Electronic Journal of Differential Equations, Vol. 2008(2008), No. 42, pp. 1–31.
ISSN: 1072-6691. URL: http://ejde.math.txstate.edu or http://ejde.math.unt.edu
ftp ejde.math.txstate.edu (login: ftp)

SOBOLEV GRADIENTS FOR DIFFERENTIAL ALGEBRAIC EQUATIONS

ROBIN NITTKA, MANFRED SAUTER

Abstract. Sobolev gradients and weighted Sobolev gradients have been used for the solution of a variety of ordinary as well as partial differential equations. In the article at hand we apply this method to linear and non-linear ordinary differential algebraic equations and construct suitable gradients for such problems based on a new generic weighting scheme. We explain how they can be put into practice. In the last part, we discuss the performance of our publicly available implementation on some differential algebraic equations and present further applications.

1. Introduction

Differential algebraic equations (DAE) have a wide range of applications. Inter alia they appear in electrical circuit simulation [13], control theory [19], and in models of mechanical systems [39], to name only a few prominent examples. Recently, a considerable amount of research has been put into this field, see [20] and references therein. The general formulation of a DAE is

$$f(t, u(t), u'(t)) = 0, \quad t \in (0, T) \tag{1.1}$$

with some function $f\colon \mathbb{R} \times \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}^m$ whose partial derivative with respect to the third argument may be singular. A sufficiently smooth function $u\colon (0, T) \to \mathbb{R}^n$ satisfying (1.1) is called *solution of the DAE*. Though looking similar to ordinary differential equations, differential algebraic equations show fundamental differences in many aspects. Even for linear problems the solution space can be infinite dimensional, and initial conditions do in general not impose uniqueness. Furthermore, initial conditions might not admit a local solution, and guessing feasible initial conditions is virtually impossible in many cases [5, Subsection 5.3.4].

Computing a numerical solution to a given DAE is a very delicate task. The algebraically coupled equations arising in various engineering fields tend to be numerically difficult [10]. For example thermo-fluid systems are naturally described by high index DAE [15], as are DAE resulting from batch distillation process modeling [14]. Most methods need antecedent transformations reducing the coupling.

2000 *Mathematics Subject Classification.* 65L80, 41A60, 34A09.
Key words and phrases. Differential algebraic equations; weighted Sobolev gradients; steepest descent; non-linear least squares; consistent initial conditions.

Submitted March 4, 2008. Published March 20, 2008.

Those *index reductions* are complicated and can introduce considerable numerical error by themselves [20, Chapter 6]. Additionally, the special structure of the problem is often lost.

Here we present an alternative way to deal with DAE that has several significant advantages over the usual approaches. We use a steepest descent method based on Sobolev gradients to minimize an error functional in an appropriate function space. This very general method has been successfully employed to treat Ginzburg-Landau equations for superconductivity, conservation equations, minimal flow problems and minimal surface problems, among others [31]. The theoretical framework of Sobolev steepest descent was first presented by John W. Neuberger [30]. Our method treats the given DAE directly, without differentiation or other prior transformations. Furthermore, it is not necessary to impose the initial values, which is a great advantage over the step-by-step methods that are usually employed. But in case one wants to impose supplementary conditions, this is possible with little additional theoretical effort. For example, it is possible to solve initial value or boundary value problems. The only other software for solving differential algebraic boundary value problems we know of is Ascher's and Spiteri's `COLDAE` [3].

We propose steepest descent methods using weighted Sobolev gradients for the numerical treatment of DAE. In section 2 we provide the underlying theory. In section 3 we take an operator theoretical point of view to introduce a space which seemingly has not been considered before in this generality within the literature about Sobolev steepest descent. We prove that, in a sense motivated by section 2.1, this space which is related the problem itself has advantages over the usual Sobolev spaces. We continue this idea in section 4 where we explain that it is superior also in some other sense involving the Fredholm property. In section 5 we show how various Sobolev gradients can be applied to fully non-linear DAE, following the usual ideas as well as generalizing the concept of section 3. In section 6 we discuss the discretization techniques used for the numerics, also covering non-linear problems and supplementary conditions. Section 7 contains details of our publicly available implementation [32] and shows, via tables and plots, how our program behaves on some intricate examples. Finally, section 8 summarizes our results.

2. Sobolev Steepest Descent

In section 2.1 we list some basic facts about the theory of Sobolev gradients. For details we refer to John W. Neuberger's monograph [31]. In section 2.2 we focus on the basic case of linear DAE with constant coefficients. The general form of this equation is

$$M_1 u'(t) + M_2 u(t) = b(t), \quad t \in (0,T), \tag{2.1}$$

where $M_1, M_2 \in \mathbb{R}^{m\times n}$ are constant matrices. The function $b \in L^2(0,T;\mathbb{R}^m)$ is called the *inhomogeneity* or *right hand side*. We look for weak solutions in $L^2(0,T;\mathbb{R}^n)$.

2.1. General Setting.

Let V and H be Hilbert spaces, $A \in \mathscr{L}(V,H)$, and $b \in H$. Usually, A is a differential operator and V an appropriate Sobolev space. We are looking for solutions $u \in V$ of the equation $Au = b$.

The (continuous) Sobolev gradient approach to this problem is the following. Define the quadratic functional

$$\psi\colon V \to \mathbb{R}_+,\ u \mapsto \tfrac{1}{2}\,\|Au-b\|_H^2$$

and try to find a zero (or at least a minimizer) by steepest descent, i. e., by solving the Hilbert space valued ordinary differential equation

$$\dot{\varphi}(t) = -\nabla\psi\left(\varphi(t)\right), \quad \varphi(0) = u_0 \tag{2.2}$$

for an arbitrary initial estimate $u_0 \in V$ and letting $t \to \infty$. Here $\nabla\psi(u)$ denotes the unique representation of the Fréchet derivative $\psi'(u) \in V'$ as a vector in V whose existence is guaranteed by the Riesz-Fréchet representation theorem. The derivative of ψ is

$$\langle \psi'(u), h\rangle = (Au - b \mid Ah)_H = (A^*Au - A^*b \mid h)_V, \tag{2.3}$$

hence

$$\nabla\psi(u) = A^*Au - A^*b.$$

In [31, Theorems 4–6], the following facts are proved.

Theorem 2.1. *If $b \in \operatorname{Rg} A$, then $\varphi(t)$ defined in (2.2) converges to some $\omega \in V$ in the norm of V, and $A\omega = b$. The vector ω is the zero of ψ nearest to u_0 in the metric of V. Furthermore, for every $b \in H$ the images $A\varphi(t)$ converge to $P_{\overline{\operatorname{Rg} A}}b$ as $t \to \infty$, i. e., to the orthogonal projection of b onto the closure of the range of A.*

Thus, we can characterize convergence of $\varphi(t)$ in terms of the range of A.

Corollary 2.2. *There exists a global solution φ to the differential equation (2.2). The trajectory $(\varphi(t))_{t\in\mathbb{R}_+}$ converges in V if and only if*

$$P_{\overline{\operatorname{Rg} A}}b \in \operatorname{Rg} A. \tag{2.4}$$

Then the limit is the solution of the problem $Au = P_{\overline{\operatorname{Rg} A}}b$ with minimal distance to u_0.

Proof. First note that the unique solution of equation (2.2) is

$$\varphi(t) = \mathrm{e}^{-tA^*A}u_0 + \int_0^t \mathrm{e}^{-(t-s)A^*A}A^*b \,\mathrm{d}s.$$

Using the decomposition $b = P_{\overline{\operatorname{Rg} A}}b + P_{\ker A^*}b$, we see that $\varphi(t)$ depends only on $P_{\overline{\operatorname{Rg} A}}b$, not on b itself. Replacing b by its projection onto $\overline{\operatorname{Rg} A}$, theorem 2.1 asserts that under condition (2.4) the steepest descent converges and the limit has the claimed property.

For the converse implication, assume that $\varphi(t)$ converges to some $\omega \in V$. Then

$$A\omega \leftarrow A\varphi(t) \to P_{\overline{\operatorname{Rg} A}}b$$

by theorem 2.1 and continuity of A. Hence $P_{\overline{\operatorname{Rg} A}}b \in \operatorname{Rg} A$, and thus condition (2.4) is fulfilled. □

The corollary shows in particular that the operator A has closed range if and only if $\varphi(t)$ converges for every $b \in H$. But if $\operatorname{Rg} A$ is not closed, then arbitrarily small perturbations of b in the norm of H alter the convergence behavior. However, it can be proved that $\dot{\varphi}(t) \to 0$ for every $b \in H$ if ψ is non-negative and convex.

2.2. **Application to differential algebraic equations.** Now we turn to linear, autonomous, first-order DAE, allowing time-dependent inhomogeneities. This means we fix matrices $M_1, M_2 \in \mathbb{R}^{m\times n}$ such that $\ker M_1 \neq \{0\}$ and a function $b \in L^2(0,T;\mathbb{R}^m)$ and consider the DAE

$$M_1 u' + M_2 u = b, \quad u \in H^1(0,T;\mathbb{R}^n). \tag{2.5}$$

For $V := H^1(0,T;\mathbb{R}^n)$, $H := L^2(0,T;\mathbb{R}^m)$ and $Au := M_1u' + M_2u$ this fits into the framework of section 2.1. For convenience, we frequently identify a matrix $M \in \mathbb{R}^{m\times n}$ with a bounded linear operator from $L^2(0,T;\mathbb{R}^n)$ to $L^2(0,T;\mathbb{R}^m)$ acting as $(Mu)(x) := M(u(x))$. It is obvious that these operators map $H^1(0,T;\mathbb{R}^n)$ into $H^1(0,T;\mathbb{R}^m)$.

We already have discovered that the steepest descent converges whenever there is a solution to converge to—and then it picks the nearest solution. But even if there is no solution the steepest descent might converge. As we have seen in corollary 2.2 this happens if and only if $P_{\overline{\operatorname{Rg} A}} b \in \operatorname{Rg} A$ for the given $b \in L^2(0,T;\mathbb{R}^m)$. Hence it is natural to ask whether $\operatorname{Rg} A$ is closed because then the steepest descent converges for every b. Unfortunately, in general this is not the case as the following necessary condition shows.

Proposition 2.3. *If the operator A defined above has closed range, then*

$$\operatorname{Rg}(M_2|_{\ker M_1}) \subset \operatorname{Rg} M_1. \tag{2.6}$$

In other words, if A has closed range, then M_2 maps $\ker M_1$ into $\operatorname{Rg} M_1$.

For the proof we use the following simple observation.

Lemma 2.4. *Let V be a subspace of $\mathbb{R}^n$ and assume that $u \in H^1(0,T;\mathbb{R}^n)$ satisfies $u(x) \in V$ for almost every $x \in (0,T)$. Then $u'(x) \in V$ for almost every $x \in (0,T)$.*

Proof. Let P_V denote a projection of $\mathbb{R}^n$ onto V. We consider P_V also as an operator on $H^1(0,T;\mathbb{R}^n)$ defined by pointwise application. Then linearity of differentiation yields

$$u'(x) = (P_V u)'(x) = P_V u'(x) \in V$$

for almost every $x \in (0,T)$. This proves the claim. □

Proof of proposition 2.3. Assume that $\operatorname{Rg} A$ is closed and that condition (2.6) does not hold, i. e., that there exists a vector $e \in \ker M_1 \subset \mathbb{R}^N$ such that $M_2 e \notin \operatorname{Rg} M_1$. Fix any sequence (v_k) in $H^1(0,T)$ converging to a function $v \in L^2(0,T) \setminus H^1(0,T)$ in the norm of $L^2(0,T)$ and define $u_k := v_k e$. Then $v_k M_2 e = Au_k \in \operatorname{Rg} A$ for all $k \in \mathbb{N}$ by lemma 2.4, hence $vM_2e = \lim v_k M_2 e \in \overline{\operatorname{Rg} A}$. Since we assumed that $\operatorname{Rg} A = \overline{\operatorname{Rg} A}$ there exists $u \in H^1(0,T;\mathbb{R}^n)$ such that $Au = vM_2e$. We decompose

$$u = u_1 + u_2, \quad \text{where } u_1 := P_{(\ker M_1)^\perp} u \text{ and } u_2 := P_{\ker M_1} u,$$

and note that $u_2' \in \ker M_1$ almost everywhere by the above lemma, whence

$$vM_2e = Au = M_1u_1' + M_2u.$$

Now fix a row vector $q \in \mathbb{R}^{1\times m}$ satisfying $qM_2e = 1$ and $qM_1 = 0$. Such a vector exists because e is chosen such that $\operatorname{span}\{M_2e\} \cap \operatorname{Rg} M_1 = \{0\}$. Finally, observe that

$$qM_2u = q(vM_2e - M_1u_1') = v \notin H^1(0,T),$$

contradicting $u \in H^1(0,T;\mathbb{R}^n)$. □

The following simple examples of DAE show the different behavior that may occur regarding the closedness of the range. We will revisit them in section 3.

Example 2.5. Let $M_1 := \left(\begin{smallmatrix} 0 & 0 \\ 1 & 0 \end{smallmatrix}\right)$ and $M_2 := \left(\begin{smallmatrix} 0 & 0 \\ 0 & 1 \end{smallmatrix}\right)$. Then $A\left(\begin{smallmatrix} u \\ v \end{smallmatrix}\right) = \left(\begin{smallmatrix} 0 \\ u'+v \end{smallmatrix}\right)$, whence $\operatorname{Rg} A = \left\{\left(\begin{smallmatrix} 0 \\ f \end{smallmatrix}\right) : f \in L^2(0,T)\right\}$ is closed.

Example 2.6. Let $M_1 := \left(\begin{smallmatrix} 1 & 0 \\ 1 & 0 \end{smallmatrix}\right)$ and $M_2 := \left(\begin{smallmatrix} 0 & 0 \\ 0 & 1 \end{smallmatrix}\right)$. Then $A\left(\begin{smallmatrix} u \\ v \end{smallmatrix}\right) = \left(\begin{smallmatrix} u' \\ u'+v \end{smallmatrix}\right)$. Proposition 2.3 shows that $\operatorname{Rg} A$ is not closed.

Example 2.7. Let $M_1 := \left(\begin{smallmatrix} 0 & 1 \\ 0 & 0 \end{smallmatrix}\right)$ and $M_2 := \left(\begin{smallmatrix} 1 & 0 \\ 0 & 1 \end{smallmatrix}\right)$. Then $A\left(\begin{smallmatrix} u \\ v \end{smallmatrix}\right) = \left(\begin{smallmatrix} v'+u \\ v \end{smallmatrix}\right)$. We will prove later that $\operatorname{Rg} A$ is not closed, see example 3.6. We point out, however, that this does not follow from proposition 2.3.

As we have seen, we cannot expect the steepest descent to converge for any right hand side b. But some regularity assumption on b might ensure convergence. More precisely, the authors suggest to investigate whether $b \in H^1(0,T;\mathbb{R}^m)$ implies $P_{\overline{\operatorname{Rg} A}} b \in \operatorname{Rg} A$.

3. Closedness

Considering $V = H^1(0,T;\mathbb{R}^n)$ as done in section 2 is natural since this space is the maximal subspace of $L^2(0,T;\mathbb{R}^n)$ for which u' can be defined. However, noting that the equation $M_1u' + M_2u = b$ can also be written as $(M_1u)' + M_2u = b$, we see that it suffices to require M_1u to be in $H^1(0,T;\mathbb{R}^m)$, which may be the case even if $u \notin H^1(0,T;\mathbb{R}^n)$. More precisely, the part of u in $\ker M_1$ is allowed to be only L^2 instead of H^1. Indeed, the following lemma shows that this describes the maximal subspace of $L^2(0,T;\mathbb{R}^n)$ to which A can be extended.

Proposition 3.1. *Define*

$$D(\bar{A}) := \left\{u \in L^2(0,T;\mathbb{R}^n) : M_1u \in H^1(0,T;\mathbb{R}^m)\right\} \subset L^2(0,T;\mathbb{R}^n),$$
$$\bar{A}u := (M_1u)' + M_2u.$$

Then the operator $\bar{A} : L^2(0,T;\mathbb{R}^n) \supset D(\bar{A}) \to L^2(0,T;\mathbb{R}^m)$ is closed. It is the closure of the operator $A : L^2(0,T;\mathbb{R}^n) \supset H^1(0,T;\mathbb{R}^n) \to L^2(0,T;\mathbb{R}^m)$ defined in section 2.2.

Proof. Denote $V := D(\bar{A})$. To show that $\bar{A}$ is closed, fix a sequence (u_k) in V converging in the norm of $L^2(0,T;\mathbb{R}^n)$ to a function u such that $\bar{A}u_k$ converges to a function v in $L^2(0,T;\mathbb{R}^m)$. We have to prove that $u \in V$ and $\bar{A}u = v$. Define $w_k := M_1u_k \in H^1(0,T;\mathbb{R}^m)$. Then $w_k \to M_1u$ in $L^2(0,T;\mathbb{R}^n)$ and

$$\bar{A}u_k = w_k' + M_2u_k \to v \text{ in } L^2(0,T;\mathbb{R}^m),$$

hence

$$w_k' \to v - M_2u \text{ in } L^2(0,T;\mathbb{R}^m).$$

The differentiation operator on $L^2(0,T;\mathbb{R}^m)$ with domain $H^1(0,T;\mathbb{R}^m)$ is closed, hence $w_k \to M_1u$ and $w_k' \to v - M_2u$ implies that $M_1u \in H^1(0,T;\mathbb{R}^n)$ and $(M_1u)' = v - M_2u$. This means precisely that $u \in V$ and $\bar{A}u = v$. We have shown that $\bar{A}$ is closed.

Now let P be a projection of $\mathbb{R}^n$ onto $\ker M_1$. We claim that

$$V = \left\{u \in L^2(0,T;\mathbb{R}^n) : (I-P)u \in H^1(0,T;\mathbb{R}^n)\right\}. \tag{3.1}$$

To see this, note that the restriction $\widetilde{M_1}\colon \operatorname{Rg}(I-P) \to \operatorname{Rg} M_1$ of M_1 to $\operatorname{Rg}(I-P)$ is invertible and satisfies $\widetilde{M_1}^{-1} M_1 u = (I-P)u$. This shows that $(I-P)u$ is in $H^1(0,T;\mathbb{R}^n)$ whenever $M_1 u$ is in $H^1(0,T;\mathbb{R}^m)$. The other inclusion similarly follows from $M_1 u = M_1(I-P)u$.

To show that $\bar{A}$ is the closure of A, for each $u \in V$ we have to find a sequence $(u_k) \subset H^1(0,T;\mathbb{R}^n)$ such that $u_k \to u$ in $L^2(0,T;\mathbb{R}^n)$ and $Au_k \to \bar{A}u$ in $L^2(0,T;\mathbb{R}^m)$. Fix $u \in V$ and define $w := (I-P)u$ and $v := Pu$. The representation (3.1) shows that $w \in H^1(0,T;\mathbb{R}^n)$. Since $H^1(0,T;\mathbb{R}^n)$ is dense in $L^2(0,T;\mathbb{R}^n)$, there exists a sequence (v_k) in $H^1(0,T;\mathbb{R}^n)$ which converges to v in $L^2(0,T;\mathbb{R}^n)$. Define $u_k := w + Pv_k \in H^1(0,T;\mathbb{R}^n)$. Then $u_k \to w + Pv = w + v = u$ in $L^2(0,T;\mathbb{R}^n)$, thus

$$Au_k = M_1 w' + M_2 u_k \to M_1 w' + M_2 u = \bar{A}u \text{ in } L^2(0,T;\mathbb{R}^m).$$

This shows that (u_k) is a sequence with the desired property. □

The following corollary restates the closedness of $\bar{A}$ in different words, using a well-known characterization of closed operators.

Corollary 3.2. *The space $V := D(\bar{A})$ equipped with the inner product*

$$(u \mid v)_V := (u \mid v)_{L^2(0,T;\mathbb{R}^n)} + \big(\bar{A}u \mid \bar{A}v\big)_{L^2(0,T;\mathbb{R}^m)}$$

is a Hilbert space. The operator $\bar{A}\colon V \to L^2(0,T;\mathbb{R}^m)$ is bounded.

This shows how to apply the method of steepest descent to the operator $\bar{A}$. In general, this will lead to trajectories and limits which are different from those obtained by the approach in section 2, since $\nabla\psi$ is taken with respect to some other inner product. So the question arises which space should be used (also compare to section 6.2). The next corollary shows that from a theoretical point of view the situation improves if $H^1(0,T;\mathbb{R}^n)$ is replaced with V.

Lemma 3.3. *Let $A\colon X \supset D(A) \to Y$ be a closable operator, and let $\bar{A}$ be its closure. Then $\operatorname{Rg} A \subset \operatorname{Rg} \bar{A} \subset \overline{\operatorname{Rg} A}$. In particular, if $\operatorname{Rg} A$ is closed, then $\operatorname{Rg} \bar{A}$ is closed.*

Proof. The first inclusion is obvious since $\bar{A}$ extends A. Now let $y \in \operatorname{Rg} \bar{A}$. Then there exists $x \in D(\bar{A})$ such that $\bar{A}x = y$. By definition of the closure there exists a sequence $(x_n) \subset D(A)$ such that $x_n \to x$ in X and $Ax_n \to \bar{A}x = y$ in Y. But this proves that y is a limit of a sequence in $\operatorname{Rg} A$, hence $y \in \overline{\operatorname{Rg} A}$. □

Corollary 3.4. *Let $b \in L^2(0,T;\mathbb{R}^m)$ and consider problem (2.1). If the steepest descent with respect to the inner product in $H^1(0,T;\mathbb{R}^n)$ converges for the right hand side b, then the steepest descent with respect to the inner product from corollary 3.2 converges for that right hand side as well.*

Proof. This follows from corollary 2.2 combined with lemma 3.3. □

To illustrate that using of $\bar{A}$ instead of A may improve the situation, but not always does, we again consider the examples of section 2.2. Here again, A refers to the operator defined in section 2.2, whereas $\bar{A}$ and V are as in corollary 3.2. The examples also show that relation (2.6) is independent of $\operatorname{Rg} \bar{A}$ being closed.

Example 3.5. Let M_1 and M_2 be as in example 2.6. Then

$$V = \Big\{ \begin{pmatrix} u \\ v \end{pmatrix} : u \in H^1(0,T),\ v \in L^2(0,T) \Big\},$$
$$\operatorname{Rg} \bar{A} = \Big\{ \begin{pmatrix} u' \\ u' + v \end{pmatrix} : u \in H^1(0,T),\ v \in L^2(0,T) \Big\} = L^2(0,T;\mathbb{R}^2).$$

We used that every function in $L^2(0,T)$ is the derivative of a function in $H^1(0,T)$. This shows that $\bar{A}$ is surjective. In particular $\operatorname{Rg} \bar{A}$ is closed, whereas $\operatorname{Rg} A$ is not as seen in example 2.6.

Example 3.6. Consider again the matrices M_1 and M_2 from example 2.7. Then

$$V = \Big\{ \begin{pmatrix} u \\ v \end{pmatrix} : u \in L^2(0,T),\ v \in H^1(0,T) \Big\},$$
$$\operatorname{Rg} \bar{A} = \Big\{ \begin{pmatrix} v' + u \\ v \end{pmatrix} : u \in L^2(0,T),\ v \in H^1(0,T) \Big\} = L^2(0,T) \times H^1(0,T).$$

Hence $\operatorname{Rg} \bar{A}$ is dense in $L^2(0,T;\mathbb{R}^2)$, but not closed. By lemma 3.3 this implies that also $\operatorname{Rg} A$ is not closed. This proves the claim of example 2.7.

4. Fredholm Property

Assuming that there exists a solution of (2.5) we are interested in the convergence behavior of the Sobolev steepest descent. For example the so-called Łojasiewicz-Simon inequality can be used to investigate the rate of convergence [17]. On the other hand, for the non-linear case treated in the next section a special instance of this inequality has been used to prove convergence for arbitrary initial estimates [31, Section 4.2].

A particularly simple method to show that a Łojasiewicz-Simon inequality holds locally near a critical point $u_0 \in V$ is by checking that $\psi''(u_0) = A^*A$ is a Fredholm operator [12, Corollary 3]. Unfortunately, theorem 4.2 shows that we never are in this situation when A is the operator of section 2. This fact is interesting in its own right. Of course this does not mean that the Łojasiewicz-Simon inequality cannot be fulfilled for any steepest descent coming from a DAE; we give an example at the end of the section.

Lemma 4.1. *Let $D\colon H^1(0,T) \to L^2(0,T)$, $u \mapsto u'$. Then $D^*D = I - (I - \Delta_N)^{-1}$, where Δ_N denotes the Neumann Laplacian $\Delta_N u = u''$ with domain*

$$D(\Delta_N) = \left\{ u \in H^2(0,T) : u'(0) = u'(T) = 0 \right\}.$$

Proof. By definition, $(D^*Du \mid v)_{H^1} = (Du \mid Dv)_{L^2}$ for all $u, v \in H^1(0,T)$. Thus it suffices to show that

$$\begin{aligned} \int_0^T u'v' &\overset{!}{=} \left(\left(I - (I - \Delta_N)^{-1} \right) u \mid v \right)_{H^1} \\ &= \int_0^T uv + \int_0^T u'v' - \int_0^T \left((I - \Delta_N)^{-1} u \right) v - \int_0^T \left((I - \Delta_N)^{-1} u \right)' v'. \end{aligned}$$

This is an immediate consequence of the integration by parts formula, using that $(I-\Delta_N)^{-1}u \in D(\Delta_N)$. In fact,

$$\begin{aligned}\int_0^T \left((I-\Delta_N)^{-1}u\right)' v' &= \left((I-\Delta_N)^{-1}u\right)' v\big|_0^T - \int_0^T \left((I-\Delta_N)^{-1}u\right)'' v \\ &= \int_0^T \left((I-\Delta_N)(I-\Delta_N)^{-1}u - (I-\Delta_N)^{-1}u\right) v.\end{aligned}$$

Collecting the terms, the claimed identity follows. □

As the embedding of $H^2(0,T)$ into $H^1(0,T)$ is compact, the above lemma shows that D^*D is a compact perturbation of the identity. This result generalizes to $D\colon H^1(0,T;\mathbb{R}^n) \to L^2(0,T;\mathbb{R}^n)$, $u \mapsto u'$ by considering every component separately.

Theorem 4.2. *Consider the operator $A\colon H^1(0,T;\mathbb{R}^n) \to L^2(0,T;\mathbb{R}^m)$ defined by $A := DM_1 + \iota M_2$ as introduced in section 2. Here the matrices M_1 and M_2 act as operators from $H^1(0,T;\mathbb{R}^n)$ into $H^1(0,T;\mathbb{R}^m)$, and the differentiation D and the embedding ι map from $H^1(0,T;\mathbb{R}^n)$ into $L^2(0,T;\mathbb{R}^n)$. Then $A^*A = M_1^TM_1 + K$ for a compact operator K acting on $H^1(0,T;\mathbb{R}^n)$ which shows that A^*A is a Fredholm operator if and only if* $\ker M_1 = \{0\}$.

Proof. The embedding ι is compact, hence also ι^* is a compact operator. By lemma 4.1, $D^*D = I + \tilde{K}$ for a compact operator $\tilde{K}$. Using the ideal property of compact operators, we obtain

$$A^*A = M_1^*M_1 + K = M_1^TM_1 + K$$

for a compact operator K on $H^1(0,T;\mathbb{R}^n)$. Because compact perturbations of Fredholm operators remain Fredholm operators [1, Corollary 4.47], A^*A is a Fredholm operator if and only if $M_1^TM_1$ is. If M_1 has trivial kernel, then $M_1^TM_1$ is invertible and hence a Fredholm operator. If on the other hand $\ker M_1 \neq \{0\}$, then $\dim\ker M_1^TM_1 = \infty$ as an operator on $H^1(0,T;\mathbb{R}^n)$, implying that $M_1^TM_1$ is not a Fredholm operator. □

However, the next example shows that $\bar{A}^*\bar{A}$ might be a Fredholm operator even though A^*A is not. This shows that also in this sense we can improve the situation by replacing A by $\bar{A}$.

Example 4.3. For $M_1 := \left(\begin{smallmatrix}1&0\\0&0\end{smallmatrix}\right)$ and $M_2 := \left(\begin{smallmatrix}0&0\\0&1\end{smallmatrix}\right)$ let $\bar{A}$ be defined as in proposition 3.1. It is easy to check that

$$\ker\bar{A} = \left\{\begin{pmatrix}u\\0\end{pmatrix} : u \equiv c \in \mathbb{R}\right\} \quad\text{and}\quad \operatorname{Rg}\bar{A} = L^2(0,T)\times L^2(0,T),$$

proving that $\bar{A}$ is a Fredholm operator of index 1. This shows that also $\bar{A}^*\bar{A}$ is a Fredholm operator, see [1, Theorems 4.42 and 4.43].

On the other hand, $\bar{A}^*\bar{A}$ is not necessarily a Fredholm operator, e. g. it is not for $M_1 := \begin{pmatrix}1&0\end{pmatrix}$ and $M_2 := \begin{pmatrix}0&1\end{pmatrix}$. It would be useful to have a characterization of $\bar{A}^*\bar{A}$ being a Fredholm operator in terms of the matrices M_1 and M_2. This would provide a tool to investigate the rate of convergence of the steepest descent.

5. The Non-Linear Case

In this section we consider the general, fully non-linear first order DAE

$$f(t, u(t), u'(t)) = 0 \tag{5.1}$$

where $f\colon [0,T] \times \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}^m$. We treat this case in utmost generality, not caring about convergence. Instead, we focus on the theoretical background needed to formulate various steepest descent equations corresponding to the gradients introduced in sections 2 and 3.

We need to formulate the DAE (5.1) in a functional analytic way in order to apply Sobolev gradient methods. We want to define the operator

$$F\colon H^1(0,T;\mathbb{R}^n) \to L^2(0,T;\mathbb{R}^m),\ F(u) := t \mapsto f(t, u(t), u'(t)) \tag{5.2}$$

and to minimize the (non-linear) functional

$$\psi\colon H^1(0,T;\mathbb{R}^n) \to \mathbb{R},\ \psi(u) := \tfrac{1}{2}\|F(u)\|_2^2. \tag{5.3}$$

Such an operator F is frequently called *Nemytskii operator* [2, Chapter 1] or *differential operator* [4]. We require it to be well-defined and at least differentiable. This is the case if f fulfills certain regularity and growth conditions. For the sake of completeness, we prove a lemma of this kind. Similar conditions involving higher order partial derivatives can be found which guarantee F to be of higher regularity, for example of class C^2.

We say that a function $g\colon [0,T] \times \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}^m$ satisfies the growth assumption (G) if for every compact set $K \subset \mathbb{R}^n$ there exist constants $C, M > 0$ only depending on f, T and K such that

$$\forall t \in [0,T] \quad \forall x \in K \quad \forall y \in \mathbb{R}^n \quad |g(t,x,y)| \le C\,|y| + M \tag{G}$$

where $|\cdot|$ denotes a norm in $\mathbb{R}^m$ or $\mathbb{R}^n$, respectively. Similarly, we say that g satisfies the boundedness assumption (B) if for every compact set $K \subset \mathbb{R}^n$ there exists $L > 0$ only depending on f, T and K such that

$$\forall t \in [0,T] \quad \forall x \in K \quad \forall y \in \mathbb{R}^n \quad |g(t,x,y)| \le L. \tag{B}$$

Lemma 5.1. *Let $f\colon [0,T]\times\mathbb{R}^n\times\mathbb{R}^n \to \mathbb{R}^m$ be measurable, and denote its arguments by (t,x,y). Assume that f is of class C^2 with respect to (x,y). We denote the matrix-valued partial derivative of f with respect to x by f_x, and similarly for y and higher order partial derivatives. Assume that f, f_x, f_{xx} and f_{xy} satisfy* (G) *and that f_y and f_{yy} satisfy* (B). *Then F as in* (5.2) *is a mapping of class C^1 from $H^1(0,T;\mathbb{R}^n)$ to $L^2(0,T;\mathbb{R}^m)$, and its derivative at $u \in H^1(0,T;\mathbb{R}^n)$ applied to $h \in H^1(0,T;\mathbb{R}^n)$ is*

$$(F'(u)h)(t) = f_x(t, u(t), u'(t))h(t) + f_y(t, u(t), u'(t))h'(t) \tag{5.4}$$

for almost every $t \in [0,T]$.

Proof. Let $u \in H^1(0,T;\mathbb{R}^n)$ be arbitrary. As $H^1(0,T)$ continuously embeds into $\mathrm{C}[0,T]$, u can be chosen to be a continuous function. Thus there exists R such that $|u(t)| \le R$ for all $t \in [0,T]$. Let K be the closure of the ball $B(0, R+1)$. For this K, fix constants C, M and L simultaneously satisfying (G) and (B) for all the functions in the assumptions. The estimate $|F(u)(t)| \le C|u'(t)| + M$, $t \in [0,T]$,

shows $F(u) \in L^2(0,T;\mathbb{R}^m)$. Similarly, for $F'(u)$ defined by (5.4) we obtain

$$\begin{aligned}\|F'(u)h\|_2^2 &= \int |(F'(u)h)(t)|^2 \le \int 2\Big(\big(C|u'(t)| + M\big)^2 |h(t)|^2 + L^2|h'(t)|^2\Big) \\ &\le 4\left(C^2\|u'\|_2^2 + TM^2\right)\|h\|_\infty^2 + 2L^2\|h'\|_2^2 .\end{aligned}$$

Because $H^1(0,T)$ embeds into $L^\infty(0,T)$ continuously, this proves the boundedness of $F'(u)$ as an operator from $H^1(0,T;\mathbb{R}^n)$ to $L^2(0,T;\mathbb{R}^m)$.

Next, we show that $F'(u)$ is indeed the derivative of F at u. For every $t \in \mathbb{R}$ and $x, y \in \mathbb{R}^n$, denote by $o_{t,x,y}\colon \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}^m$ the error in the expansion

$$f(t, x+\varepsilon_1, y+\varepsilon_2) = f(t,x,y) + f_x(t,x,y)\varepsilon_1 + f_y(t,x,y)\varepsilon_2 + o_{t,x,y}(\varepsilon_1,\varepsilon_2)\Big|\begin{pmatrix}\varepsilon_1\\ \varepsilon_2\end{pmatrix}\Big|.$$

We have to show that the error

$$\Big(F(u+h)(t) - F(u)(t) - (F'(u)h)(t)\Big)\Big|\begin{pmatrix}h(t)\\ h'(t)\end{pmatrix}\Big|^{-1} = o_{t,u(t),u'(t)}(h(t),h'(t))$$

approaches zero as a function in t with respect to the norm of $L^2(0,T;\mathbb{R}^m)$ as h tends to zero in $H^1(0,T;\mathbb{R}^n)$. For this we employ the estimate

$$|g(x+h) - g(x) - g'(x)h| \le \sum_{i,j=1}^N \sup_{y\in[x,x+h]} \big|g_{x_ix_j}(y)\big|\,|h_i|\,|h_j|$$

for functions $g\colon \mathbb{R}^N \to \mathbb{R}$ of class C^2 which can be verified by iterated applications of the mean value theorem. Thus by the assumptions on the second derivatives, for small enough $h \in H^1(0,T;\mathbb{R}^n)$ we obtain that

$$\begin{aligned}\big|o_{t,u(t),u'(t)}(h(t),h'(t))\big| &\le \frac{\sup|f_{xx}|\,|h|^2 + 2\sup|f_{xy}|\,|h|\,|h'| + \sup|f_{yy}|\,|h'|^2}{(|h|^2+|h'|^2)^{1/2}} \\ &\le 3\big(C(|u'|+|h'|)+M\big)|h| + L|h'|\end{aligned}$$

for every $t \in [0,T]$. By similar arguments as above, this estimate shows that $o_{\cdot,u(\cdot),u'(\cdot)}(h(\cdot),h'(\cdot))$ goes to zero in $L^2(0,T;\mathbb{R}^m)$ as h tends to zero in $H^1(0,T;\mathbb{R}^n)$. This proves that $F'(u)$ is the derivative of F at u.

Finally, the continuity of the operator-valued function F' on $H^1(0,T;\mathbb{R}^n)$ can be proved in a similar manner. For this, we have to make use of the growth conditions on the second order derivatives. □

Remark. The lemma suffices for most applications. For example for quasi-linear problems, i. e., for $f(t,x,y) = g(t,x)y + h(t,x)$, and thus in particular for linear and semi-linear problems, the above assumptions are fulfilled whenever g and h are sufficiently smooth, independently of their growth behavior.

The assumptions on f can be weakened by imposing more regularity on the solution u as the following corollary shows.

Corollary 5.2. *Assume that $f\colon [0,T]\times\mathbb{R}^n\times\mathbb{R}^n$ is of class C^2. Then F defined as in (5.2) is a mapping of class C^1 from $H^2(0,T;\mathbb{R}^n)$ to $L^2(0,T;\mathbb{R}^m)$, and its derivative is as stated in equation (5.4).*

Proof. Since functions in $H^1(0,T)$ are bounded, the values of $f(t,u(t),u'(t))$ remain in a bounded set as t ranges over $[0,T]$ and u ranges over the unit ball in $H^2(0,T;\mathbb{R}^n)$, and the same statement holds for the partial derivatives. Using this fact, the arguments are similar to the proof of the lemma. □

However, it might happen that solutions of (5.1) are of class H^1 but not of class H^2, see for example equation (7.3) in section 7.4. In such cases we impose too much regularity when choosing this Sobolev space. For a general discussion about the technique of using spaces of higher order than strictly necessary for Sobolev descent methods, we refer to [31, Section 4.5].

For the moment, we assume that $F\colon H^1(0,T;\mathbb{R}^n) \to L^2(0,T;\mathbb{R}^n)$ is of class C^1. Later we will need higher regularity. By the chain rule, the derivative of ψ defined in (5.3) is

$$\psi'(u)h = (F(u) \mid F'(u)h)_{L^2} = (F'(u)^*F(u) \mid h)_{H^1}\,.$$

Analogously to the linear case, we define the H^1 *Sobolev gradient* as

$$\nabla_{H^1}\psi(u) = F'(u)^*F(u)$$

and consider trajectories of the corresponding steepest descent equation (2.2). It is possible to find sufficient conditions under which those trajectories converge to a solution of (5.1). In fact, this is one of the main topics in the monograph [31].

However, it is known that for some examples using a weighted Lebesgue measure for the computation of the Sobolev gradient—giving rise to *weighted Sobolev gradients*—improves the situation significantly, cf. [24, 25, 26, 27]. This complements our discussion in section 3 where we showed that the convergence behavior can be improved by choosing an inner product related to the problem itself. We now generalize the inner product considered in that section to the non-linear case. To this end, we equip $H^1(0,T;\mathbb{R}^n)$ with a variable inner product making it into a Riemannian manifold. A similar idea has been investigated by Karátson and Neuberger in a recent article [21] where they identify Newton's method as a steepest descent with respect to a certain variable inner product. The resulting method is quite similar to what we present here. However, they make assumptions which are not fulfilled in our case.

For the rest of this section, we make use of the notations of [22].

Lemma 5.3. *Let $F\colon H^1(0,T;\mathbb{R}^n) \to L^2(0,T;\mathbb{R}^m)$ be of class C^2. Choose $\lambda > 0$. Then the mapping*

$$g_2\colon H^1(0,T;\mathbb{R}^n) \to \mathcal{L}^2_{\mathrm{sym}}\left(H^1(0,T;\mathbb{R}^n)\right)$$

defined by

$$g_2(u) := \Big((f,g) \mapsto \lambda\,(f \mid g)_{H^1(0,T;\mathbb{R}^n)} + (F'(u)f \mid F'(u)g)_{L^2(0,T;\mathbb{R}^m)}\Big)$$

makes $H^1(0,T;\mathbb{R}^n)$ into an infinite dimensional Riemannian manifold.

Proof. We choose only one chart as the atlas of the manifold $X := H^1(0,T;\mathbb{R}^n)$, namely the identity mapping onto the Banach space $\mathbf{E} := H^1(0,T;\mathbb{R}^n)$. The tangent bundle is trivial, i. e., $TX \cong X \times \mathbf{E}$. In this case, a Riemannian metric on X is a sufficiently smooth mapping $g = (g_1, g_2)$ from X to $X \times \mathcal{L}^2_{\mathrm{sym}}(\mathbf{E})$ such that $g_1 = \mathrm{id}$ and $g_2(u)$ is positive definite for every $u \in X$. Choose $g = (\mathrm{id}, g_2)$ with g_2 as above. Then g is of class C^1 by the chain rule, and $g_2(u)$ is positive definite. □

Here $\lambda > 0$ can be chosen arbitrarily. Large values of λ increase the distance of g_2 to a singular form, whereas for small values of λ the metric is closer to the original problem. Both effects are desirable, so one has to find a balance between these goals when choosing λ.

We want to apply the steepest descent method on Riemannian manifolds. For finite dimensional manifolds, a discussion of this can be found for example in [38, Section 7.4]. We have to compute the gradient $\nabla_g \psi$ of the functional ψ defined in (5.3). By definition, the gradient at $u \in H^1(0,T;\mathbb{R}^n)$ satisfies

$$\begin{aligned}\psi'(u)h &= (F(u) \mid F'(u)h)_{L^2} = (F'(u)^* F(u) \mid h)_{H^1} \\ &= (\nabla_g \psi(u) \mid h)_g = \lambda\,(\nabla_g \psi(u) \mid h)_{H^1} + (F'(u)\nabla_g \psi(u) \mid F'(u)h)_{L^2}\end{aligned}$$

for every $h \in H^1(0,T;\mathbb{R}^n)$. Thus, we obtain the representation

$$\nabla_g \psi(u) = \left(\lambda + F'(u)^* F'(u)\right)^{-1} F'(u)^* F(u)$$

for $u \in H^1(0,T;\mathbb{R}^n)$. If F is of class C^2, there exists a (local) solution to the steepest descent equation (2.2) for any initial value $u_0 \in H^1(0,T;\mathbb{R}^n)$.

Note that if the problem is linear, i. e., if there exist matrices M_1 and M_2 and a function b such that $F(u)(t) = M_1 u'(t) + M_2 u(t) - b(t)$, then the Riemannian metric in lemma 5.3 equals the inner product corresponding to the graph norm of the operator $Au = M_1 u' + M_2 u$. Thus our approach indeed generalizes the discussion of section 3 to the non-linear case.

We mention that these considerations lead to numerical computations similar to the *Levenberg-Marquardt* algorithm. This algorithm adapts to local properties of the functional by varying λ. Of course we could resemble this in our setting by letting λ smoothly depend on $u \in H^1(0,T;\mathbb{R}^n)$, thus introducing a slightly more complicated Riemannian metric on the space. If we let λ tend to zero, we arrive at the *Gauss-Newton method* for solving non-linear least squares problems. For a detailed treatment these methods see for example [33, Section 10.3].

In the literature about Sobolev gradient methods, one notices that a lot of properties of linear problems carry over to the non-linear ones under some regularity conditions. But it seems to be an open question whether there exists a non-linear analogue to the fact that the Sobolev descent converges to the *nearest* solution of the equation, if one exists. It is natural to assume that this question is closely related to the theory of Riemannian metrics. More precisely, it is quite possible that up to reparametrization the trajectories of the steepest descent are geodesics of a suitable Riemannian metric. If this is the case, then this fact should be regarded as the appropriate generalization of the linear result. Those questions are beyond the scope of this article, but we propose this investigation as a rewarding topic of research.

6. Numerics

First we deal with general linear non-autonomous DAE. We explain our discretization and how we calculate a Sobolev gradient. In the abstract setting different norms lead to different gradients. We show how this can be transferred to the finite dimensional numerical setting taking the *graph norm* introduced in corollary 3.2 as an example. We introduce several different gradients with varying numerical properties. After that we discuss the overall steepest descent algorithm and the step size calculation. Then we move on to the fully non-linear case as in

section 5 and show how the numerics of the linear case can be generalized. Finally, we show how supplementary conditions can be integrated into Sobolev steepest descent.

6.1. **Discrete Formulation of Linear DAE.** First, we treat equation (2.1) where the matrices M_1 and M_2 may depend on $t \in [0,T]$. For all discretizations we employ the finite differences scheme. We fix an equidistant partition of $[0,T]$ into N subintervals of length $\delta := \frac{T}{N}$. We define a finite dimensional version of a vector valued function w as the vector $\tilde{w}$ containing the values $w(0), w(\delta), \dots, w(T)$. Hence a numerical solution is represented by $\tilde{u} \in \mathbb{R}^{(N+1)n}$ with structure

$$\tilde{u} = \big(\tilde{u}_k\big)_{k=0}^{N}, \quad \tilde{u}_k \approx u(\delta k) \in \mathbb{R}^n \text{ for } k = 0, \dots, N.$$

Define the block diagonal matrices $A, B \in \mathbb{R}^{(N+1)m \times (N+1)n}$ with blocks $M_1(0)$, $M_1(\delta)$, …, $M_1(T)$ and $M_2(0)$, $M_2(\delta)$, …, $M_2(T)$, respectively. An approximation of the functional ψ is given by

$$\tilde{\psi}\colon \mathbb{R}^{(N+1)n} \to \mathbb{R}_+,\ \tilde{u} \mapsto \tfrac{T}{2(N+1)} \big\| Q\tilde{u} - \tilde{b} \big\|^2_{\mathbb{R}^{(N+1)m}}, \tag{6.1}$$

where the matrix Q is defined as

$$Q = AD_1 + B, \quad Q \in \mathbb{R}^{(N+1)m \times (N+1)n} \tag{6.2}$$

for a matrix $D_1 \in \mathbb{R}^{(N+1)n \times (N+1)n}$ that numerically differentiates each component of a discretized function. The matrix Q is a discrete version of the differential operator of the DAE. Note that we replaced the L^2 function space norm by the corresponding finite dimensional Euclidean norm.

There are many possible choices for the matrix D_1. We use central differences involving both neighbor grid points in the interior and forward and backward differences at the boundary, all of them $\mathcal{O}(\delta^2)$ approximations. For $n = 1$ the differentiation matrix is

$$D_1^{(1)} = \frac{1}{2\delta} \begin{pmatrix} 1 & -4 & 3 & & \\ -1 & 0 & 1 & & \\ & \ddots & & \ddots & \\ & & -1 & 0 & 1 \\ & & -3 & 4 & -1 \end{pmatrix} \in \mathbb{R}^{(N+1)\times(N+1)}. \tag{6.3}$$

In general it is

$$D_1 = D_1^{(n)} = D_1^{(1)} \otimes I_n,$$

where $\otimes$ denotes the *Kronecker matrix product* (see e. g. [20, p. 220]) and I_n the $n \times n$ identity matrix.

6.2. **Different Gradients in Finite Dimensional Spaces.** We regard the derivative $\tilde{\psi}'(\tilde{u})$ as a linear functional acting on $\mathbb{R}^{(N+1)n}$. Then the ordinary Euclidean gradient of $\tilde{\psi}$ at $\tilde{u}$ can be calculated in terms of the matrix Q as follows.

$$\tilde{\psi}'(\tilde{u})h = \tfrac{T}{N+1} \Big(Q\tilde{u} - \tilde{b} \mid Qh \Big)_{\mathbb{R}^{(N+1)m}} = \Big(\tfrac{T}{N+1} \big(Q^T Q\tilde{u} - Q^T\tilde{b} \big) \mid h \Big)_{\mathbb{R}^{(N+1)n}}$$

This equality holds for all $h \in \mathbb{R}^{(N+1)n}$, thus

$$\nabla_{\mathrm{e}}\tilde{\psi}(\tilde{u}) := \tfrac{T}{N+1} \big(Q^T Q\tilde{u} - Q^T\tilde{b} \big) \tag{6.4}$$

is the *Euclidean gradient*.

Now we explain how to compute different Sobolev gradients. To this end, note that the above Euclidean gradient does not correspond in any way to the gradient of ψ in the abstract setting. In fact, Q^T is the adjoint of Q with respect to the Euclidean inner product whereas in (2.3) the adjoint is taken with respect to the norm in H^1. Thus, we have to discretize the H^1 inner product and use it to calculate the corresponding finite dimensional adjoint.

Any inner product can be related to the ordinary Euclidean inner product via a positive definite matrix. For $H^1(0,T;\mathbb{R}^n)$ we choose

$$S_{\mathrm{H}} := I_{(N+1)n} + D_1^T D_1. \tag{6.5}$$

By definition, the Sobolev gradient $\nabla_{\mathrm{H}}\tilde{\psi}(\tilde{u})$ at the point $\tilde{u}$ has to satisfy

$$\tilde{\psi}'(\tilde{u})h = \Big(\nabla_{\mathrm{H}}\tilde{\psi}(\tilde{u}) \mid h\Big)_H = \Big(S_{\mathrm{H}}\nabla_{\mathrm{H}}\tilde{\psi}(\tilde{u}) \mid h\Big)_{\mathbb{R}^{(N+1)n}} = \Big(\nabla_{\mathrm{e}}\tilde{\psi}(\tilde{u}) \mid h\Big)_{\mathbb{R}^{(N+1)n}}$$

for all $h \in \mathbb{R}^{(N+1)n}$. Therefore, to calculate the gradient ∇_{H} numerically it suffices to solve the linear system

$$S_{\mathrm{H}}x = \nabla_{\mathrm{e}}\tilde{\psi}(\tilde{u}) \tag{6.6}$$

for the unknown $x \in \mathbb{R}^{(N+1)n}$.

Using the Sobolev gradient ∇_{H} instead of ∇_{e} already results in significantly better numerical performance. Nevertheless, further improvements can be achieved using appropriately weighted Sobolev gradients. For a detailed treatment of steepest descent in weighted Sobolev spaces in the context of ODE and PDE with singularities, we refer to [24].

Section 3 already indicated the graph norm as a promising candidate for a norm that is tailored to the structure of the DAE. Hence we consider inner products in finite dimensions that are related to the graph norm. Natural candidates are associated with the positive definite matrices

$$\begin{aligned} S_{\mathrm{W}_1,\lambda} &:= \lambda I_{(N+1)n} + A^T D_1^T D_1 A, \\ S_{\mathrm{W}_2,\lambda} &:= \lambda I_{(N+1)n} + A^T D_1^T D_1 A + B^T B, \\ S_{\mathrm{G},\lambda} &:= \lambda I_{(N+1)n} + Q^T Q, \end{aligned} \tag{6.7}$$

for $\lambda > 0$. The identity matrix guarantees positive definiteness, while the respective other part determines the relation to the DAE. By choosing λ smaller, the graph part gains more weight. Note that $S_{\mathrm{G},1}$ is a straightforward discretization of the graph norm. We can calculate the corresponding Sobolev gradients $\nabla_{\mathrm{W}_1,\lambda}$, $\nabla_{\mathrm{W}_2,\lambda}$ and $\nabla_{\mathrm{G},\lambda}$ as before by solving linear systems similar to equation (6.6).

We mention that the matrices in (6.7) are still sparse but structurally more complicated than the matrix S_{H} defined in (6.5) which corresponds to the H^1 inner product. The matrix S_{H} is block-diagonal, which allows us to solve the linear system individually within each block. All the n blocks equal $I_{N+1} + (D_1^{(1)})^T D_1^{(1)}$ which is a band matrix depending only on the choice of numerical differentiation. As it usually is tridiagonal or pentadiagonal, efficient solvers are available for the corresponding linear systems.

6.3. **Discrete Steepest Descent Algorithm and Step Size Calculation.** We want to discretize the continuous steepest descent (2.2). Once we have decided which gradient ∇ to use, we follow the usual scheme of steepest descent algorithms and the more general *line search methods* [33, Chapter 3]. First we fix an initial estimate $\tilde{u}_0$. Then we know that $-\nabla\tilde{\psi}(\tilde{u}_0)$ is a descent direction of $\tilde{\psi}$ at $\tilde{u}_0$,

i. e., $\tilde{\psi}(\tilde{u}_0)$ locally decreases along the direction of the negative gradient. More precisely, the negative gradient specifies the direction in which the directional derivative (Gâteaux derivative, cf. [2]) becomes minimal among all directions of unit length which is where the choice of the norm comes in.

For a discretization of the continuous steepest descent (2.2), we have to make steps which are small enough such that $\tilde{\psi}$ still decreases, and large enough such that it decreases significantly. A straight-forward choice for the *step size* s^* is the least non-negative real number that minimizes $\tilde{\psi}(\tilde{u} - s^*\nabla)$, assuming that such a number exists. Here we abbreviated $\nabla\tilde{\psi}(\tilde{u})$ by ∇. Of course, if $\nabla \neq 0$ there exists a positive s^* such that $\tilde{\psi}(\tilde{u} - s^*\nabla) < \tilde{\psi}(\tilde{u})$. Since this is the only occurrence of the gradient in the algorithm, the scaling of the gradient can be compensated by the choice of s^*. Thus the results remain the same if we drop the factor $\frac{T}{N+1}$ in formula (6.4) for our calculations.

In the linear case it is easy to calculate the optimal s^* by interpolation, as along a line the functional is a quadratic polynomial. But in the non-linear case this is a more difficult problem. In practice, it usually is sufficient to calculate a local minimizer instead of the global minimizer s^*. Nocedal and Wright give a description of sophisticated step-length selection algorithms [33, Section 3.5]. Those algorithms try to use function values and gradient information as efficiently as possible and produce step sizes satisfying certain descent conditions. In our implementation we assume local convexity and search along an exponentially increasing sequence for the first increase of $\tilde{\psi}$ on the line. We then perform a ternary search with this upper bound yielding a local minimizer of $\tilde{\psi}$.

We experienced that usually it is advantageous to damp the step size s^*, i. e., to multiply s^* by a factor $\mu \in (0, 1)$, when using the gradient itself for the direction. Alternatively, our implementation provides the possibility to incorporate previous step directions and step sizes into the calculation of the new ones. This pattern is employed in *non-linear conjugate gradient methods*, and it can be used with Sobolev gradients as well; see for example the Polak-Ribière or Fletcher-Reeves formulae [33, Section 5.2].

Algorithm 1 is a summary of our final discrete steepest descent procedure. This is a straight-forward application of the general discrete steepest descent method for a given cost functional. Sufficient conditions for convergence to a minimizer involving convexity and gradient inequalities can be found for example in [33, Chapter 3].

Algorithm 1. Discrete steepest descent

```
Generate some initial guess ũ_0.                          | e. g. a constant function
i ← 0
while ũ_i does not have target precision do
  ∇_e ← Euclidean gradient of ψ̃ at ũ_i                    | see equation (6.4)
  Build linear system incorporating supp. cond. at ũ_i.   | sections 6.2 and 6.6
  ∇_S ← solution of linear system for right hand side ∇_e | see equation (6.5)
  s* ← choose good step size for ∇_S                      | section 6.3
  ũ_{i+1} ← ũ_i − μ s* ∇_S                                | damped update, 0 < μ ≤ 1
  i ← i + 1
end while
```

6.4. **Least Squares Method.** We now describe the close connection between the Sobolev gradient $\nabla_{G,\lambda}$ coming from $S_{G,\lambda}$ as in (6.7) and the well-known *least squares*

method. In the limit $\lambda \to 0$ the resulting linear system might be singular, but is still solvable for the given right hand side. In fact, for $\lambda \to 0$ the linear system corresponding to equation (6.6) becomes

$$Q^T Q x = Q^T (Q\tilde{u} - \tilde{b}).$$

Note that we have rescaled the Euclidean gradient by the factor $\frac{N+1}{T}$ as justified in section 6.3. Starting the discrete steepest descent at an initial guess $\tilde{u}_0$ we compute x and take a step of length $\delta \geq 0$ into the direction $-x$. The parameter δ is chosen such that $\psi(\tilde{u} - \delta x)$ is minimal. We claim that $\delta = 1$. In fact, for this δ we arrive at $\tilde{u} - x$ which satisfies the normal equations of the problem $Qy = \tilde{b}$, i. e.,

$$Q^T Q(\tilde{u} - x) = Q^T Q\tilde{u} - \left(Q^T Q\tilde{u} - Q^T\tilde{b}\right) = Q^T\tilde{b}.$$

This shows that $\tilde{u} - x$ globally minimizes the functional, thus proving $\delta = 1$. Moreover, this shows that in the limit descent with $\nabla_{G,\lambda}$ converges to the solution of the least squares problem in the first step. Note, however, that positive definiteness is a very desirable property for a linear system and a direct solution of the normal equations may be numerically considerably more difficult.

This relation indicates a possible reason why also for the non-linear case the convergence of the steepest descent is observed to be fastest for $\nabla_{G,\lambda}$ with small λ, at least among the gradients we have used.

6.5. **The Non-Linear Case.** In the setting of equation (5.1), define $A(\tilde{u})$ and $B(\tilde{u})$ as block diagonal matrices with blocks $f_y(k\delta, \tilde{u}_k, D_1\tilde{u}_k)$ and $f_x(k\delta, \tilde{u}_k, D_1\tilde{u}_k)$ for $k = 0, \dots, N$, respectively. We use the function

$$\widetilde{F}\colon \mathbb{R}^{(N+1)n} \to \mathbb{R}^{(N+1)m},\ \tilde{u} \mapsto \left(f(k\delta, \tilde{u}_k, D_1\tilde{u}_k)\right)_k$$

as discretization of F defined by (5.2). Observe that $\widetilde{F}'(\tilde{u})h = A(\tilde{u})D_1 h + B(\tilde{u})h$, which resembles (5.4). Then $\tilde{\psi}(\tilde{u}) := \frac{1}{2}\left\|\widetilde{F}(\tilde{u})\right\|_2^2$ has derivative

$$\tilde{\psi}'(\tilde{u})h = \left(\widetilde{F}(\tilde{u}) \mid (A(\tilde{u})D_1 + B(\tilde{u}))\, h\right) = \left(Q(\tilde{u})^T \widetilde{F}(\tilde{u}) \mid h\right), \tag{6.8}$$

where we set $Q := AD_1 + B$ as in the notation of the linear case.

Now we can proceed as in the linear case. The only difference is that the matrices A and B depend on the current position $\tilde{u}$, and hence the positive definite matrices defined as in (6.7) change during the process as well. This corresponds to steepest descent under a variable inner product introduced in lemma 5.3. It is also connected to *quasi-Newton methods* which update an approximation of the Hessian at each step. For details on quasi-Newton methods see [33, Chapter 6].

Originally we came up with this method for non-linear DAE as a direct generalization of the linear case. Only for a formal justification we have equipped $H^1(0, T; \mathbb{R}^n)$ with a natural structure making it into a Riemannian manifold leading to the gradient we use. However, consequently following this second approach we would have been led to a algorithm which slightly differs from algorithm 1. As in general Riemannian manifolds do not carry a vector space structure, there are no "straight lines" the steepest descent could follow. One usually employs the exponential map of the manifold as a substitute, traveling along geodesics. Although there is no difference between these two variants for continuous steepest descent, i. e., in the limit of infinitesimally small step size, for the numerics one

has to choose. We decided in favor of the straight lines since computing the exponential map means solving an ordinary differential equation which is a much more complicated operation unnecessarily complicating the implementation.

6.6. **Supplementary Conditions.** To support *linear supplementary conditions*, we want the steepest descent steps to preserve specified features of the initial function. Therefore, we use Sobolev gradients that do not change these features. We remark that the methods of this chapter can be applied using any gradient. We have chosen the space $H^1(0,T;\mathbb{R}^n)$ with its usual norm only for clarity of exposition. More precisely, let $u_0 \in H^1(0,T;\mathbb{R}^n)$ be an initial estimate satisfying the supplementary conditions. Denote by H_{a} the closed linear subspace of $H^1(0,T;\mathbb{R}^n)$ such that $u_0 + H_{\mathrm{a}}$ is the space of all functions in $H^1(0,T;\mathbb{R}^n)$ satisfying the supplementary conditions. We call H_{a} the *space of admissible functions*.

Define the functional ψ_{a} as

$$\psi_{\mathrm{a}} \colon H_{\mathrm{a}} \to \mathbb{R}_+, \quad \psi_{\mathrm{a}}(u) := \psi(u_0+u) = \tfrac{1}{2}\,\|F(u_0+u)\|_{L^2}^2 \,.$$

We have to calculate the gradient of ψ_{a} with respect to the space H_{a} equipped with the inner product induced by $H^1(0,T;\mathbb{R}^n)$. As this gradient naturally lies in the space of admissible functions, steepest descent starting with u_0 will preserve the supplementary conditions while minimizing ψ_{a}.

Let P_{a} be the orthogonal projection of $H^1(0,T;\mathbb{R}^n)$ onto H_{a}. Now $\psi_{\mathrm{a}}'(u)h = \psi'(u_0+u)h$ for $h \in H_{\mathrm{a}}$, and consequently

$$\begin{aligned} \psi_{\mathrm{a}}'(u)P_{\mathrm{a}}h &= \psi'(u_0+u)P_{\mathrm{a}}h = \big((\nabla\psi)(u_0+u) \mid P_{\mathrm{a}}h\big)_{H^1} \\ &= \big(P_{\mathrm{a}}(\nabla\psi)(u_0+u) \mid h\big)_{H^1} \end{aligned} \tag{6.9}$$

for all $h \in H^1(0,T;\mathbb{R}^n)$. It follows that $(\nabla\psi_{\mathrm{a}})(u) = P_{\mathrm{a}}(\nabla\psi)(u_0+u)$.

Now we transfer this to the finite dimensional setting in a numerically tractable way. Let $C \in \mathbb{R}^{k\times(N+1)n}$ be a matrix such that $\widetilde{H}_{\mathrm{a}} := \ker C$ is a finite dimensional version of H_{a}. The set of functions satisfying the supplementary conditions introduced by the matrix C is given by $\tilde{u}_0 + \widetilde{H}_{\mathrm{a}}$ for any valid function $\tilde{u}_0$. We understand $\tilde{\psi}_{\mathrm{a}}(\tilde{u})$ as a functional on $\widetilde{H}_{\mathrm{a}}$ analogously to the above definition of ψ_{a}.

Denote by $\widetilde{P}_{\mathrm{a}}$ the orthogonal projection in $\mathbb{R}^{(N+1)n}$ onto $\ker C$ with respect to the Euclidean inner product. We search for $\nabla_S \in \widetilde{H}_{\mathrm{a}}$ satisfying $\tilde{\psi}_{\mathrm{a}}(\tilde{u}) = (\nabla_S \mid h)$ for all $h \in \widetilde{H}_{\mathrm{a}}$. Similarly to (6.9), we calculate for any $h \in \mathbb{R}^{(N+1)n}$

$$\begin{aligned} \tilde{\psi}_{\mathrm{a}}'(\tilde{u})\widetilde{P}_{\mathrm{a}}h &= \tilde{\psi}'(\tilde{u}_0+\tilde{u})\widetilde{P}_{\mathrm{a}}h = \Big(\big(\nabla_{\mathrm{e}}\tilde{\psi}\big)(\tilde{u}_0+\tilde{u}) \mid \widetilde{P}_{\mathrm{a}}h\Big)_{\mathrm{e}} \\ &= \Big(\widetilde{P}_{\mathrm{a}}\big(\nabla_{\mathrm{e}}\tilde{\psi}\big)(\tilde{u}_0+\tilde{u}) \mid h\Big)_{\mathrm{e}} \overset{!}{=} \Big(\nabla_S \mid \widetilde{P}_{\mathrm{a}}h\Big)_S = \Big(\widetilde{P}_{\mathrm{a}}S\widetilde{P}_{\mathrm{a}}\nabla_S \mid h\Big)_{\mathrm{e}} \,. \end{aligned}$$

Defining $S_{\mathrm{a}} := \widetilde{P}_{\mathrm{a}}S\widetilde{P}_{\mathrm{a}}$, it is obvious that S_{a} is positive definite if restricted to $\widetilde{H}_{\mathrm{a}}$ since S is positive definite. To calculate the discrete Sobolev gradient we have to solve the linear system

$$S_{\mathrm{a}}x = \widetilde{P}_{\mathrm{a}}\,(\nabla_{\mathrm{e}}\psi)(\tilde{u}_0+\tilde{u})$$

for x in $\widetilde{H}_{\mathrm{a}}$. Note that one could use the conjugate gradient method for solving this system, as the right hand side is in $\widetilde{H}_{\mathrm{a}}$, cf. [13, Algorithm 13.2] and [33, Algorithm 5.2].

This approach allows us to impose very general linear supplementary conditions, like boundary conditions or periodic boundary conditions for the function as well

as for its derivative, or other not necessarily local linear conditions at arbitrary grid points. Only fixing values is computationally unproblematic, as this corresponds to deleting appropriate columns and rows in S and the calculated gradient. But more general supplementary conditions result in a non-sparse orthogonal projection matrix $\widetilde{P}_{\mathrm{a}}$ and a dense inner product matrix S_{a}. This renders sparse solvers useless. Then it might help to regard the calculation of the gradient under supplementary conditions as an equality constrained linear least squares optimization problem. For details, we refer to the book of Golub and van Loen [16, Section 12.1].

7. Implementation Details and Examples with Tables and Plots

We present numerical results for some of the problems we used in the development of our algorithm to illustrate its strengths and weaknesses. Altogether we utilized various sample problems from different sources to test the correctness and to study the performance in representative cases. The reader is welcome to check out the source code containing many example problems, which is freely available online [32].

In subsection 7.1 we discuss the results for an interesting example investigated by Mahavier who studied ODE problems with singularities in the context of Sobolev gradients [26, 24]. Another interesting problem, posing difficulties to several solvers, is discussed in subsection 7.2 which we found in [36]. The results for a few more intricate example problems from the IVP Testset [28] are discussed in subsection 7.3. One possible application utilizing the feature that no initial conditions have to be specified is explained in section 7.4. In this context, an example exposing an at first sight surprising behavior is described in 7.4.3. For testing purposes, several boundary value problems have been treated, among them examples from Ascher and Spiteri [3]. Other employed test problems stem from the books of Hairer and Wanner [18], Kunkel and Mehrmann [20] and Carey, Richardson, Reed and Mulvaney [13] and from the IVP and BVP website of Cash [11].

When we speak of the *index of a DAE* in an example we always refer to the *differentiation index* which agrees with the *index of nilpotency* for linear DAE with constant coefficients [18, Section VII.1]. There are several other index concepts for DAE, each stressing different aspects of the equation [36, Section 2.4]. By the *maximal absolute error* and the *average absolute error* of a numerical approximation $\tilde{u}$ we mean

$$E_{\mathrm{abs}}(\tilde{u}) := \max_{i=0,\dots,N} \|u(t_i) - \tilde{u}(t_i)\|_\infty \text{ and } E_{\mathrm{avg}}(\tilde{u}) := \tfrac{T}{N+1} \sum_{i=0}^{N} \|u(t_i) - \tilde{u}(t_i)\|_2^2,$$

respectively, where u is a (highly) exact solution. The value $\tilde{\psi}(\tilde{u})$ is called the *residual at* $\tilde{u}$.

Please keep in mind that our implementation aims at generality and could be considerably optimized for more specialized cases. Thus it would not make sense to give timings of program runs in the following. We mention that the computer architecture as well as compiler flags have impact on the numerical values. There are parameters affecting the rate of convergence and the quality of the numerical results which we do not describe here in detail. Those parameters include precision bounds for termination checks of the linear solvers and control details of the line search in the gradient direction. However, all parameters which *severely* affect the results are documented.

7.1. **Non-Linear ODE with Irregular Singularity.** ODE with meromorphic coefficients can be formulated as a singular ODE which is a special case of a DAE. More precisely, one can remove an k^{th} order pole at t_0 by multiplying the corresponding equation with $(t-t_0)^k$, thereby getting a coefficient function in front of the highest derivative $y^{(n)}$ with a root at t_0. However, these examples are very special and relatively simple examples of DAE and hence are often not regarded as actual DAE. The following example is of this type.

We consider the non-linear ordinary differential equation

$$\begin{cases} t^2y'(t) = 2ty(t) + y(t)^2 \text{ for } t \in [0,1] \\ y(1) = 1 \end{cases} \quad \text{with solution } y(t) = \frac{t^2}{2-t},$$

which is discussed in [26, Section 4]. Note that Mahavier introduces and employs *weighted Sobolev descent* for such problems. He calculates the gradient with respect to a discretized weighted Sobolev space tailored to the problem. For semi-linear ODE problems our gradient ∇_{W_1} of section 6.2 corresponds directly to the weighted Sobolev gradients there. We solve the above problem on the interval $[0,1]$ starting with the initial function $u_0(t) = t$. In tables 1 and 2 we did not damp the steepest descent to allow for comparison with Mahavier's results [26]. However, there remain minor differences due to discretization, scaling and the employed line-search method.

Table 1 lists the convergence behavior for several gradients. The Euclidean gradient shows poor performance whereas the usual Sobolev gradient already improves the situation significantly. Best performance is achieved by the weighted Sobolev gradient and the gradient corresponding to the graph norm, the latter being slightly ahead. Similar observations can be made about the average errors listed in table 2. Damping with 0.85 yields considerably better results. For example in the graph norm case with $N = 1000$ and $\lambda = 1$ this damping yields to convergence in less than 1000 steps to a residual below $3 \cdot 10^{-15}$, an average error of about $2 \cdot 10^{-10}$ and a maximal error of about $4 \cdot 10^{-4}$. This is significantly better than what is achieved in the same setting without damping after 10000 steps. In that case the residual is only reduced to about $1 \cdot 10^{-14}$ despite of the higher number of steps. The reason for this improvement lies in the usual convergence behavior of steepest descent methods. Often they exhibit the tendency to zig-zag [33, see Section 3.3] which is somewhat mitigated by this simple damping. For this reason we always used a damping factor of $\mu = 0.85$ in the numerics unless otherwise stated.

To compare the behavior of the gradients for increasing grid size we show results for a finer equidistant grid of size $N = 10000$ in table 3. Note that the residual of the exact solution is approximately $6.79 \cdot 10^{-17}$ for this grid. The Euclidean gradient completely fails to converge and exhibits horrible numerical performance. The ordinary Sobolev gradient copes fairly with the finer grid. Among the gradients with $\lambda = 1$, again ∇_{W_1} and ∇_G achieve the best results.

However, for all gradients under consideration an appropriate choice of λ improves the numerical performance. For example using $\nabla_{W_1,0.05}$ in the above setting we get a residual of $2 \cdot 10^{-17}$ after 1000 steps, with $E_{\text{avg}} \approx 10^{-11}$ and $E_{\text{abs}} \approx 2 \cdot 10^{-4}$. But in the case of steepest descent with ∇_G, the impact of a smaller value is extraordinary. The rightmost two columns of table 3 show the results for $\lambda = 10^{-3}$ and $\lambda = 10^{-5}$. The norm of the gradient dropped below 10^{-12} in 110 and 42 steps, respectively. With even smaller values of λ even better results are achieved.

TABLE 1. Residuals for grid size $N = 100$ without damping ($\mu = 1$).

	Residual (starting with $4.06 \cdot 10^{-1}$ for $\tilde{u}_0$)			
Steps	**Euclidean ∇_{e}**	**Sobolev ∇_{H}**	**Weighted ∇_{W_1}**	**Graph ∇_{G}**
5	$3.7 \cdot 10^{-1}$	$6.7 \cdot 10^{-4}$	$2.2 \cdot 10^{-05}$	$6.4 \cdot 10^{-06}$
10	$3.6 \cdot 10^{-1}$	$3.8 \cdot 10^{-4}$	$3.5 \cdot 10^{-06}$	$7.2 \cdot 10^{-07}$
20	$3.3 \cdot 10^{-1}$	$2.0 \cdot 10^{-4}$	$5.9 \cdot 10^{-07}$	$9.9 \cdot 10^{-08}$
50	$2.8 \cdot 10^{-1}$	$7.9 \cdot 10^{-5}$	$5.6 \cdot 10^{-08}$	$8.4 \cdot 10^{-09}$
100	$2.2 \cdot 10^{-1}$	$3.7 \cdot 10^{-5}$	$9.7 \cdot 10^{-09}$	$1.4 \cdot 10^{-09}$
200	$1.5 \cdot 10^{-1}$	$1.6 \cdot 10^{-5}$	$1.7 \cdot 10^{-09}$	$2.5 \cdot 10^{-10}$
500	$7.1 \cdot 10^{-2}$	$5.5 \cdot 10^{-6}$	$1.8 \cdot 10^{-10}$	$3.7 \cdot 10^{-11}$
1000	$2.9 \cdot 10^{-2}$	$2.3 \cdot 10^{-6}$	$4.3 \cdot 10^{-11}$	$1.8 \cdot 10^{-11}$
2000	$9.8 \cdot 10^{-3}$	$9.9 \cdot 10^{-7}$	$2.0 \cdot 10^{-11}$	$1.5 \cdot 10^{-11}$
5000	$1.7 \cdot 10^{-3}$	$3.1 \cdot 10^{-7}$	$1.5 \cdot 10^{-11}$	$1.4 \cdot 10^{-11}$
10000	$3.9 \cdot 10^{-4}$	$1.2 \cdot 10^{-7}$	$1.4 \cdot 10^{-11}$	$1.4 \cdot 10^{-11}$

TABLE 2. Average errors for grid size $N = 100$ without damping ($\mu = 1$).

	Average Error (starting with $6.17 \cdot 10^{-2}$ for $\tilde{u}_0$)			
Steps	**Euclidean ∇_{e}**	**Sobolev ∇_{H}**	**Weighted ∇_{W_1}**	**Graph ∇_{G}**
5	$6.0 \cdot 10^{-2}$	$7.8 \cdot 10^{-3}$	$2.3 \cdot 10^{-4}$	$1.0 \cdot 10^{-4}$
10	$5.9 \cdot 10^{-2}$	$5.7 \cdot 10^{-3}$	$8.1 \cdot 10^{-5}$	$3.1 \cdot 10^{-5}$
20	$5.8 \cdot 10^{-2}$	$4.0 \cdot 10^{-3}$	$2.8 \cdot 10^{-5}$	$9.6 \cdot 10^{-6}$
50	$5.4 \cdot 10^{-2}$	$2.3 \cdot 10^{-3}$	$7.0 \cdot 10^{-6}$	$2.2 \cdot 10^{-6}$
100	$4.8 \cdot 10^{-2}$	$1.5 \cdot 10^{-3}$	$2.4 \cdot 10^{-6}$	$7.6 \cdot 10^{-7}$
200	$4.0 \cdot 10^{-2}$	$9.4 \cdot 10^{-4}$	$8.5 \cdot 10^{-7}$	$2.6 \cdot 10^{-7}$
500	$2.6 \cdot 10^{-2}$	$4.9 \cdot 10^{-4}$	$2.1 \cdot 10^{-7}$	$6.6 \cdot 10^{-8}$
1000	$1.6 \cdot 10^{-2}$	$3.0 \cdot 10^{-4}$	$7.6 \cdot 10^{-8}$	$2.5 \cdot 10^{-8}$
2000	$8.6 \cdot 10^{-3}$	$1.8 \cdot 10^{-4}$	$2.9 \cdot 10^{-8}$	$1.2 \cdot 10^{-8}$
5000	$3.1 \cdot 10^{-3}$	$9.0 \cdot 10^{-5}$	$1.1 \cdot 10^{-8}$	$9.0 \cdot 10^{-9}$
10000	$1.3 \cdot 10^{-3}$	$5.3 \cdot 10^{-5}$	$9.3 \cdot 10^{-9}$	$8.0 \cdot 10^{-9}$

For $\lambda = 10^{-19}$ and $\mu = 1$ we obtain a residual of about 10^{-23} in 6 steps, with $E_{\mathrm{avg}} \approx 10^{-15}$ and $E_{\mathrm{abs}} \approx 4 \cdot 10^{-6}$. However, in general choosing λ that small leads to failure of the linear solver and huge numerical errors.

In the setting of this singular ODE, Sobolev descent with respect to the graph norm gives results which are superior to the steepest descent method relying on weighted Sobolev spaces as in [26]. Additionally, choosing a small λ vastly improves the rate of convergence in many cases, but is numerically more demanding. However, steepest descent with respect to the graph norm is computationally more expensive than weighted Sobolev steepest descent. This is because S_{G} has to be constructed in each iteration because it depends on $\tilde{u}$, whereas S_{W_1} remains the same during the process for this example.

7.2. **A Small Non-Trivial Linear DAE.** Consider for $\eta \in \mathbb{R}$ the non-autonomous linear index 2 DAE

$$\begin{pmatrix} 1 & 0 \\ 1 & \eta t \end{pmatrix} u'(t) + \begin{pmatrix} 1 & \eta t \\ 0 & 1+\eta \end{pmatrix} = \begin{pmatrix} \exp(-t) \\ 0 \end{pmatrix}. \tag{7.1}$$

TABLE 3. Residuals for grid size $N = 10000$ with damping factor $\mu = 0.85$.

	Residual (starting with $4.0 \cdot 10^{-1}$ for $\tilde{u}_0$)					
Steps	∇_{e}	∇_{H}	∇_{W_1}	$\nabla_{\mathrm{G},1}$	$\nabla_{\mathrm{G},10^{-3}}$	$\nabla_{\mathrm{G},10^{-5}}$
5	$4.0 \cdot 10^{-1}$	$8.3 \cdot 10^{-4}$	$2.6 \cdot 10^{-05}$	$5.2 \cdot 10^{-06}$	$2.4 \cdot 10^{-09}$	$2.4 \cdot 10^{-09}$
10	$4.0 \cdot 10^{-1}$	$2.6 \cdot 10^{-4}$	$3.9 \cdot 10^{-06}$	$5.8 \cdot 10^{-07}$	$7.7 \cdot 10^{-14}$	$1.5 \cdot 10^{-17}$
20	$4.0 \cdot 10^{-1}$	$1.2 \cdot 10^{-4}$	$4.2 \cdot 10^{-07}$	$9.3 \cdot 10^{-09}$	$9.8 \cdot 10^{-16}$	$1.2 \cdot 10^{-20}$
30	$4.0 \cdot 10^{-1}$	$5.7 \cdot 10^{-5}$	$1.4 \cdot 10^{-08}$	$2.5 \cdot 10^{-09}$	$2.5 \cdot 10^{-16}$	$4.9 \cdot 10^{-21}$
40	$4.0 \cdot 10^{-1}$	$4.2 \cdot 10^{-5}$	$4.6 \cdot 10^{-09}$	$1.2 \cdot 10^{-09}$	$8.8 \cdot 10^{-17}$	$1.3 \cdot 10^{-21}$
50	$4.0 \cdot 10^{-1}$	$1.5 \cdot 10^{-5}$	$3.0 \cdot 10^{-09}$	$6.3 \cdot 10^{-10}$	$3.5 \cdot 10^{-17}$	
100	$3.9 \cdot 10^{-1}$	$9.1 \cdot 10^{-6}$	$2.3 \cdot 10^{-10}$	$3.1 \cdot 10^{-11}$	$4.3 \cdot 10^{-19}$	
150	$3.9 \cdot 10^{-1}$	$3.3 \cdot 10^{-6}$	$3.4 \cdot 10^{-11}$	$2.9 \cdot 10^{-12}$		
200	$3.9 \cdot 10^{-1}$	$1.2 \cdot 10^{-6}$	$1.0 \cdot 10^{-11}$	$2.0 \cdot 10^{-12}$		
400	$3.9 \cdot 10^{-1}$	$2.8 \cdot 10^{-7}$	$3.3 \cdot 10^{-13}$	$4.7 \cdot 10^{-14}$		
1000	$3.5 \cdot 10^{-1}$	$6.9 \cdot 10^{-8}$	$3.2 \cdot 10^{-14}$	$2.6 \cdot 10^{-15}$		
E_{avg}	$5.9 \cdot 10^{-2}$	$3.3 \cdot 10^{-5}$	$1.3 \cdot 10^{-09}$	$3.0 \cdot 10^{-10}$	$1.4 \cdot 10^{-12}$	$4.5 \cdot 10^{-14}$
E_{abs}	$3.3 \cdot 10^{-1}$	$4.3 \cdot 10^{-2}$	$8.9 \cdot 10^{-04}$	$5.6 \cdot 10^{-04}$	$8.5 \cdot 10^{-05}$	$1.9 \cdot 10^{-05}$

This equation has been introduced by Petzold, Gear and Hsu in [34]. In the range $\eta < -0.5$ it is known to pose difficulties to several numerical methods for DAE, among them the implicit Euler method, BDF, and RadauIIA. More information about this equation along with some numerical tests can be found in [36, Section 4.2]. It has a unique solution given by

$$u(t) = \begin{pmatrix} (1 - \eta t)\exp(-t) \\ \exp(-t) \end{pmatrix}.$$

Thus, the only consistent initial value is $u(0)$. In our test we always set $\eta = -0.8$, choose the initial function to equal 2 in both components, and solve the equation on the interval $[0, 3]$ for the grid size $N = 1000$. Apart from the least squares case where we do not use damping, we always set $\mu = 0.85$. The residual of the initial function is 2.99, with $E_{\mathrm{avg}} = 15.37$ and $E_{\mathrm{abs}} = 1.95$. In table 4 we show results of the steepest descent method applied to problem (7.1) using the gradient $\nabla_{\mathrm{G},\lambda}$ for different values of the parameter λ. To facilitate comparison we also list results of ordinary Sobolev descent employing ∇_{H} and of the least squares method. The latter can be applied since the problem is linear. Steps which reached optimal residuals with respect to solver and floating point precision are marked with *. We omit the results for the Euclidean gradient which decreases the residual only to about $6.09 \cdot 10^{-1}$ in the first 10000 steps. In figures 1 and 2 we illustrate the development of the steepest descent for these gradients. The individual plots depict the results' first components after several steps, where darker color corresponds to more iterations.

Again, descent with $\nabla_{\mathrm{G},1}$ reduces the residual faster than descent with the ordinary Sobolev gradient. Decreasing λ results in even better convergence. However, note that ordinary Sobolev descent attains superior error values. This can be understood by looking at the plots in figure 1. The solutions found by ordinary Sobolev descent approach the solution slowly, but in a very regular and smooth way, whereas descent according to the graph norm approaches the solution rapidly in the interior of the interval $[0, 3]$ and only afterwards and more slowly at the interval boundaries. Regarding the error values, λ has to be decreased to 10^{-3} for

TABLE 4. Results for problem (7.1).

Gradient	**Steps**	**Residual**	**Avg. Error**	**Max. Error**
∇_{H}	100	$2.6 \cdot 10^{-04}$	$5.6 \cdot 10^{-02}$	$3.2 \cdot 10^{-1}$
	1000	$1.0 \cdot 10^{-06}$	$1.1 \cdot 10^{-03}$	$1.1 \cdot 10^{-1}$
	10000	$7.9 \cdot 10^{-09}$	$5.1 \cdot 10^{-05}$	$4.5 \cdot 10^{-2}$
$\nabla_{\mathrm{G},1}$	100	$2.8 \cdot 10^{-05}$	$2.1 \cdot 10^{-01}$	$2.1 \cdot 10^{+0}$
	1000	$7.5 \cdot 10^{-08}$	$2.8 \cdot 10^{-02}$	$1.8 \cdot 10^{+0}$
	10000	$3.8 \cdot 10^{-10}$	$3.1 \cdot 10^{-03}$	$8.7 \cdot 10^{-1}$
$\nabla_{\mathrm{G},10^{-3}}$	10	$5.2 \cdot 10^{-07}$	$5.4 \cdot 10^{-02}$	$2.0 \cdot 10^{+0}$
	100	$5.9 \cdot 10^{-10}$	$3.8 \cdot 10^{-03}$	$9.6 \cdot 10^{-1}$
	1000	$6.9 \cdot 10^{-13}$	$3.2 \cdot 10^{-04}$	$1.3 \cdot 10^{-1}$
	10000	$5.0 \cdot 10^{-15}$	$6.5 \cdot 10^{-05}$	$2.7 \cdot 10^{-2}$
$\nabla_{\mathrm{G},10^{-5}}$	10	$6.2 \cdot 10^{-10}$	$3.8 \cdot 10^{-03}$	$9.6 \cdot 10^{-1}$
	100	$4.0 \cdot 10^{-13}$	$2.7 \cdot 10^{-04}$	$1.1 \cdot 10^{-1}$
	1000	$6.3 \cdot 10^{-16}$	$3.3 \cdot 10^{-05}$	$1.4 \cdot 10^{-2}$
	10000	$1.4 \cdot 10^{-17}$	$8.2 \cdot 10^{-06}$	$4.0 \cdot 10^{-3}$
$\nabla_{\mathrm{G},10^{-10}}$	5	$1.7 \cdot 10^{-08}$	$2.6 \cdot 10^{-05}$	$1.0 \cdot 10^{-2}$
	10	$1.3 \cdot 10^{-16}$	$1.1 \cdot 10^{-05}$	$5.6 \cdot 10^{-3}$
	20	$5.2 \cdot 10^{-18}$	$3.4 \cdot 10^{-06}$	$2.5 \cdot 10^{-3}$
	60	$2.8 \cdot 10^{-23}$	$1.3 \cdot 10^{-11}$	$5.0 \cdot 10^{-6}$
	300*	$2.9 \cdot 10^{-28}$	$4.8 \cdot 10^{-11}$	$7.9 \cdot 10^{-6}$
Least Squares	1	$1.7 \cdot 10^{-13}$	$8.0 \cdot 10^{-02}$	$4.6 \cdot 10^{-1}$
	5	$3.2 \cdot 10^{-23}$	$1.1 \cdot 10^{-11}$	$5.2 \cdot 10^{-6}$
	10*	$2.3 \cdot 10^{-28}$	$4.8 \cdot 10^{-11}$	$7.9 \cdot 10^{-6}$

the graph norm gradient to deliver results on par with descent according to ∇_{H}. Interestingly enough, the least squares method needs 10 steps to reach the optimal residual. The oscillations depicted in the right plot of figure 2 are due to numerical errors of the linear solver. If one further decreases λ, descent with respect to $\nabla_{\mathrm{G},\lambda}$ becomes more similar to solving the linear least squares problem and starts to show oscillating behavior, too.

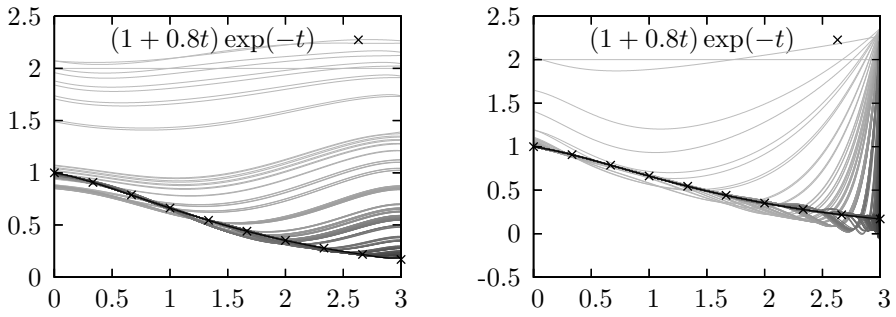


FIGURE 1. Some descent steps with ∇_{H} (left) and with $\nabla_{\mathrm{G},1}$ (right).

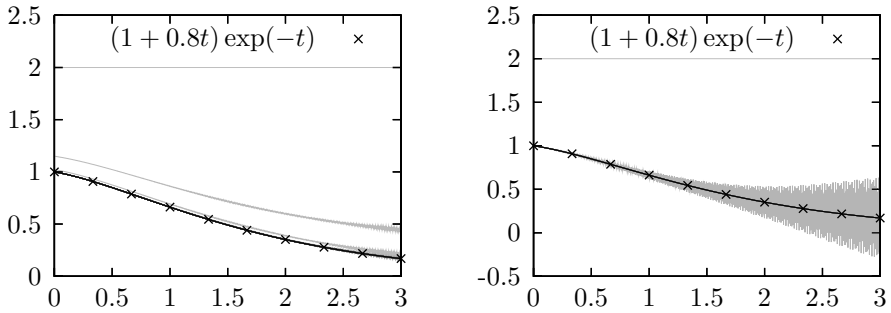


FIGURE 2. left: some descent steps with $\nabla_{G,10^{-10}}$, right: development solving least squares problem

7.3. **More Involved Test Problems.** We ran our implementation on two problems of the IVP Testset [28] of the University of Bari (formerly released by CWI Amsterdam).

7.3.1. *Chemical Akzo Nobel problem.* This is a initial value problem consisting of a stiff semi-linear DAE of index 1 with $n = m = 6$. As the square-root is taken in several components the domain of ψ is not the whole space $H^1(0, 180; \mathbb{R}^6)$. This poses difficulties for the line search algorithm, as we have to ensure that we do not leave the domain of definition, decreasing the step width if necessary.

Another problem is that our implementation does not cope well with stiff problems. This is not surprising, as we did not incorporate any mechanisms to refine the grid (also compare to section 7.6). But the algorithm tries to find the optimal numerical solution a fixed given grid. We can, however, apply our method in a step-by-step manner by solving the equation on short intervals if we ensure by appropriate initial conditions that we can glue the solutions together. This works reasonably well and as a byproduct ensures that the solution stays in the domain of definition. Still, it is computationally quite demanding to get highly exact numerical results with this approach.

7.3.2. *Andrews' Squeezing Mechanism.* The equation of Andrews' squeezing mechanism is a non-stiff semi-linear index 3 DAE with 14 differential and 13 algebraic equations. It is described in detail by Hairer and Wanner [18, Section VII.7]. However, our implementation is not designed to cope with the extreme scaling issues between the individual components and has problems with this equation. To get a rough idea, be aware that at a fixed position the solution vector has the structure $y = \begin{pmatrix} q & \dot{q} & \ddot{q} & \lambda \end{pmatrix}^T$, where $q \in \mathbb{R}^7$ and $\lambda \in \mathbb{R}^6$. For the correct solution, the magnitude of y_5 is of order 10^{-2} on the interval $[0, 0.003]$, whereas y_{16} is of order 10^6. Without appropriate preconditioning, the linear solver cannot handle this discrepancy. A preconditioning strategy for index 3 DAE arising from multibody systems is proposed in [6] in the context of Newmark integration schemes.

7.4. **Solution Space Estimation.** The structure of the set of solutions of general DAE can be quite complicated. Even the calculation of consistent initial values is a non-trivial task (see section 1). The presented steepest descent method allows to start from any initial function. The choice of the initial function determines the calculated numerical solution. Thus, it is natural to ask whether valuable

information about the set of solutions can be acquired by running the steepest descent method multiple times for a large number of sufficiently different initial functions.

The question arises which initial functions to choose to get sufficiently diverse solutions and whether this method really has the tendency to exhaust the full set of solutions. Here, we always took linear functions as initial estimates. We generated them by prescribing random function values, uniformly distributed in $[-2,2]^n$, at both interval boundary points of $[0,T]$. Figure 3 shows plots of initial values of corresponding numerical solutions. The 3-dimensional plots belong to specifically designed linear DAE with known solution space.

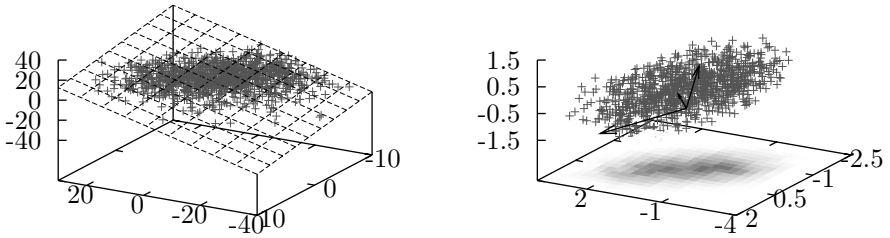


FIGURE 3. A 2-dimensional space (left) and 3-dimentional space (right) of consistent initial values.

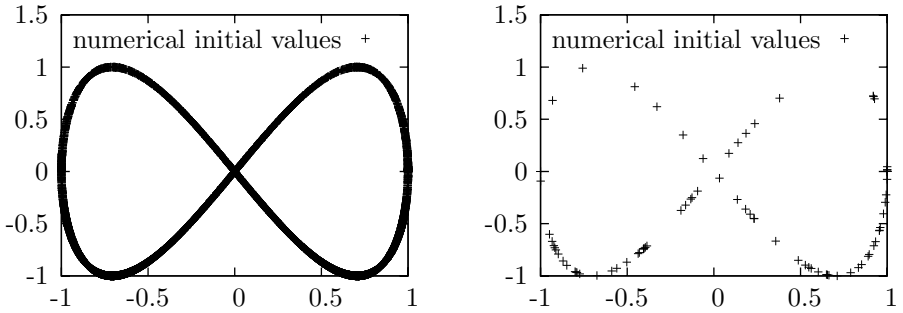


FIGURE 4. Numerical initial values of the Figure Eight Problem; left: 10000 points, right: 100 points

7.4.1. *The Figure Eight Problem.* We ran our steepest descent implementation 10000 times, starting with random initial functions, on the problem

$$\begin{cases} u_1(t)^2 + u_1'(t)^2 - 1 = 0, \\ 2u_1(t)u_1'(t) - u_2(t) = 0, \end{cases}$$

where $t \in [0,1]$. We used a uniform grid with $N = 300$ grid points, a damping factor of 0.85 and made 30 descent steps using the gradient $\nabla_{G,10^{-5}}$. Usually the residual dropped below 10^{-16} within the first 12 descent steps. We only cared about the function value at $t = 0$ of the numerical solution. The plot of these numerically consistent initial conditions is shown in figure 4. From the picture it

becomes evident why this problem is called the "figure eight problem". In the right picture, only the first 100 initial values are shown. Obviously the distribution is not uniform. We found this equation in the sample problem collection of a DAE solver written by Rheinboldt [35].

We remark that this is the only problem in the whole article to which lemma 5.1 does not apply. In fact, the example actually does not fit into our framework because F as in (5.2) does not even map into $L^2(0,T;\mathbb{R}^m)$. Nevertheless, the numerics work without any difficulties.

7.4.2. *Dimension Estimation.* For linear DAE the set of consistent initial values, in the following denoted by $C \subset \mathbb{R}^n$, is a finite dimensional affine subspace of $\mathbb{R}^n$. To estimate its dimension we produce a large amount of numerically consistent initial values. When using these vectors to determine the dimension of C one faces the problem that they are numerically disturbed and have almost surely full dimension.

Assume we have a number of numerical initial values $v_1, \dots v_N$, $N \gg n$. A first idea to determine the dimension is based on the Gaussian elimination method with *complete pivoting*, which is also called *total* or *maximal pivoting* [16, Section 3.4]. First we shift the vectors v_j such that they have sample mean 0. If one takes the matrix A containing all the v_j as columns one expects the total pivots during the Gaussian elimination process to decrease and to relate in some way to the dimension of this point set. The index of the most significant drop would then be the estimate of the dimension of C.

A second approach utilizes *principal component analysis* (PCA), which is a well established method to reduce the dimension of data by introducing a change to a lower dimensional new coordinate system. An introduction to PCA is given by Lay [23, Section 8.5]. More precisely, for some target dimension $d \in \mathbb{N}_0$, $d \le n$, PCA can be used to determine a shift $\hat{c} \in \mathbb{R}^n$ and an orthogonal projection $\hat{P} \in \mathbb{R}^{n\times n}$ onto a d-dimensional linear subspace of $\mathbb{R}^n$ which minimizes the quadratic error

$$E(d) := \min\Big\{\sum_{i=1}^{N} \|(I-P)(v_i-c)\|_2^2 : c \in \mathbb{R}^n,\ P \text{ a } d\text{-dim. orth. proj.}\Big\}.$$

Since we are only interested in the error $E(d)$, the necessary calculations are relatively simple. One can show that $\hat{c}$ can be chosen independently of d as the sample mean of the vectors v_i, $i = 1, \dots, N$. The scaled covariance matrix

$$S := \sum_{i=1}^{N} (r_i - \hat{c})(r_i - \hat{c})^T,$$

is a positive semi-definite matrix whose eigenvalues we denote by $\lambda_1 \ge \cdots \ge \lambda_n$. It can be shown that

$$E(d) = \sum_{i=d+1}^{n} \lambda_i.$$

Using a numerical method to calculate eigenvalues, we are able to compute $E(d)$. We estimate the dimension as $d^* = 0$ if the entire variance $E(0)$ is below some bound and otherwise as the minimal $d^* \in \mathbb{N}_0$ such that $E(d^*) \le 0.999 \cdot E(0)$. Hence we reduce the dimension as much as possible while still preserving 99.9% of the entire variance.

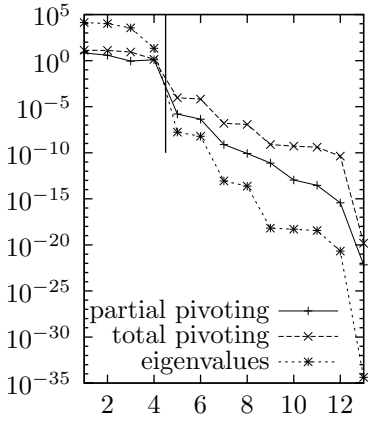
We successfully applied this to several small linear DAE with index up to 3. Using the theory of regular matrix pencils [18, Section VII.1] it is easy to construct

linear DAE with constant coefficients with certain index and known solution space. Table 5 shows the results for such a constructed index 3 system with $n = m = 13$ where the solution space is 4-dimensional. We point out that the method using pivots also works surprisingly well. However, the automatic decision should be improved. In this example, it finds 99.9% of the variance in a subspace of dimension 3 although the most significant drop occurs between the fourth and fifth eigenvalue.

Unfortunately, this method has some deficiencies. One problem is the dependency on highly accurate numerical solutions, which are harder to obtain for the higher index case. Additionally, it is heavily dependent on the scaling of the problem and tends to underestimate the real dimension for problems with a higher dimensional solution space because of insufficient diversity of the numerical solutions. The latter problem could possibly be addressed by a more sophisticated choice of initial functions.

TABLE 5. left: pivots and eigenvalues for an index 3 DAE with 4-dimensional solution space, right: logarithmic plot of the pivots' absolute values and the eigenvalues

Number	**Total pivots**	**Eigenvalues**
1	-13.1	13468.2
2	12.8	10269.7
3	8.5	3517.9
4	-1.3	22.0
5	$0.9 \cdot 10^{-04}$	$1.6 \cdot 10^{-08}$
6	$0.6 \cdot 10^{-04}$	$6.2 \cdot 10^{-09}$
7	$-1.5 \cdot 10^{-07}$	$8.5 \cdot 10^{-14}$
8	$-1.1 \cdot 10^{-07}$	$2.3 \cdot 10^{-14}$
9	$7.8 \cdot 10^{-10}$	$6.5 \cdot 10^{-19}$
10	$5.2 \cdot 10^{-10}$	$4.9 \cdot 10^{-19}$
11	$-3.9 \cdot 10^{-10}$	$3.6 \cdot 10^{-19}$
12	$4.2 \cdot 10^{-11}$	$2.1 \cdot 10^{-21}$
13	$-1.5 \cdot 10^{-20}$	$4.1 \cdot 10^{-35}$



7.4.3. *Detection of Non-Uniqueness.* A DAE might have an infinite dimensional solution space, and even for given initial values the problem need not have a unique solution. An example of a linear DAE which exhibits non-uniqueness for some given initial value is

$$\begin{pmatrix} -t & t^2 \\ -1 & t \end{pmatrix} u' + \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} u = 0, \quad u(0) = 0 \text{ for } t \in [0, 2]. \tag{7.2}$$

This problem is taken from Kunkel and Mehrmann [20, Chapter 8, Exercise 3]. The left plot in figure 5 shows the first component of 100 numerical solutions having started at random linear functions satisfying the initial condition of equation (7.2). The residuals of the numerical solutions are of magnitude 10^{-12}.

Another interesting example problem was presented by Ascher and Spiteri [3, Example 2] as a test problem for their boundary value problem solver `COLDAE`. This problem is a boundary value problem admitting exactly 2 different classical

solutions. One can simplify it to an equivalent initial value problem as follows, without changing the behavior of our program significantly.

$$\begin{aligned} u'(t) &= y(t) + \cos(t) \\ 0 &= (u(t) - \sin(t))(y(t) - \exp(t)) \end{aligned} \quad \text{on } [0,1], \text{ where } u(0) = 0 \tag{7.3}$$

The two solutions are

$$\begin{pmatrix} u(t) \\ y(t) \end{pmatrix} = \begin{pmatrix} \sin(t) + \exp(t) - 1 \\ \exp(t) \end{pmatrix} \quad \text{and} \quad \begin{pmatrix} u(t) \\ y(t) \end{pmatrix} = \begin{pmatrix} \sin(t) \\ 0 \end{pmatrix}.$$

For the numerics we used a uniform grid with $N = 1000$ and a damping factor of 0.85. Applying our method using the discretized H^1 norm and starting from random linear initial functions which satisfy the initial conditions, we experienced poor convergence speed and arrived at an approximation for the second solution most of the time. The right plot in figure 5 shows a numerical solution after 10000 steps using H^1 descent. Its residual is about $6.41 \cdot 10^{-8}$ and $\tilde{y}$ obviously deviates from $y(t) = 0$, especially at the interval boundaries. Using the discretized graph norm the steepest descent converged in around 100 steps to a numerical solution with a residual of about 10^{-20}.

However, for both gradients certain initial functions resulted in unexpected numerical solutions, e. g. the left plot in figure 6. The jump in the component $\tilde{y}$ and the bend in the first component harm classical differentiability. However, looking at equation (7.3) we see that $y(t)$ need not be differentiable if we understand the problem as in section 3. In fact, our numerical solution resembles a weak solution in $D(\bar{A})$ (compare to proposition 3.1). This shows that even with finite differences we are able to find weak solutions in accordance with the abstract theory. This particular plot was generated using graph norm descent. Ordinary H^1 descent has a tendency to smooth things out which interferes with jump solutions and results in worse convergence.

However, our discretization with finite differences yields problems. Using steepest descent with ∇_G, we also get solutions as depicted in the right picture in figure 6. This originates in the central differences of (6.3) used for the numerical approximation of the derivative. In this case we experience a decoupling of the grid into even and odd indices where $\tilde{y}(t) \approx \exp(t)$. There, $\tilde{u}$ oscillates between two possible local solutions. Using the finite element method instead would solve such deficiencies.

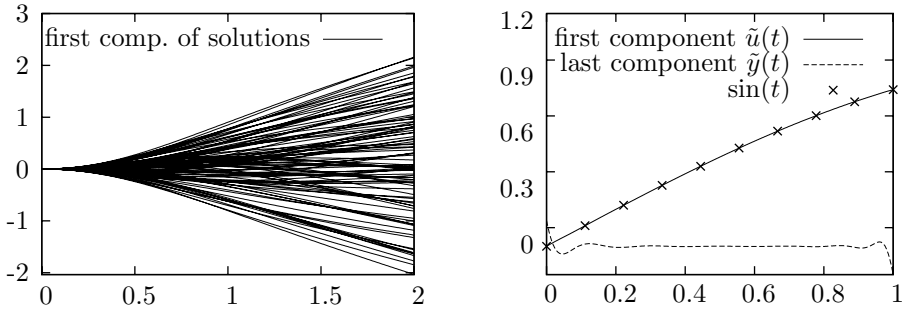


FIGURE 5. left: example of IVP without unique solution,
right: results for (7.3) after H^1 descent

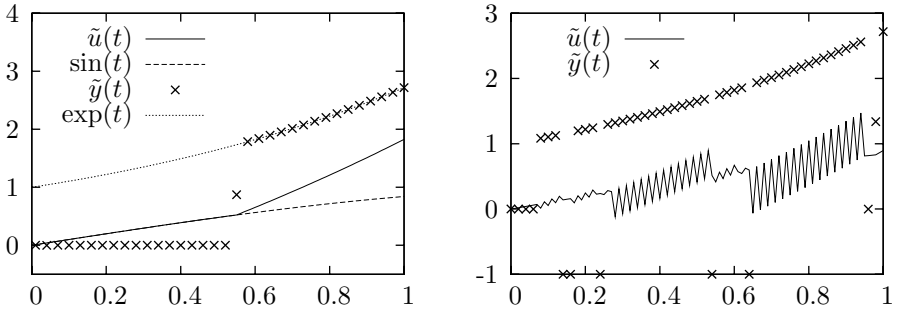


FIGURE 6. left: generalized numerical solution for (7.3),
right: example illustrating problems of the discretization

7.5. **Implementation.** We implemented our software in the `C++` programming language using the paradigm of compile-time polymorphism. It makes heavy use of the Standard Template Library and the basic linear algebra routines provided by the `ublas` library, which is part of the well known Boost project [9]. The software was developed using GNU/Linux on x86 architecture and compiled with the GCC `C++` compiler. For solving the involved linear systems we used the conjugate gradient method [33, Chapter 5], Gauss-Seidel with successive over-relaxation [16, Section 10.1] and the impressive PARDISO solver [37], being part of the Intel Math Kernel Library [29], which can solve sparse positive definite and indefinite systems efficiently via parallel factorization. We also used functions provided by the MKL to calculate the singular value decomposition of a matrix. We need this to construct projection matrices for general supplementary conditions. For auxiliary calculations we used the computer algebra system Maple.

7.6. **Possible Improvements.**

7.6.1. *Finite elements.* Finite element methods could turn out to be a rewarding alternative to the finite difference scheme admitting convergence even if the solutions fail to be smooth. This should be not too difficult to implement. Numerically it is more demanding because numerical integration methods have to be employed to get the coefficients for the chosen basis.

7.6.2. *Non-uniform grids.* Analyzing the structure of the given DAE, it may be possible to calculate an optimal grid for the discretization, or to refine the grid during the descent process. Refinements of the grid are technically easy to integrate, since new intermediate grid points can be inserted interpolating the current values in a neighborhood. However, updating to the new matrices is an expensive operation.

7.6.3. *Functionals defined only on subsets of $H^1(0,T;\mathbb{R}^n)$.* In Section 5, we assumed the function f to be defined on $[0,T]\times\mathbb{R}^n\times\mathbb{R}^n$. If f can only be evaluated on a subset of this space (e. g. because of a square root) the domain $D(\Psi)$ of Ψ is not the whole space $H^1(0,T;\mathbb{R}^n)$. Usual steepest descent does not respect this and can leave the domain, even if there exists a solution $u\in D(\Psi)$. We have discussed this phenomenon in section 7.3.1. This issue could probably be addressed by assigning a penalty to the Sobolev gradient calculation prohibiting strong tendencies towards the boundary of the domain $D(\Psi)$.

7.6.4. *Other projections.* The projection onto the feasible functions used in Section 6.6 does not have to be the orthogonal one. One can choose among all projections trying to find one with beneficial properties, i. e., a sparse projection that still is numerically stable. Basic building blocks of such matrices have been constructed in [7].

7.6.5. *Combination with conventional methods.* If desired, one can mix conventional techniques with our approach. For example one could estimate consistent initial conditions using Sobolev descent locally at the left boundary, then run a multi-step method to calculate a rough approximate solution, and then refine this initial guess again by Sobolev gradient methods on the whole interval.

7.6.6. *Implementation Issues.* Our step size control should be replaced by more robust line search algorithms enforcing Wolfe conditions, cf. [33, Chapter 3]. Error estimates, failure tolerance, and a decent user interface have to be provided. The efficiency of the whole algorithm has to be improved in order to meet the standards of current DAE solvers.

8. Conclusion

As pointed out before, the method of Sobolev steepest descent differs greatly from the usual step-by-step methods, thus introducing both new kinds of problems and advantages.

Our approach has some drawbacks. The procedure itself tends to be expensive in terms of runtime and memory usage compared to the conventional multi-step methods. It is complicated to generate an appropriate mesh, since we have to fix a mesh size a priori whereas step-by-step methods may adjust their mesh according to local error estimates. Such refinements can be put into practice in our setting, too, but changes of the mesh are expensive. Moreover, convergence of the Sobolev descent is guaranteed only under restrictive conditions. Hence currently the user has to investigate for each kind of DAE separately whether the algorithm behaves as desired. It certainly is an interesting problem for further research to find general conditions under which Sobolev descent finds a solution for a DAE.

Yet, a new technique also introduces new possibilities. We regard it as one of the main features that no initial conditions need to be supplied. Only some initial estimate for the solution is needed, not necessarily a good one. In general, it is difficult to find consistent initial conditions for DAE. In [5, Subsection 5.3.4] they state, that "Often the most difficult part of solving a DAE system in applications is to determine a consistent set of initial conditions with which to start the computation". For more information on this topic see [8] or [40, Chapter 2].

Another advantage is the possibility to impose arbitrary linear supplementary conditions, even non-local ones. The authors do not know of any other program that can handle such general data. In principle, the user can point the algorithm towards a solution with particular characteristics by providing it with a suitable initial estimate, although admittedly it is not clear in what sense the algorithm respects this hint. Moreover, no previous transformations such as differentiation of the equations have to be applied, and hence we do not artificially increase the number of equations and the numerical errors.

As the next step for the theory of solving DAE via Sobolev descent the authors suggest to generalize the results of section 3 to the non-autonomous, the semi-linear

and the fully non-linear case. We mention that the concept of the graph norm for Sobolev gradient descent is rather generic and easily generalizes to arbitrary differential problems, even involving non-local and partial differential operators, to which the theory could finally be extended. Although it is easier and more common to use an ordinary Sobolev space for all applications, we emphasize that using a metric more closely related to the equation itself obviously improves the algorithm. Thus this modification should at least be considered whenever Sobolev gradients are employed. It may provide some insight to discuss the effects of the different metrics and to compare the convergence theoretically.

As for the continuation of our practical efforts, one should consider to address the deficiencies of our implementation discussed in 7.6. In particular, it seems to be important to put 7.6.1 into practice since finite element methods are the more natural choice when working with Sobolev spaces.

Acknowledgments.
This article has been inspired by John W. Neuberger who suggested applying Sobolev gradient methods to the field of DAE. The authors would like to thank him for his patience and aid. This work has been started during the authors' stay at the University of North Texas as a guests.

Parts of this article were developed while one of the authors was sponsored by the graduate school "Mathematical Analysis of Evolution, Information and Complexity" of the University of Ulm.

References

[1] Y.A. Abramovich and C.D. Aliprantis, *An Invitation to Operator Theory*, American Mathematical Society, 2002.
[2] A. Ambrosetti and G. Prodi, *A Primer of Nonlinear Analysis*, Cambridge Univ. Press, 1993.
[3] U.M. Ascher and R.J. Spiteri, *Collocation Software for Boundary Value Differential-Algebraic Equations*, SIAM Journal on Scientific Computing **15** (1994), no. 4, 938–952.
[4] J. Appell and P.P. Zabrejko, *Nonlinear superposition operators*, Cambridge Univ. Press, 1990.
[5] K.E. Brenan, S.L. Campbell, and L.R. Petzold, *Numerical Solution of Initial-Value Problems in Differential-Algebraic Equations*, Elsevier Science Publishing, New York, 1989.
[6] C.L. Bottasso, D. Dopico, and L. Trainelli, *On the optimal scaling of index three DAEs in multibody dynamics*, Multibody System Dynamics **19** (2008), no. 1–2, 3–20.
[7] M.W. Berry, M.T. Heath, I. Kaneko, M. Lawo, R.J. Plemmons, and R.C. Ward, *An Algorithm to Compute a Sparse Basis of the Null Space*, Num. Mathematik **47** (1985), no. 4, 483–504.
[8] P.N. Brown, A.C. Hindmarsh, and L.R. Petzold, *Consistent Initial Condition Calculation for Differential-Algebraic Systems*, SIAM Journal on Scientific Computing **19** (1998), no. 5, 1495–1512.
[9] *Boost C++ Library*, `http://www.boost.org`.
[10] S.L. Campbell, *High-Index Differential Algebraic Equations*, Mechanics Based Design of Structures and Machines **23** (1995), no. 2, 199–222.
[11] J. Cash, *BVP and IVP software page*, `http://www.ma.ic.ac.uk/~jcash`.
[12] R. Chill, *The Łojasiewicz-Simon gradient inequality in Hilbert spaces*, `http://www.math.univ-metz.fr/~chill/procloja.pdf`, 2006.
[13] G.F. Carey, W.B. Richardson, C.S. Reed, and B.J. Mulvaney, *Circuit, Device and Process Simulation: Mathematical and Numerical Aspects*, Wiley, 1996.
[14] E.F. Costa, R.C. Vieira, A.R. Secchi, and E.C. Biscaia, *Dynamic simulation of high-index models of batch distillation processes*, Latin American Applied Research **33** (2003), 155–160.
[15] B.W. Gordon and H. Asada, *Modeling, Realization, and Simulation of Thermo-Fluid Systems Using Singularly Perturbed Sliding Manifolds*, Journal of Dynamic Systems, Measurement, and Control **122** (2000), 699–707.
[16] G.H. Golub and C.F. van Loan, *Matrix Computations*, third ed., John Hopkins University Press, Baltimore and London, 1996.

[17] A. Haraux, M. Ali Jendoubi, and O. Kavian, *Rate of decay to equilibrium in some semilinear parabolic equations*, Journal of Evolution Equations **3** (2003), no. 3, 463–484.
[18] E. Hairer and G. Wanner, *Solving ordinary differential equations, 2. Stiff and differential algebraic problems*, second revised ed., Springer Series in Computational Mathematics, vol. 14, Springer, 1996.
[19] A. Kumar and P. Daoutidis, *Control of nonlinear differential algebraic equation systems*, Chapman & Hall/CRC, 1999.
[20] P. Kunkel and V. Mehrmann, *Differential-Algebraic Equations: Analysis and Numerical Solution*, Textbooks in Mathematics, European Mathematical Society, 2006.
[21] J. Karátson and J.W. Neuberger, *Newton's method in the context of gradients*, Electronic Journal of Differential Equations **2007** (2007), no. 124, 1–13.
[22] S.A. Lang, *Differential and Riemannian Manifolds*, Springer, 1995.
[23] D.C. Lay, *Linear Algebra and Its Applications*, Addison-Wesley, 1994.
[24] W.T. Mahavier, *A Numerical Method for Solving Singular Differential Equations Utilizing Steepest Descent in Weighted Sobolev Spaces*, Ph.D. thesis, University of North Texas, 1995.
[25] W.T. Mahavier, *A convergence result for discreet steepest decent in weighted sobolev spaces*, Abstract and Applied Analysis **2** (1997), no. 1, 67–72.
[26] W.T. Mahavier, *A numerical method utilizing weighted Sobolev descent to solve singular differential equations*, Nonlinear World **4** (1997), no. 4, 435–456.
[27] W.T. Mahavier, *Weighted Sobolev Descent for singular first order partial Differential Equations*, Southwest Journal of Pure and Applied Mathematics **1** (1999), 41–50.
[28] F. Mazzia, F. Iavernaro, and C. Magherini, *Test Set for Initial Value Problem Solvers*, `http://pitagora.dm.uniba.it/~testset/`, 2006, Release 2.3 September 2006.
[29] *Intel Math Kernel Library*, `http://www.intel.com/cd/software/products/asmo-na/eng/perflib/mkl/`
[30] J.W. Neuberger, *Projection Methods for Linear and Nonlinear Systems of Partial Differential Equations*, Dundee Conference on Differential Equations, vol. 564, Springer Lecture Notes, 1976, pp. 341–349.
[31] J.W. Neuberger, *Sobolev gradients and differential equations*, Springer, 1997.
[32] R. Nittka and M. Sauter, *Implementation with source code and examples of Sobolev Gradients for Differential Algebraic Equations*, `http://cantor.mathematik.uni-ulm.de/m5/nittka/research/2007/sobolev_dae/`, 2007.
[33] J. Nocedal and S.J. Wright, *Numerical Optimization*, second ed., Springer Series in Operations Research, Springer, 2006.
[34] L. R. Petzold, C. W. Gear, and H. H. Hsu, *Differential-Algebraic Equations Revisited*, Proceedings Oberwolfach Workshop on Stiff Equations, Institut für Geometrie und Praktische Mathematik der TH Aachen, June 1981, Bericht 9.
[35] W.C. Rheinboldt, *Sample problems for* `dae_solve.tgz`, `http://www.netlib.org/ode/daesolve/`, 2000.
[36] S. Schulz, *Four Lectures on Differential-Algebraic Equations*, `http://www.math.auckland.ac.nz/Research/Reports/` (497), 2003.
[37] O. Schenk and K. Gärtner, *On fast factorization pivoting methods for sparse symmetric indefinite systems*, Electronic Transactions on Numerical Analysis **23** (2006), 158–179.
[38] C. Udriste, *Convex Functions and Optimization Methods on Riemannian Manifolds*, Springer, 1994.
[39] R. von Schwerin, *Multibody System Simulation: Numerical Methods, Algorithms, and Software*, Springer, 1999.
[40] K.D. Yeomans, *Initialization Issues in General Differential Algebraic Equation Integrators*, Ph.D. thesis, North Carolina State University, 1997.

Robin Nittka
University of Ulm, Institute of Applied Analysis, D-89069 Ulm, Germany
E-mail address: `robin.nittka@uni-ulm.de`

Manfred Sauter
University of Ulm, Institute of Applied Analysis, D-89069 Ulm, Germany
E-mail address: `manfred.sauter@uni-ulm.de`