 1 \) as unique solution.

In the subsequent sections we consider the solution of (3) by using the steepest descent algorithm with exact line search and the finite element method. In doing so, two approaches are compared: In the first one, the continuous formulation of \( f \) in (3) is employed in order to compute the gradient used for the optimisation method. Then, the finite element discretisation is applied. In the second one, the discretisation is performed first, whereupon the gradient is computed with respect to the \( L^2 \) inner product of the coefficient vectors. We will see that the formulation used in the second approach is mesh dependent by a scaling of the local finite element mass matrix. Results of the optimisation procedure using both approaches are compared for non-uniform meshes.

5. Finite element discretisation. Let us now consider the nodal finite element discretisation of the continuous optimisation problem (3). Let \( T \) be a tessellation of the domain \( \Omega \) by topologically identical \( n \)-polytopes \( K \). Typically the cells, \( K \) are simplices or hypercubes. Assume that any \( K \in T \) is equipped with an individual set of linearly independent functions \( \mathcal{P} = \{ \phi_1, \ldots, \phi_p \} \subset L^2(K) \) and another set of functions \( \mathcal{N} = \{ N_1, \ldots, N_p \} \), where
$N_1, ..., N_p : L^2(K) \to \mathbb{R}$ are called nodal variables and form a basis of $\mathcal{P}'$, the dual space of $\mathcal{P}$. Then, each $(K, \mathcal{P}, \mathcal{N})$ forms a finite element following Ciarlet’s definition in [5]. If $v$ is a function in $L^2(K)$, the local interpolant of $v$ on $K$ is defined by

$$I_K(v) = \sum_{i=1}^{p} N_i(v) \phi_i,$$

following the formulation in [3]. Thus, $v$ is approximated by the sum of basis functions $\phi_1, ..., \phi_p$ and coefficients $N_1(v), ..., N_p(v)$. Typically, the application of $N_1, ..., N_p$ to $v$ involves the evaluation of $v$ or one of its derivatives on some given point on $K$. Let $u \in \mathcal{V} \subseteq C^m(\Omega) \subset L^2$, where $m$ is the order of the highest partial derivatives involved in evaluating $N_i(u|_K)$ for $i = 1, ..., p$. Then, the global interpolant of $u$ on $\mathcal{T}$ is defined by

$$I_T(u)|_K = I_K(u|_K)$$

for any $K \in \mathcal{T}$. Note that $I_T$ may have multiple values at the interfaces of elements. However, this is not problematic since these sets have Lebesgue measure equal to zero. It is obvious that $I_T(u)$ can be written similarly to (4) when $\phi_1, ..., \phi_p$ are expanded to $\Omega$ as follows: in case $\phi_i|_K = 0$, set $\phi_i$ equal to zero outside of $K$. If $\phi(x) \neq 0$ for some $x \in \partial K$, set $\phi_i$ equal to the basis functions on neighbouring elements that are also not equal to zero in $x$. On all other elements, set $\phi_i$ equal to zero. The extensions of the basis functions of all elements are given by the so-called global basis functions $\varphi_1, ..., \varphi_d$, and their corresponding global nodal variables by $N_1^g, ..., N_d^g$, where

$$u_i := N_i^g(u) = N_i(u|_K),$$

for $\varphi_i|_K = \phi_i \neq 0$ on $K$, where $\varphi_i$ denotes the basis function associated with the global basis function $\varphi_i$ on $K$. Hence, the global interpolant $I_T(u) \in \mathcal{V}_h \subset \mathcal{V}$ with $\dim(\mathcal{V}_h) = d$ can be expressed as

$$I_T(u)(x) = \sum_{i=1}^{d} u_i \varphi_i(x).$$

One computes

$$f(I_T(u)) = \int_{\Omega} \frac{1}{2} \left(1_i - I_T(u)(x)\right)^2 \, dx$$

$$= \frac{1}{2} \int_{\Omega} \left(\sum_{i=1}^{d} (1_i - u_i) \varphi_i(x)\right)^2 \, dx$$

$$= \frac{1}{2} \sum_{i,j=1}^{d} (1_i - u_i)(1_i - u_j) \int_{\Omega} \varphi_i(x) \varphi_j(x) \, dx,$$

where $1_i = N_i^g(1)$ for any $i = 1, ..., d$. Given the mass matrix $M$ with $M_{ij} := \int_{\Omega} \varphi_i \varphi_j \, dx$, the finite element discretised version of the original optimality problem (3) can be expressed as

$$\min_{I_T(u)} f(I_T(u)) = \min_{\tilde{u} \in \mathbb{R}^d} \left\{ f(\tilde{u}) = \frac{1}{2} (\tilde{1} - \tilde{u})^T M (\tilde{1} - \tilde{u}) \right\},$$

where $\tilde{u} = (u_1, \ldots, u_d)$, $\tilde{1} = (1_1, \ldots, 1_d) \in \mathbb{R}^d$. Here, we identified $f|_{\mathcal{V}_h}$ with the function

$$\tilde{u} \mapsto \frac{1}{2} (\tilde{1} - \tilde{u})^T M (\tilde{1} - \tilde{u}).$$

This makes sense since for any $u \in L^2$ there is a unique $\tilde{u} \in \mathbb{R}^d$ such that (7) holds true.
6. Iteration count using $L^2$ inner product. In order to solve problem (3), we apply the steepest descent method while representing the gradient according to Riesz theorem with respect to the $L^2$ inner product. Initialized by $u_0$, the first iterate $u_1$ is given by $u_1 = u_0 - \alpha R_{L^2}(f'(u_0))$, where $R_{L^2}(f'(u_0))$ denotes the Riesz reprenter of the first Fréchet derivative $f'(u_0)$ in $L^2$. We compute $f'(u) \in L(L^2, \mathbb{R})$ as

$$f'(u)(\cdot) = -\frac{1}{2} \left( \langle \cdot, 1 - u \rangle_{L^2} + (1 - u, \cdot)_{L^2} \right)$$

(13)

$$= -(1 - u, \cdot)_{L^2},$$

(14)

where we used the symmetry of $(\cdot, \cdot)_{L^2}$. Hence,

$$R_{L^2}(f'(u)) = -(1 - u).$$

(15)

The second Fréchet derivative $f''(u) \in L(L^2, L(L^2, \mathbb{R}))$ is given by

$$f''(u) \cdot \cdot = (\cdot, \cdot)_{L^2}.$$  

(16)

For any $n > 2$, we find $F(u(n)) = 0$. Applying Taylor’s theorem for function spaces at $u_1 = u_0 - \alpha R(f'(u_0))$ using (15) yields

$$f(u_1) = f(u_0) - \alpha f'(u_0) R_{L^2} (f'(u_0)) + \frac{1}{2} \alpha^2 f''(u_0) (R_{L^2} (f'(u_0)))^2$$

(17)

$$= \left( 1 - \frac{1}{2} \alpha + \frac{1}{2} \alpha^2 \right) \|1 - u_0\|^2_{L^2}.$$

(18)

One easily computes that the minimum of $f(u_1)$ is found for $\alpha^* = 1$, i.e. $\alpha^*$ is the step size of an exact line search, and $u_1 \equiv 1$. Hence, the solution of the optimality problem (3) is found after one steepest descent iteration, independently of the initial $u_0$. This still holds for the finite element discretised version (11) of problem (3). This can be seen from the fact that

$$R_{L^2}(f'(u_0)) = u_0 - 1 = \sum_{i=1}^d (u_0 - 1)_i \cdot \varphi_i = I_T (R_{L^2} (f'(u_0))),$$

(19)

for any $u_0 = \sum_{i=1}^d (u_0)_i \varphi_i \in I_T(L^2) := \{ I_T(u) : u \in L^2 \}$, where $(u_0 - 1)_i := (u_0)_i - 1_i$ for any $i = 1, \ldots, d$. Hence, the first steepest descent iterate for the discretised optimisation problem (11) equals the constant function 1, which is the optimum.

7. Iteration count estimate using $\ell^2$ inner product. In this section, an iteration count estimate is derived for the steepest descent method using gradients with respect to the $\ell^2$ inner product, computed based on the finite element discretised version (11) of the optimality problem (3). Since $M$ is symmetric, one computes

$$f'(\bar{u}) = - (\bar{u} - \bar{u})^T M = - M (\bar{u} - \bar{u}),$$

(20)

where $(\cdot, \cdot)_{\ell^2}$ denotes the $\ell^2$ inner product, i.e. $(\bar{u}, \bar{v})_{\ell^2} := \bar{u}^T \bar{v} = \sum_{i=1}^d u_i v_i$ for any $\bar{u}, \bar{v} \in \mathbb{R}^d$. Thus,

$$R_{\ell^2}(f'(\bar{u})) = \nabla f(\bar{u}) = - M (\bar{u} - \bar{u}).$$

(21)

Notice that $R_{\ell^2}(f')$ differs from $R_{L^2}(f')$ in (15) by the multiplication of the mass matrix $M$. Since $M$ reflects the structure of the underlying mesh, that is, the spatial distribution of elements and their sizes, it is expected that the convergence of the steepest descent method using $R_{\ell^2}(f')$ is mesh dependent.
Using (21), the \((k + 1)\)th iterate of the steepest descent method is given by \(\vec{u}_{k+1} = \vec{u}_k + \alpha_k M (\vec{I} - \vec{u}_k)\). In order to minimise
\[
(22) \quad f(\vec{u}_{k+1}) = \frac{1}{2} (\vec{I} - (\vec{u}_k + \alpha_k M (\vec{I} - \vec{u}_k)))^T M (\vec{I} - (\vec{u}_k + \alpha_k M (\vec{I} - \vec{u}_k))),
\]
with respect to \(\alpha_k\), we set \(\frac{\partial f}{\partial \alpha_k}(\vec{u}_{k+1}) = 0\) such that
\[
(23) \quad \alpha_k = \frac{(\vec{I} - \vec{u}_k)^T M^2 (\vec{I} - \vec{u}_k)}{(\vec{I} - \vec{u}_k)^T M^3 (\vec{I} - \vec{u}_k)}
\]
Note that the Hessian of \(f\) equals \(M\). Considering the second derivative of \(f(\vec{u}_{k+1})\) with respect to \(\alpha_k\) and using the fact that mass matrices are positive definite shows that \(\alpha_k\) as given in (23) is the unique minimiser, i.e. \(\alpha_k\) is the step size of an exact line search.

Using steepest descent with exact line search on a strongly convex quadratic function, we may apply the Lemma of Kantorovich from section 8.6 in [15]. The lemma provides a recursive error estimate in \(f\) we may apply the Lemma of Kantorovich from section 8.6 in [15]. The lemma provides an
\[
T(29)
\]

terminations, i.e.
\[
(28)
\]

An obvious question is how to determine \(\kappa(M)\) from the underlying discretisation of the domain. Corollary 1 of [7] gives an upper bound for the condition number of mass matrices depending on the minimum and maximum eigenvalues of their local mass matrices. Let \(M_K\) be the local mass matrix associated with element \(K\), and let \(\lambda_{max}^{M_K}\) and \(\lambda_{min}^{M_K}\) denote the maximum and minimum eigenvalues of \(M_K\), respectively. Applying the estimate to \(M\), we obtain
\[
(27) \quad f(\vec{u}) = (1, \ldots, 1) \in \mathbb{R}^d \) is the optimal value such that \(f(\vec{u}^*) = 0\).

An obvious question is how to determine \(\kappa(M)\) from the underlying discretisation of the domain. Corollary 1 of [7] gives an upper bound for the condition number of mass matrices depending on the minimum and maximum eigenvalues of their local mass matrices. Let \(M_K\) be the local mass matrix associated with element \(K\), and let \(\lambda_{max}^{M_K}\) and \(\lambda_{min}^{M_K}\) denote the maximum and minimum eigenvalues of \(M_K\), respectively. Applying the estimate to \(M\), we obtain
\[
(28) \quad \kappa(M) \leq \frac{\max_K \lambda_{max}^{M_K}}{\min_K \lambda_{min}^{M_K}}
\]
where \(p_{max}\) is the maximum number of elements around any nodal point. In order to determine \(\lambda_{max}^{M_K}\) and \(\lambda_{min}^{M_K}\), respectively, we consider the bijective transformation \(T_K\) from a reference element \((\hat{K}, \mathcal{P} = \{\psi_1, \ldots, \psi_p\}, \mathcal{N} = \{\hat{N}_1, \ldots, \hat{N}_p\})\) to the local element \((K, \mathcal{P}^K = \{\phi_1^K, \ldots, \phi_p^K\}, \mathcal{N}^K = \{N_1^K, \ldots, N_p^K\})\), where \(\phi_i^K \circ T_K = \psi_i\). Let us assume affine transformations, i.e.
\[
(29) \quad T_K : \hat{K} \to K \quad x \mapsto J_K x + y_K,
\]
for $J_K \in \mathbb{R}^{n \times n}$ invertible and $y_K \in \mathbb{R}^n$. Applying singular value decomposition to the Jacobian matrix $J_K$ of the transformation $T_K$, there are unitary $U, V \in \mathbb{R}^{n \times n}$ such that

$$J_K = U H_K V,$$

where $H_K \in \mathbb{R}^{n \times n}$ is a diagonal matrix with positive real numbers $h^K_1, \ldots, h^K_n$ on the diagonal. Thus,

$$|\det(J_K)| = |\det(H_K)| = h^K_1 \cdot \ldots \cdot h^K_n. \quad (30)$$

The factors $h^K_1, \ldots, h^K_n$ describe the scaling of lengths in orthogonal directions between reference element $\hat{K}$ and local element $K$, respectively. Using the transformation theorem, an entry $\mu^K_{lm}$ with $l,m = 1, \ldots, p$ of the local mass matrix $M_K$ associated with element $K$ can therefore be written as

$$\mu^K_{lm} = \int_{\hat{K}} \phi^K_l(x)\phi^K_m(x)dx$$

$$= \int_{\hat{K}} \phi^K_l(T_K(\zeta))\phi^K_m(T_K(\zeta))|\det(J_K)|d\zeta$$

$$= h^K_1 \cdot \ldots \cdot h^K_n \int_{\hat{K}} \psi_l(\zeta)\psi_m(\zeta)d\zeta$$

$$= h^K_1 \cdot \ldots \cdot h^K_n \cdot \hat{\mu}_{lm}. \quad (32)$$

Consequently, the local mass matrix $M_K$ has the form

$$M_K = h^K_1 \cdot \ldots \cdot h^K_n \hat{M}, \quad (36)$$

where $\hat{M} = (\hat{\mu}_{lm})_{l,m=1,\ldots,p}$ is the mass matrix associated with the reference element $\hat{K}$. Hence, the eigenvalues of $M_K$ are proportional to the eigenvalues of $\hat{M}$, with proportionality factor equal to $h_1 \cdot \ldots \cdot h_n$. Note that the matrix $\hat{M}$ is independent of the underlying element $K$.

Let $\lambda^M_{\text{max}}$ and $\lambda^M_{\text{min}}$ denote the largest and smallest eigenvalues of $\hat{M}$, respectively. The estimate (28) can now be written as

$$\kappa(M) \leq p_{\text{max}} \frac{\lambda^M_{\text{max}}}{\lambda^M_{\text{min}}} \max \left\{ \prod_{i=1}^n h^K_i : K \right\}$$

$$\leq p_{\text{max}} \frac{\lambda^M_{\text{max}}}{\lambda^M_{\text{min}}} \prod_{i=1}^n \max_K h^K_i \min_K h^K_i. \quad (37)$$

Due to the role of $h^K_1, \ldots, h^K_n$ as scaling factors of the element $K$ relative to the reference element $\hat{K}$ in orthogonal directions, the expression $\max \left\{ \prod_{i=1}^n h^K_i : K \right\} / \min \left\{ \prod_{i=1}^n h^K_i : K \right\}$ may be considered as a measure of non-uniformity in the mesh.

In the particular case where the maximum and minimum of the scaling factors $h^K_i$ in orthogonal directions coincide for $i = 1, \ldots, n$, we may set $h_{\text{max}} := \max_K h^K_i$ and $h_{\text{min}} := \min_K h^K_i$ for any $i$, respectively, such that (38) simplifies to

$$\kappa(M) \leq p_{\text{max}} \frac{\lambda^M_{\text{max}}}{\lambda^M_{\text{min}}} \left( \frac{h_{\text{max}}}{h_{\text{min}}} \right)^n. \quad (39)$$
Assuming that the underlying mesh contains significant non-uniformity, i.e. \( h_{\text{max}}/h_{\text{min}} \gg 1 \), the following approximation holds true:

\[
\log \left( \frac{p_{\max} \, \lambda_{\text{max}}^{\alpha} \left( \frac{h_{\text{max}}}{h_{\text{min}}} \right)^n - 1}{p_{\max} \, \lambda_{\text{max}}^{\alpha} \left( \frac{h_{\text{max}}}{h_{\text{min}}} \right)^n + 1} \right) = \sum_{\nu=1}^{\infty} \frac{(-1)^\nu}{\nu} \left( \int_{\lambda_{\text{max}}}^{\lambda_{\text{min}}} \left( \frac{h_{\text{max}}}{h_{\text{min}}} \right)^n - 1 \right)^\nu \prod_{\nu=1}^{n} \nu \frac{\lambda_{\text{max}}^{\alpha}}{\lambda_{\text{min}}^{\alpha}} \left( \frac{h_{\text{max}}}{h_{\text{min}}} \right)^n + 1 \right)^{-\nu} \approx -2 \cdot \left( \frac{\lambda_{\text{max}}^{\alpha}}{\lambda_{\text{min}}^{\alpha}} \left( \frac{h_{\text{max}}}{h_{\text{min}}} \right)^n + 1 \right)^{-1}.
\]

Setting (27) smaller or equal to \( \varepsilon \), one then obtains

\[
k \geq \frac{1}{4} \left( \log \left( \frac{\varepsilon}{f(u_0)} \right) \cdot \left( \frac{\lambda_{\text{max}}^{\alpha}}{\lambda_{\text{min}}^{\alpha}} \left( \frac{h_{\text{max}}}{h_{\text{min}}} \right)^n + 1 \right) \right) =: \hat{k}.
\]

Thus, after \( \hat{k} \) iterations the error of the steepest descent method is smaller or equal to \( \varepsilon \). Careful interpretation of this result yields that an iteration number satisfying that the associated error is smaller or equal to \( \varepsilon \) is at most polynomial in the ratio between largest and smallest directional element size of order equal to the domain dimension \( n \). The subsequent simulations show that this dependency is also observed numerically.

**Remark:** In what follows, we briefly justify the simplification of considering (2) instead of (3) in the context of deriving (43). With the notation used above, discretising using finite elements leads to

\[
\alpha(u, u)_H + \beta(u, v)_H + \gamma,
\]

where \( v_1, ..., v_n \) are the coefficients of the interpolant \( I_T(v) = \sum_{i=1}^{n} v_i \phi_i \) of \( v \in H \), and \( M_H = \left( \langle \phi_i, \phi_j \rangle_H \right)_{i,j} \). In the special case of \( H = L^2([0, 1]) \), it holds that \( M_H \) coincides with the Galerkin mass matrix \( M \). Since adding constants does not change the solution of an optimisation problem, minimising the right hand side of (45) is equivalent to minimising

\[
\frac{1}{2} \tilde{u}^T \tilde{M}_H \tilde{u} + \tilde{u}^T \tilde{v},
\]

where \( \tilde{M}_H = 2\alpha M_H \) and \( \tilde{v} = \beta M_H \tilde{v} \). Analogously, minimising (46) is equivalent to minimising

\[
\frac{1}{2} \tilde{u}^T \tilde{M}_H \tilde{u} + \tilde{u}^T \tilde{v} + \frac{1}{2} \tilde{u}^T \tilde{M}_H \tilde{u} = \frac{1}{2} \left( \tilde{u} - \tilde{u}^* \right)^T \tilde{M}_H \left( \tilde{u} - \tilde{u}^* \right),
\]

where \( \tilde{u}^* \) is the optimal solution, for which obviously \( \tilde{M}_H \tilde{u}^* = -\tilde{v} \). Since Kantorovich’s inequality is valid for any \( \tilde{u}^* \) and the estimate in (25) depends only on the condition number of the Hessian matrix of the quadratic problem, one may conclude that the iteration count estimate is valid for minimising (47) in the same way as for the minimisation of

\[
\frac{1}{2} \left( \tilde{u} - \tilde{I} \right)^T \tilde{M}_H \left( \tilde{u} - \tilde{I} \right).
\]
We need to show that (43) applies for $\tilde{M}_H$ similarly as for $M$. For the sake of clarity, substitute $\tilde{M}_H$ with $M$. Let $\langle \cdot , \cdot \rangle_{H(K)}$ and $\langle \cdot , \cdot \rangle_{\hat{H}(\hat{K})}$ be defined as the inner products of the same integral type as $H$, with integration domains equal to $K$ and $\hat{K}$, respectively. The local finite element matrix $M_{H,K}$ corresponding to $M_H$ is then given by the entries

$$M_{H,K}^{H,K} = 2\alpha \cdot \langle \phi^K_l, \phi^K_m \rangle_{H(K)},$$

where $\phi^K_l$ with $l = 1, \ldots, p$ denote the local basis functions associated with element $K$. In a manner analogous to the proof of Theorem 1 in [7], one can show that the condition number estimate (28) holds true for $M_H$, i.e.

$$\kappa(M_H) \leq p_{\max} \frac{\max_K \lambda_{M_{H,K}}^{M_{H,K}}}{\min_K \lambda_{M_{H,K}}^{M_{H,K}}}.$$

The reference finite element matrix $M_H$ corresponding to $M_H$ is given by the entries

$$M_{H,K}^{H,K} = 2\alpha \cdot \langle \psi_l, \psi_m \rangle_{\hat{H}(\hat{K})},$$

where $\psi$ with $l = 1, \ldots, p$ denote the basis functions associated with the reference element $\hat{K}$. Since the inner products are of integral type, the transformation theorem applies, analogously to (32)-(35), for the mapping from the reference element to the local element, i.e.

$$\mu_{H,K}^{H,K} = 2\alpha \cdot \langle \phi^K_l, \phi^K_m \rangle_{H(K)}$$

$$= 2\alpha \cdot h^K_1 \cdots h^K_n \langle \psi_l, \psi_m \rangle_{\hat{H}(\hat{K})}$$

$$= h^K_1 \cdots h^K_n \mu_{H}^{H}.$$

Since the remaining derivation does not change, it is easy to see that formula (43) is valid for the finite element discretised version of (2), when $f(\vec{u}_0)$ is replaced by $f(\vec{u}_0) - f(\vec{u}^*)$, and $\lambda_{\hat{M}}^{\max}, \lambda_{\hat{M}}^{\min}$ are replaced by $\lambda_{M_{H,K}}^{\max}, \lambda_{M_{H,K}}^{\min}$, respectively.

8. Numerical experiments. In this section, the results above are verified by solving the previously discussed generic optimisation problem (2) numerically. The behaviour predicted is also observed to carry over to the numerical solution of a PDE constrained optimisation problem in two dimensions. The simulations were performed using the FEniCS automated finite element system [14]. For the PDE constrained problem, the optimisations were conducted according to the framework presented in [8]. It makes use of the reduced problem formulation and of the automated optimisation facilities of dolfin-adjoint [6]. The actual code employed to conduct the simulations is in the supplement to this paper.

8.1. Generic optimisation problem. The iteration number estimate from the last section applies if the $\ell^2$ inner product is used in the representation of the derivative of the objective functional $f$. Corresponding simulations for the solution of (3) with initial $u_0 = 0$ were run on successively refined meshes. The $(r + 1)$-times refined mesh is derived from the $r$-times refined mesh by doubling the mesh resolution in part of the $r$-times refined mesh region. That way, $h_{\max}/h_{\min}$ doubles after every refinement. Figure (1) displays the evolution of iteration numbers with respect to $h_{\max}/h_{\min}$ using linear Lagrange elements in one, two and three dimensions and the corresponding analytical iteration count estimate. The polynomial dependency between iteration number and $h_{\max}/h_{\min}$ in the estimate (43) is numerically confirmed. Analogous dependencies were found for other and higher order finite elements implemented in FEniCS, emphasising that the iteration number evolution indeed exhibits the polynomial relationship stated in (43), independently of the finite element choice.
The iteration numbers from Figure (1) compare to an iteration number that is always, and independently of the mesh or the dimension of the underlying domain, equal to 1 when the functional derivative of $f$ is represented with respect to $L^2$ corresponding to the control variable space.

Fig. 1: Iterations of the steepest descent method with exact line search and $\ell^2$ represented gradients, using linear Lagrange finite elements on non-uniform refined meshes with respect to $h_{\text{max}}/h_{\text{min}}$ on a log-log scale, and with error threshold $\varepsilon = 10^{-15}$ in the functional.

By *uniform refinement* we mean that the mesh is refined by an equal factor throughout the domain such that the ratio $h_{\text{max}}/h_{\text{min}}$ remains unaltered. Consequently, we do not expect the iteration number to increase under uniform refinement according to (43). Indeed, the results given in table 1 confirm this. In fact, the iterations tend to decrease slightly for increasing refinement levels. This makes sense as the relative impact of the numerical error due to the boundaries gets smaller as the total number of elements increases. For the simulations, an initially non-uniform mesh was considered. Similar results are obtained for an initially uniform mesh, with overall smaller iteration numbers compared to table 1. In summary, the increase of iteration counts depends on how the mesh is refined, and not on what initial mesh is used. Further, the iteration numbers increase for increasing orders of
the Lagrange elements, since the larger the Lagrange finite elements order is, the larger is
the ratio $h_{\text{max}}/h_{\text{min}}$.

Table 1: Iterations of the steepest descent method with exact line search and $\ell^2$ represented
gradients for Lagrange finite elements of different order (CG1-5) on uniformly refined meshes
in 1D depending on the refinement level (number of elements). The error threshold is given
by $\varepsilon = 10^{-6}$ in the functional.

| Refinement | CG1 | CG2 | CG3 | CG4 | CG5 |
|------------|-----|-----|-----|-----|-----|
| 1 (96)     | 13  | 20  | 19  | 37  | 43  |
| 2 (192)    | 12  | 20  | 16  | 36  | 41  |
| 3 (384)    | 11  | 19  | 15  | 35  | 40  |
| 4 (768)    | 10  | 19  | 13  | 35  | 38  |
| 5 (1536)   | 8   | 19  | 13  | 35  | 37  |

8.2. Optimal control of the Poisson equation. A common case for function based
optimisation to occur is PDE-constraint optimisation. To exemplify this we consider a
simple and generic problem in PDE-constraint optimisation, namely an optimal control
problem constrained by the Poisson equation. Physically, this problem can be interpreted
as finding the best heating/cooling of a hot plate to achieve a desired temperature profile.
For this problem, we analyse the performance of multiple commonly used implementations
of optimisation algorithms, based on different inner products. Mathematically, the problem
is to minimise the following tracking type functional

$$\min_{f \in H} \frac{1}{2} \| u - d \|^2_{L^2(\Omega)} + \frac{\alpha}{2} \| f \|^2_H,$$

subject to the Poisson equation with Dirichlet boundary conditions,

$$\Delta u = f \quad \text{in } \Omega$$
$$u = 0 \quad \text{on } \partial \Omega,$$

where $\Omega = [0,1]^2$, $H = L^2(\Omega)$ or $H^1(\Omega)$, $u : \Omega \to \mathbb{R}$ is the unknown temperature, $f \in H$ is
the unknown control function acting as source term ($f(x) > 0$ corresponds to heating and
$f(x) < 0$ corresponds to cooling), $d \in L^2(\Omega)$ is the given desired temperature profile and
$\alpha \in [0, \infty)$ is a Tikhonov regularisation parameter. For a proof of existence and uniqueness
of this optimisation problem, we refer to section 1.5 of [19].

All implementations apply the limited memory BFGS method in order to approximate
Hessians using information from up to a certain number of previous iterations. The L-
BFGS-B algorithm, derived in [4] and implemented as part of Scipy’s optimisation package
[11], is a limited memory BFGS algorithm with support for bounds on the controls. The
Limited Memory, Variable Metric (LMVM) method from the TAO library [2] uses a BFGS
approximation with a More-Thuente line search. The interior point method described in
[20] and implemented in IPOPT [12] is an interior point line search filter method. Here, the
BFGS method approximates the Hessian of the Lagrangian and not of the functional itself.

These optimisation method implementations are insensitive towards changes in the in-
er product space, but instead assume an Euclidean space. This compares to the BFGS
implementation from Moola ([18]), that allows the user to specify the inner products. By applying Riesz maps associated to the underlying Hilbert space, the Moola respects inner products.

In table (2), iteration counts for the different algorithm implementations solving (55) subject to (56) and (57) are displayed for $H = L^2$ and $H = H^1$, respectively. In both cases, the inner product insensitive methods exhibit mesh dependence. This can be seen from the increase in iteration numbers for increasing ratio between largest and smallest element size $h_{\text{max}}/h_{\text{min}}$, reflecting the non-uniformity in the discretisation. Iteration counts increase roughly by factors between 4-25 when comparing the least and most non-uniform meshes. Relating this to large scale optimisation problems, it is obvious that the ability of efficiently dealing with non-uniformity in the discretisation can very well decide upon the computational feasibility of solving the problem. The optimisation method that respects the inner product of the control space is approximately mesh independent.

Table 2: Iterations of different common optimisation method implementations in Python using different inner products. The error threshold is given by $\varepsilon = 10^{-7}$ in the gradient norm $\|R_H(f')\|_H$. History length for Hessian approximation equal to 10, except for LMVM in TAO, where it is equal to 5.

| inner product | Implementation | $h_{\text{max}}/h_{\text{min}}$ | 4   | 8   | 16  | 32  | 64  | 128 |
|---------------|----------------|-------------------------------|-----|-----|-----|-----|-----|-----|
| $\ell^2$      | Scipy          | BFGS                          | 27  | 47  | 75  | 97  | 133 | 189 |
| $\ell^2$      | TAO            | LMVM                          | 42  | 60  | 77  | 111 | 131 | 155 |
| $\ell^2$      | IPOPT          | Interior Point                | 28  | 38  | 65  | 88  | 117 | 139 |
| $L^2$         | Moola          | BFGS                          | 22  | 20  | 22  | 23  | 23  | 27  |

(a) $H = L^2(\Omega)$

| inner product | Implementation | $h_{\text{max}}/h_{\text{min}}$ | 4   | 8   | 16  | 32  | 64  | 128 |
|---------------|----------------|-------------------------------|-----|-----|-----|-----|-----|-----|
| $\ell^2$      | Scipy          | BFGS                          | 25  | 53  | 118 | 222 | 244 | 594 |
| $\ell^2$      | TAO            | LMVM                          | 38  | 38  | 110 | 144 | 141 | 384 |
| $\ell^2$      | IPOPT          | Interior Point                | 31  | 38  | 85  | 127 | 182 | 443 |
| $H^1$         | Moola          | BFGS                          | 4   | 4   | 4   | 4   | 4   | 4   |

(b) $H = H^1(\Omega)$

9. Conclusion. This work can be seen as a first step in quantifying the computational expense of disrespecting inner products. It illustrates the importance of employing the inner product induced by the control space for optimisations by deriving a rigorous estimate of the cost of failing to do so for a simple and generic problem. Simulations confirm the derived dependency of iteration counts mesh non-uniformity, while similar numerical results are found for a more complex PDE constrained optimisation problem. These results provide an intuitive and analytically coherent model of mesh dependence and its corresponding errors for this class of problems. The ability to employ mesh refinement to resolve features of interest is a core advantage of the finite element method. Effectively dealing with non-uniformity in the mesh should therefore be a core capability of the optimisation method.
Acknowledgements. The authors would like to thank Colin J. Cotter and Magne Nor-ndaas for valuable discussions and collaboration. This work was supported by the Grantham Institute for Climate Change, the Engineering and Physical Sciences Research Council [Mathematics of Planet Earth CDT, grant number EP/L010613/1], the Natural Environment Research Council [grant number NE/K008951/1], and the Research Council of Norway through a Centres of Excellence grant to the Center for Biomedical Computing at Simula Research Laboratory (project number 179578).

REFERENCES

[1] MATLAB and Optimization Toolbox Release 2016a, The MathWorks, Inc., Natick, Massachusetts, United States. http://mathworks.com, 2016.
[2] S. J. Benson, L. C. McInnes, J. More, and J. Sarich, TAO user manual (revision 1.8), tech. report, ANL/MCS-TM-242, Mathematics and Computer Science Division, Argonne National Laboratory, 2005.
[3] S. C. Brenner and R. Scott, The mathematical theory of finite element methods, vol. 15, Springer Science & Business Media, 2008.
[4] R. H. Byrd, P. Lu, J. Nocedal, and C. Zhu, A limited memory algorithm for bound constrained optimization, SIAM Journal on Scientific Computing, 16 (1995), pp. 1190–1208.
[5] P. G. Ciarlet, The finite element method for elliptic problems, vol. 40, SIAM, 2002.
[6] S. C. Brenner and R. Scott, The mathematical theory of finite element methods, vol. 15, Springer Science & Business Media, 2008.
[7] I. Fried, Bounds on the extremal eigenvalues of the finite element stiffness and mass matrices and their spectral condition number, Journal of Sound and Vibration, 22 (1972), pp. 407–418.
[8] S. W. Funke and P. E. Farrell, A framework for automated PDE-constrained optimisation, arXiv preprint arXiv:1302.3894, (2013).
[9] P. E. Gill, W. Murray, and M. A. Saunders, SNOPT: An SQP algorithm for large-scale constrained optimization, SIAM review, 47 (2005), pp. 99–131.
[10] A. G"unnel, R. Herzog, and E. Sachs, A note on preconditioners and scalar products in Krylov subspace methods for self-adjoint problems in Hilbert space, Electronic Transactions on Numerical Analysis, 41 (2014), pp. 13–20.
[11] E. Jones, T. Oliphant, P. Peterson, et al., SciPy: Open source scientific tools for Python. http://www.scipy.org, 2001–.
[12] Y. Kawahara, C. Laird, S. Vigerske, and A. Wächter, Introduction to Ipopt: A tutorial for downloading, installing, and using Ipopt. https://projects.coin-or.org/Ipopt/, 2015.
[13] R. C. Kirby, From functional analysis to iterative methods, SIAM review, 52 (2010), pp. 269–293.
[14] A. Logg, K.-A. Mardal, and G. Wells, Automated solution of differential equations by the finite element method: The FEniCS book, vol. 84, Springer Science & Business Media, 2012.
[15] D. G. Luenberger and Y. Ye, Linear and nonlinear programming, vol. 116, Springer Science & Business Media, 2008.
[16] K.-A. Mardal and R. Winther, Preconditioning discretizations of systems of partial differential equations, Numerical Linear Algebra with Applications, 18 (2011), pp. 1–40.
[17] J. Nocedal and S. Wright, Numerical optimization, Springer Science & Business Media, 2006.
[18] M. Nordaas and S. W. Funke, The Moola optimisation package. https://github.com/funsim/moola, 2016.
[19] R. Pinnau and M. Ulbrich, Optimization with PDE constraints, vol. 23, Springer Science & Business Media, 2008.
[20] A. Wächter and L. T. Biegler, On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming, Mathematical programming, 106 (2006), pp. 25–57.
[21] J. Young, Optimojoe: An open source software library designed to solve general purpose nonlinear optimization problems. www.optimojoe.com, Open source software, 2016.