ALGORITHMS THAT SATISFY A STOPPING CRITERION, PROBABLY

URI ASCHER AND FARBOD ROOSTA-KHORASANI *

Abstract.
Iterative numerical algorithms are typically equipped with a stopping criterion, where the iteration process is terminated when some error or misfit measure is deemed to be below a given tolerance. This is a useful setting for comparing algorithm performance, among other purposes.

However, in practical applications a precise value for such a tolerance is rarely known; rather, only some possibly vague idea of the desired quality of the numerical approximation is at hand. In this review paper we first discuss four case studies from different areas of numerical computation, where uncertainty in the error tolerance value and in the stopping criterion is revealed in different ways.

This leads us to think of approaches to relax the notion of exactly satisfying a tolerance value. We then concentrate on a *probabilistic* relaxation of the given tolerance in the context of our fourth case study which allows, for instance, derivation of proven bounds on the sample size of certain Monte Carlo methods. We describe an algorithm that becomes more efficient in a controlled way as the uncertainty in the tolerance increases, and demonstrate this in the context of some particular applications of inverse problems.

Key words. error tolerance, mathematical software, iterative method, inverse problem, Monte Carlo method, trace estimation, large scale simulation, DC resistivity

AMS subject classifications. 65C20, 65C05, 65M32, 65L05, 65F10

1. Introduction.

A typical iterative algorithm in numerical analysis and scientific computing requires a stopping criterion. Such an algorithm involves a sequence of generated iterates or steps, an error tolerance, and a method to compute (or estimate) some quantity related to the error. If this error quantity is below the tolerance then the iterative procedure is stopped and success is declared.

The actual manner in which the error in an iterate is estimated can vary all the way from being rather complex to being as simple as the normed difference between two consecutive iterates. Further, the "tolerance" may actually be a set of values involving combinations of absolute and relative error tolerances. There are several fine points to this, often application-dependent, that are typically incorporated in mathematical software packages (see for instance MATLAB's various packages for solving ordinary differential equation (ODE) or optimization problems). That makes some authors of introductory texts devote significant attention to the issue, while others attempt to ignore it as much as possible (cf. [15, 32, 4]). Let us choose here the middle way of considering a stopping criterion in a general form

$$\text{error\_estimate}(k) \le \rho, \tag{1.1}$$

where k is the iteration or step counter, and $\rho > 0$ is the tolerance, assumed given.

But now we ask, *is ρ really given?!* Related to this, we can also ask, *to what extent is the stopping criterion adequate?*

- The numerical analyst would certainly *like* ρ to be given. That is because their job is to invent new algorithms, prove various assertions regarding convergence, stability, efficiency, and so on, and compare the new algorithm to other known ones for a similar task. For the latter aspect, a rigid deterministic tolerance for a trustworthy error estimate is indispensable. Indeed, in research areas such as image processing where criteria of the form (1.1) do not seem to capture certain essential features and the "eye norm" rules, a good comparison between competing algorithms can be far more delicate. Moreover, accurate comparisons of algorithms that require stochastic input can be tricky in terms of reproducing claimed experimental results.

*Dept. of Computer Science, University of British Columbia, Vancouver, Canada ascher/farbod@cs.ubc.ca . This work was supported in part by NSERC Discovery Grant 84306.

- On the other hand, a practitioner who is the customer of numerical algorithms, applying them in the context of some complicated practical application that needs to be solved, will more often than not find it very hard to justify a particular choice of a precise value for ρ in (1.1).

Our first task in this review paper is to convince the reader that often in practice there is a significant uncertainty in the actual selection of a meaningful value for the error tolerance ρ, a value that must be satisfied. Furthermore, numerical analysts are also subconsciously aware of this fact of life, even though in most numerical analysis papers such a value is simply given, if at all, in the numerical examples section. Four typical yet different classes of problems and methods are considered in Section 2.

Once we are all convinced that there is usually a considerable uncertainty in the value of ρ (hence, we only know it "probably"), the next question is what to do with this notion. The answer varies, depending on the particular application and the situation at hand. In some cases, such as that of Section 2.1, the effective advice is to be more cautious, as mishaps can happen. In others, such as that of Section 2.2, we are simply led to acknowledge that the value of ρ may come from thin air (though one then concentrates on other aspects). But there are yet other classes of applications and algorithms, such as in Sections 2.3 and 2.4, for which it makes sense to attempt to quantify the uncertainty in the error tolerance ρ using a probabilistic framework. We are not proposing in this article to propagate an entire probability distribution for ρ: that would be excessive in most situations. But we do show, by studying an instance extended to a wide class of problems, that employing such a framework can be practical and profitable.

Following Section 2.4 we therefore consider in Section 3 a particular manner of relaxing the notion of a deterministic error tolerance, by allowing an estimate such as (1.1) to hold only within some given probability. Relaxing the notion of an error tolerance in such a way allows the development of theory towards an uncertainty quantification of Monte Carlo methods (e.g., [1, 8, 53, 37, 35]). We concentrate on the problem of estimating the trace of symmetric positive semi-definite (SPSD) matrices that are given implicitly, i.e., as a routine to compute their product with any vector of suitable size. We then use this to satisfy in a probabilistic sense an error tolerance for approximating a data misfit function for cases where many data sets are given. Some details of results of this type that are relevant in the present context and have been proved in [46, 47] are provided.

In Section 4 we apply the results of Section 3 to form a randomized algorithm for approximately solving inverse problems involving partial differential equation (PDE) systems. We concentrate on cases where many data sets are provided (which is typical in several important applications) that correspondingly require the solution of many PDEs. Conclusions and some additional general comments are offered in Section 5.

While this article concentrates on bringing to the fore a novel point of view, we also note that the observations in Section 2.2.1, and to a lesser extent also Section 2.3.1, are novel. The results displayed in Figures 2.1 and 4.1 are new.

2. Case studies.

In this section we consider four classes of problems and associated algorithms, in an attempt to highlight the use of different tests of the form (1.1) and in particular the implied level of uncertainty in the choice of ρ.

2.1. Stopping criterion in initial value ODE solvers.

Using a case study, we show in this section that numerical analysts, too, can be quick to not consider ρ as a "holy constant": we adapt to weaker conditions in different ways, depending on the situation and the advantage to be gained in relaxing the notion of an error tolerance and more generally an error criterion.

Let us consider an initial value ODE system in "time" t, written as

$$\frac{d\mathbf{u}}{dt} = \mathbf{f}(t, \mathbf{u}), \quad 0 \le t \le b, \tag{2.1a}$$

$$\mathbf{u}(0) = \mathbf{v}_0, \tag{2.1b}$$

with $\mathbf{v}_0$ a given initial value vector. A typical adaptive algorithm proceeds to generate pairs $(t_i, \mathbf{v}_i),\ i = 0, 1, 2, \ldots, N$, in N consecutive steps, thus forming a mesh π such that

$$\pi : 0 = t_0 < t_1 < \cdots < t_{N-1} < t_N = b,$$

and $\mathbf{v}_i \approx \mathbf{u}(t_i),\ i = 1, \ldots, N$.

Denoting the numerical solution on the mesh π by $\mathbf{v}^\pi$, and the restriction of the exact ODE solution to this mesh by $\mathbf{u}^\pi$, there are two general approaches for controlling the error in such an approximation.

- Given a tolerance value ρ, keep estimating the *global error* and refining the mesh (i.e., the gamut of step sizes) until roughly

  $$\|\mathbf{v}^\pi - \mathbf{u}^\pi\|_\infty \le \rho. \tag{2.2}$$

  Details of such methods can be found, for instance, in [29, 34, 13, 6].
  In (2.2) we could replace the absolute tolerance by a combination of absolute and relative tolerances, perhaps even different ones for different ODE equations. But that aspect is not what we concentrate on in this article.
- However, most general-purpose ODE codes estimate a *local error* measure for (1.1) instead, and refine the step size locally. Such a procedure advances one step at a time, and estimates the next step size using local information related to the local truncation error, or simply the difference between two approximate solutions for the next time level, one of which presumed to be significantly more accurate than the other.[1] For details see [29, 30, 6] and many references therein. In particular, the popular MATLAB codes `ode45` and `ode23s` use such a local error control.

The reason for employing local error control is that this allows for developing a much cheaper and yet more sensitive adaptive procedure, an advantage that cannot be had, for instance, for general boundary value ODE problems; see, e.g., [5].

But *does this always produce sensible results?!* The answer to this question is negative. A simple example to the contrary is the problem

$$\frac{du}{dt} = 100(u - \sin t) + \cos t, \quad u(0) = 0,\ b = 1.$$

Local truncation (or discretization) errors for this unstable initial value ODE propagate like $\exp(100t)$, a fact that is not reflected in the local behaviour of the exact solution $u(t) = \sin t$ on which the local error control is based. Thus, we may have a large error $\|\mathbf{v}^\pi - \mathbf{u}^\pi\|_\infty$ even if the local error estimate is bounded by ρ for a small value of ρ.

2.1.1. Local error control can be dangerous even for a stable ODE system. Still one can ask, *are we safe with local error control in case that we know that our ODE problem is stable?* Here, by "safe" we mean that the global error will not be much larger than the local truncation error in scaled form. The answer to this more subtle question turns out to be negative as well. The essential point is that the global error consists of an accumulation of contributions of local errors from previous time steps. If the ODE problem is asymptotically stable (typically, because it describes a damped motion) then local error contributions die away as time increases, often exponentially fast, so at some fixed time only the most recent local error contributions dominate in the sum of contributions that forms the global error. However, if the initial value ODE problem is merely marginally stable (which is the case for Hamiltonian systems) then local error contributions

[1] Recall that the local truncation error at some time $t = t_i$ is the amount by which the exact solution $\mathbf{u}^\pi$ fails to satisfy the scheme that defines $\mathbf{v}^\pi$ at this point. Furthermore, if at t_i, using the known $\mathbf{v}_i$ and a guess for t_{i+1}, we apply one step of two different Runge-Kutta methods of orders 4 and 5, say, then the difference of the two results at t_{i+1} gives an estimate for the error in the lower order method over this mesh subinterval.

propagate undamped, and their accumulation over many time steps can therefore be significantly larger than just one or a few such errors.[2]

For a simple concrete example, consider applying `ode45` with default tolerances to find the linear oscillator with a slowly varying frequency that satisfies the following initial value ODE for $p(t)$:

$$\frac{dq}{dt} = \lambda^2 p, \quad q(0) = 1,$$
$$\frac{dp}{dt} = -(1+t)^2 q, \quad p(0) = 0.$$

Here $\lambda > 0$ is a given parameter. Thus, $\mathbf{u} = (q, p)^T$ in the notation of (2.1). This is a Hamiltonian system, with the Hamiltonian function given by

$$H(q, p, t) = \frac{1}{2}\left[((1+t)q)^2 + (\lambda p)^2\right].$$

Now, since the ODE is not autonomous, the Hamiltonian is not constant in time. However, the *adiabatic invariant*

$$J(q, p, t) = H(q, p, t)/(1+t)$$

(see, e.g., [39, 7]) is almost constant for large λ, satisfying

$$[J(t) - J(0)]/J(0) = \mathcal{O}(\lambda^{-1})$$

over the interval $[0, 1]$. This condition means in particular that for $\lambda \gg 1$ and the initial values given above, $J(1) = J(0) + \mathcal{O}(\lambda^{-1}) \approx J(0)$.

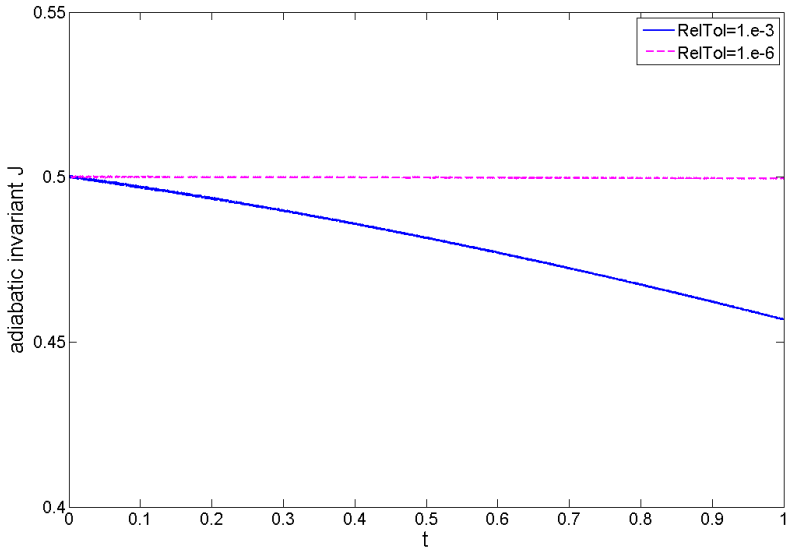


FIG. 2.1. *Adiabatic invariant approximations obtained using* MATLAB*'s package* `ode45` *with default tolerances (solid blue) and stricter tolerances (dashed magenta).*

Figure 2.1 depicts two curves approximating the adiabatic invariant for $\lambda = 1000$. Displayed are the calculated curve using `ode45` with default tolerances (absolute=1.e-6, relative=1.e-3), as well

[2]The local error control basically seeks to equalize the magnitude of such local errors at different time steps.

as what is obtained upon using `ode45` with the stricter relative tolerance RelTol=1.e-6. From the figure it is clear that when using the looser tolerance, the resulting approximation for $J(1)$ differs from $J(0)$ by far more than what λ^{-1} =1.e-3 and RelTol=1.e-3 would indicate, while the stricter tolerance gives a qualitatively correct result, using the "eye norm". Annoyingly, the qualitatively incorrect result does not look like "noise": while not being physical, it looks downright plausible, and hence could be misleading for an unsuspecting user. Adding to the pain is the fact that this occurs for default tolerance values, an option that a vast majority of users that we have observed is likely to automatically select. ♦

A similar observation holds when trying to approximate the phase portrait or other properties of an autonomous Hamiltonian ODE system over a long time interval using `ode45` with default tolerances: this may produce qualitatively wrong results. See for instance Figures 16.12 and 16.13 in [4]: the Fermi-Pasta-Ulam problem solved there is described in detail in Chapter 1 of [28]. What we have just shown here is that the phenomenon can arise also for a very modest system of *two linear ODEs that do not satisfy any exact invariant*.

We hasten to add that the documentation of `ode45` (or other such codes) does not propose to deliver anything like (2.2). Rather, the tolerance is just a sort of a knob that is turned to control local error size. However, this does not explain the popularity of such codes despite their limited offers of assurance in terms of qualitatively correct results.

Our key point in the present section is the following: we propose that one reason for the popularity of ODE codes that use only local error control is that in applications one rarely knows a precise value for ρ as used in (2.2) anyway. (Conversely, if such a global error tolerance value is known and is important then codes employing a global error control, and not `ode45`, should be used.) Opting for local error control over global error control can therefore be seen as one specific way of adjusting mathematical software in a deterministic sense to realistic uncertainties regarding the desired accuracy.[3]

2.2. Stopping criterion in iterative methods for linear systems. In this case study, extending basic textbook material, we argue not only that tolerance values used by numerical analysts are often determined solely for the purpose of the comparison of methods (rather than arising from an actual application), but also that this can have unexpected effects on such comparisons. In Section 2.2.1 we further make some novel, and we think intriguing, observations on PDE discretizations, which arise in the present context although having little to do with our main theme.

Consider the problem of finding $\mathbf{u}$ satisfying

$$A\mathbf{u} = \mathbf{b}, \tag{2.3}$$

where A is a given $s \times s$ symmetric positive definite matrix such that one can efficiently carry out matrix-vector products $A\mathbf{v}$ for any suitable vector $\mathbf{v}$, but decomposing the matrix directly (and occasionally, even looking at its elements) is too inefficient and as such is "prohibited". We relate to such a matrix as being given *implicitly*. The right hand side vector $\mathbf{b}$ is given as well.

An iterative method for solving (2.3) generates a sequence of iterates $\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_k, \ldots$ for a given initial guess $\mathbf{u}_0$. Denote by $\mathbf{r}_k = \mathbf{b} - A\mathbf{u}_k$ the residual in the kth iterate. The MINRES method, or its simpler version Orthomin(2), can be applied to reduce the residual norm so that

$$\|\mathbf{r}_k\|_2 \le \rho \|\mathbf{r}_0\|_2 \tag{2.4}$$

in a number of iterations k that in general is at worst $\mathcal{O}\left(\sqrt{\kappa(A)}\right)$, where $\kappa(A) = \|A\|_2 \|A^{-1}\|_2$ is the condition number of the matrix A. Below in Table 2.1 we refer to this method as MR. The

[3] A practical advice for this case study is to be particularly careful with local error control when the ODE problem has little or no damping. Beyond that we offer no additional method or analysis for this case study in the present article.

more popular conjugate gradient (CG) method generally performs comparably in practice. We refer to [24] for the precise statements of convergence bounds and their proofs.

A well-known and simpler-looking family of *gradient descent* methods is given by

$$\mathbf{u}_{k+1} = \mathbf{u}_k + \alpha_k \mathbf{r}_k, \tag{2.5}$$

where the scalar $\alpha_k > 0$ is the step size. Such methods have recently come under intense scrutiny because of applications in stochastic programming and sparse solution recovery. Thus, it makes sense to evaluate and understand them in the simplest context of (2.3), even though it is commonly agreed that for the strict purpose of solving (2.3) iteratively, CG cannot be significantly beaten. Note that (2.5) can be viewed as a forward Euler discretization of the artificial time ODE

$$\frac{d\mathbf{u}}{dt} = -A\mathbf{u} + \mathbf{b}, \tag{2.6}$$

with "time" step size α_k. Next we consider two choices of this step size.

The steepest descent (SD) variant of (2.5) is obtained by the greedy (exact) line search for the function

$$f(\mathbf{u}) = \frac{1}{2}\mathbf{u}^T A\mathbf{u} - \mathbf{b}^T\mathbf{u},$$

which gives

$$\alpha_k = \alpha_k^{SD} = \frac{\mathbf{r}_k^T\mathbf{r}_k}{\mathbf{r}_k^T A\mathbf{r}_k} \equiv \frac{(\mathbf{r}_k, \mathbf{r}_k)}{(\mathbf{r}_k, A\mathbf{r}_k)} \equiv \frac{\|\mathbf{r}_k\|_2^2}{\|\mathbf{r}_k\|_A^2}.$$

However, SD is very slow, requiring k in (2.4) to be proportional to $\kappa(A)$; see, e.g., [2].[4]

A more enigmatic choice in (2.5) is the lagged steepest descent (LSD) step size

$$\alpha_k = \alpha_k^{LSD} = \frac{(\mathbf{r}_{k-1}, \mathbf{r}_{k-1})}{(\mathbf{r}_{k-1}, A\mathbf{r}_{k-1})}.$$

It was first proposed in [9] and practically used for instance in [10, 16]. To the best of our knowledge, there is no known a priori bound on how many iterations as a function of $\kappa(A)$ are required to satisfy (2.4) with this method [9, 44, 22, 17].

We next compare these four methods in a typical fashion for a prototypical PDE example, where we consider the model Poisson problem

$$-\Delta u = 1, \quad 0 < x, y < 1,$$

subject to homogeneous Dirichlet BC, and discretized by the usual 5-point difference scheme on a $\sqrt{s} \times \sqrt{s}$ uniform mesh. Denote the reshaped vector of mesh unknowns by $\mathbf{u} \in \mathbb{R}^s$. The largest eigenvalue of the resulting matrix A in (2.3) is $\lambda_{\max} = 4h^{-2}(1 + \cos(\pi h))$, and the smallest is $\lambda_{\min} = 4h^{-2}(1 - \cos(\pi h))$, where $h = 1/(\sqrt{s} + 1)$. Hence by Taylor expansion of $\cos(\pi h)$, for $h \ll 1$ the condition number is essentially proportional to s:

$$\kappa(A) = \frac{\lambda_{\max}}{\lambda_{\min}} \approx \left(\frac{2}{\pi}\right)^2 s.$$

In Table 2.1 we list iteration counts required to satisfy (2.4) with $\rho = 10^{-7}$, starting with $\mathbf{u}_0 = \mathbf{0}$.

[4] The precise statement of error bounds for CG and SD in terms of the error $\mathbf{e}_k = \mathbf{u} - \mathbf{u}_k$ uses the A-norm, or "energy norm", and reads

$$\|\mathbf{e}_k\|_A \le 2\left(\frac{\sqrt{\kappa(A)} - 1}{\sqrt{\kappa(A)} + 1}\right)^k \|\mathbf{e}_0\|_A, \quad \text{for CG},$$

$$\|\mathbf{e}_k\|_A \le \left(\frac{\kappa(A) - 1}{\kappa(A) + 1}\right)^k \|\mathbf{e}_0\|_A, \quad \text{for SD}.$$

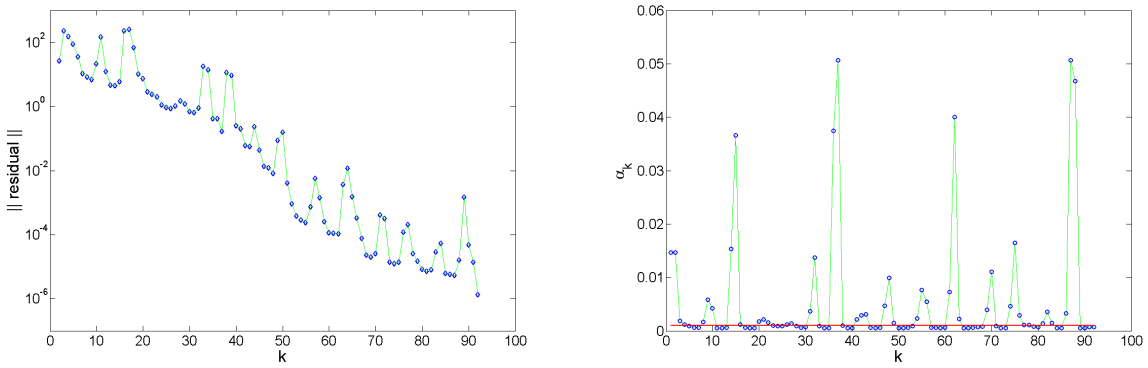
See [24].

s	MR	CG	SD	LSD
7^2	9	9	196	45
15^2	26	26	820	91
31^2	54	55	3,337	261
63^2	107	109	13,427	632
127^2	212	216	53,800	1,249

TABLE 2.1

Iteration counts required to satisfy (2.4) for the Poisson problem with tolerance $\rho = 10^{-7}$ and different mesh sizes s.

But now, returning to the topic of the present article, we ask, *why insist on $\rho = 10^{-7}$?* Indeed, the usual observation that one draws from the columns of values for MR, CG and SD in a table such as Table 2.1, is that the first two grow like $\sqrt{\kappa(A)} \propto \sqrt{s}$ while the latter grows like $\kappa(A) \propto s$. The value of ρ, so long as it is not too large, does not matter at all!



(a) Residuals (b) Step sizes

FIG. 2.2. *Relative residuals and step sizes for solving the model Poisson problem using LSD on a 15×15 mesh. The red line in (b) is the forward Euler stability limit.*

And yet, this is not quite the case for the LSD iteration counts. These do not decrease in the same orderly fashion as the others, even though they are far better (in the sense of being significantly smaller) than those for SD. Indeed, this method is chaotic [17], and the residual norm decreases rather non-monotonically, see Figure 2.2(a). Thus, the iteration counts in Table 2.1 correspond to the iteration number $k = k^*$ where the rough-looking relative residual norm first records a value below the tolerance ρ. Unlike the other three methods, here the particular value of the tolerance, picked just to be concrete, does play an unwanted role in the relative values, as a function of s, or $\kappa(A)$, of the listed iteration counts. ♦

2.2.1. How large can the step size be? – benefitting from a discretize-first approach. While on the subject of LSD, let us also remark that gradient descent for this example can be viewed as an application of the forward Euler method via (2.6) to find the steady state of the heat equation

$$\frac{\partial u}{\partial t} = \Delta u + 1. \tag{2.7}$$

(Homogeneous Dirichlet boundary conditions for all $t > 0$ are assumed already incorporated into the Poisson-Laplace operator Δ.)

The forward Euler time-stepping method with step size α, applied as a semi-discretization in time to this PDE problem, reads

$$u_{k+1} = u_k + \alpha(\Delta u_k + 1), \tag{2.8}$$

where $u_k = u_k(x, y)$. However, this cannot be stable for any $\alpha > 0$. To see this, think of what happens to a discontinuous initial condition function $u_0(x, y)$ when its spatial derivatives are repeatedly taken as in (2.8).

Next, let us discretize in space first, obtaining (2.6), and follow this with a forward Euler discretization in time (i.e., apply gradient descent (2.5)). It is well-known that if the step size $\alpha_k = \alpha$ is constant for all k then stability requires the restriction $\alpha \le .25/s$. In fact, the precise stability statement is $\alpha \le 2/\lambda_{\max}$, where $\lambda_{\max}$ is the largest eigenvalue of the matrix A given above. This restriction becomes more severe as s grows, and in the limit $s \to \infty$ we obtain the unconditionally unstable method (2.8).

So far, things are as expected. But now, if α_k is allowed to vary with k then no such stability requirement holds for all k! Indeed, if $\alpha_k = 1/\lambda_{\min}$ then although this does enlarge the magnitude of high-frequency components in the residual (we are using multigrid jargon here; see [51]), it may decrease low frequency ones rapidly. The temporary increase in high frequency component magnitudes can then be alleviated by next taking a few small step sizes which are known to be particularly efficient in reducing those high frequency amplitudes. Next we observe that as the uniform mesh size s grows, $\lambda_{\min} \to 2\pi$, which is independent of the discretization mesh width. So this large step size is in effect constant as the mesh is refined, and yet the method can be convergent. This argument follows from a discretize-first approach, and cannot be seen from (2.8).

Returning to LSD as a method for automatically choosing the step sizes in the forward Euler iteration, the results are recorded in Figure 2.2(b). In fact, in this particular example we find upon decreasing the spatial mesh width that $\max_k \alpha_k \approx .05$, independently of s. Thus, what was merely possible in the previous paragraph is in fact automatically achieved by an LSD method that provably converges to steady state.

To the best of our knowledge, this result is novel, and moreover we have found that it is unintuitive to many experts in numerical PDEs. Hence we chose to incorporate it in this article, despite the fact that it does not directly relate to its main theme.

2.3. Data fitting and inverse problems. In the previous two case studies we have encountered situations where the intuitive use of an error tolerance within a stopping criterion could differ widely (and wildly) from the notion that is embodied in (1.1) for the consumer of numerical analysts' products. We next consider a family of problems where the value of ρ in a particular criterion (1.1) is more directly relevant.

Suppose we are given observed data $\mathbf{d} \in \mathbb{R}^l$ and a *forward operator* $f_i(m)$, $i = 1, \ldots, l$, which provides predicted data for each instance of a distributed parameter function m. The (unknown) function m is defined in some domain Ω in physical space and possibly time. We are particularly interested here in problems where f involves the solution u in Ω of some linear PDE system, sampled in some way at the points where the observed data are provided. Further, for a given mesh π discretizing Ω, we consider a corresponding discretization (i.e., nodal representation) of m and u, as well as the differential operator. Reshaping these mesh functions into vectors we can write the resulting approximation of the forward operator as

$$\mathbf{f}(\mathbf{m}) = P\mathbf{u} = PG(\mathbf{m})\mathbf{q}, \tag{2.9}$$

where the right hand side vector $\mathbf{q}$ is commonly referred to as a source, G is a discrete Green's function (i.e., the inverse of the discretized PDE operator), $\mathbf{u} = G(\mathbf{m})\mathbf{q}$ is the field (i.e., the PDE solution, here an interim quantity), and P is a projection matrix that projects the field to the locations where the data values $\mathbf{d}$ are given.

This setup is typical in the thriving research area of inverse problems; see, e.g., [20, 52]. A specific example is provided in Section 4.

The inverse problem is to find $\mathbf{m}$ such that the predicted and observed data agree to within noise $\boldsymbol{\eta}$: ideally,

$$\mathbf{d} = \mathbf{f}(\mathbf{m}) + \boldsymbol{\eta}. \tag{2.10}$$

To obtain such a model $\mathbf{m}$ that satisfies (2.10) we need to estimate the *misfit function* $\phi(\mathbf{m})$, i.e., the normed difference between observed data $\mathbf{d}$ and predicted data $\mathbf{f}(\mathbf{m})$. An iterative algorithm is then designed to sufficiently reduce this misfit function. But, *which norm should we use to define the misfit function?*

It is customary to conveniently assume that the noise satisfies $\boldsymbol{\eta} \sim \mathcal{N}(0, \sigma^2 I)$, i.e., that the noise is normally distributed with a scaled identity for the covariance matrix, where σ is the standard deviation. Then the maximum likelihood (ML) data misfit function is simply the squared ℓ_2-norm[5]

$$\phi(\mathbf{m}) = \|\mathbf{f}(\mathbf{m}) - \mathbf{d}\|_2^2. \tag{2.11}$$

In this case, the celebrated Morozov *discrepancy principle* yields the stopping criterion

$$\phi(\mathbf{m}) \le \rho, \quad \text{where } \rho = \sigma^2 l, \tag{2.12}$$

see, e.g., [20, 40, 38]. So, here is a class of problems where we do have a meaningful and directly usable tolerance value!

Assuming that a known tolerance ρ must be satisfied as in (2.12) is often too rigid in practice, because realistic data do not quite satisfy the assumptions that have led to (2.12) and (2.11). Well-known techniques such as L-curve and GCV (see, e.g., [52]) are specifically designed to handle more general and practical cases where (2.12) cannot be used or justified. Also, if (2.12) is used then a typical algorithm would try to find $\mathbf{m}$ such that $\phi(\mathbf{m})$ is (smaller but) not much smaller than ρ, because having $\phi(\mathbf{m})$ too small would correspond to fitting the noise – an effect one wants to avoid. The latter argument and practice do not follow from (2.12).

Moreover, keeping the misfit function $\phi(\mathbf{m})$ in check does not necessarily imply a quality reconstruction (i.e., an acceptable approximation $\mathbf{m}$ for the "true solution" $\mathbf{m}_{\text{true}}$, which can be an elusive notion in itself). However, $\phi(\mathbf{m})$, and not direct approximations of $\|\mathbf{m}_{\text{true}} - \mathbf{m}\|$, is what one typically has to work with.[6] So any additional a priori information is often incorporated through some regularization.

Still, despite all the cautious comments in the preceding two paragraphs, we have in (2.12) in a sense a more meaningful practical expression for stopping an iterative algorithm than hitherto.

2.3.1. Regularization and constrained formulations. Typically there is a need to regularize the inverse problem, and often this is done by adding a regularization term to (2.11). Thus, one attempts to *approximately* solve the Tikhonov-type problem

$$\min_{\mathbf{m}} \ \phi(\mathbf{m}) + \lambda R(\mathbf{m}), \tag{2.13}$$

where $R(\mathbf{m}) \ge 0$ is a prior (we are thinking of some norm or semi-norm of $\mathbf{m}$), and $\lambda \ge 0$ is a regularization parameter. Two other forms of the same problem (2.13) immediately arise upon interpreting λ or λ^{-1} as a Lagrange multiplier. These are the constrained optimization formulations

$$\min_{\mathbf{m}} \ \phi(\mathbf{m}), \quad \text{s.t. } R(\mathbf{m}) \le \tau; \quad \text{and} \tag{2.14}$$

[5] For a more general symmetric positive definite covariance matrix Σ, such that $\boldsymbol{\eta} \sim \mathcal{N}(0, \Sigma)$, we get weighted least squares, or an "energy norm", with the weight matrix Σ^{-1} for ϕ. But let's not go there in this article.

[6] The situation here is different from that in Section 2.1, where the choice of local error criterion over a global one was made based on convenience and efficiency considerations. Here, although controlling $\phi(\mathbf{m})$ is merely a necessary and not sufficient condition for obtaining a quality reconstruction $\mathbf{m}$, it is usually all we have to work with.

$$\min_{\mathbf{m}} \; R(\mathbf{m}), \quad \text{s.t. } \phi(\mathbf{m}) \le \rho. \tag{2.15}$$

The non-negative parameters ρ, τ and λ are related to one another in a nontrivial manner that makes these three formulations indeed equivalent; see, e.g., [10].

Each of these formulations has its fan club. The inevitable discussion regarding which is best becomes more heated when R involves the ℓ_1-norm, corresponding to a prior that favours some form of sparsity; see [18] and references therein. This is so not only because sparsity is popular and extremely useful, but also because use of the ℓ_1-norm introduces lower smoothness in $R(\mathbf{m})$. In such circumstances, the formulation (2.14) can be more directly amenable to efficient solution techniques (see the equivocal [10]), while (2.15) has the advantage that the tolerance ρ is known (cf. (2.12)) with far less a priori uncertainty than τ or λ. Thus, the popular question of choosing the most practically advantageous formulation from among (2.13)–(2.15) (see, e.g., [31]) is tied to the topic of the present article.

2.4. Least squares data fitting with many data sets. In our fourth case study we generalize the setting of Section 2.3, allowing not only one but several (indeed, many) data sets $\mathbf{d}_i$, $i = 1, \ldots, s$, where each data set has length l, so $\mathbf{d}_i \in \mathbb{R}^l$. Unless s is small, working with such a problem can be prohibitively expensive, so we next use randomized approximations to the misfit function. This in turn provides a new meaning to the uncertainty in the given tolerance ρ, which we further explore in later sections.

We can conveniently define an $l \times s$ data matrix D whose ith column is $\mathbf{d}_i$. Correspondingly, there are forward operators

$$\mathbf{f}_i(\mathbf{m}) \equiv \mathbf{f}(\mathbf{m}, \mathbf{q}_i) = PG(\mathbf{m})\mathbf{q}_i, \quad i = 1, \ldots, s, \tag{2.16a}$$

where $\mathbf{q}_i$ are different sources, and we arrange the forward operator in form of an $l \times s$ matrix function $F(\mathbf{m})$ which has $\mathbf{f}_i$ as its ith column. The inverse problem is to find $\mathbf{m}$ such that the predicted and observed data agree to within noise,

$$\mathbf{d}_i = \mathbf{f}_i(\mathbf{m}) + \boldsymbol{\eta}_i, \quad i = 1, 2, \ldots, s. \tag{2.16b}$$

Assuming next that $\boldsymbol{\eta}_i \sim \mathcal{N}(0, \sigma^2 I)$, the ML data misfit function is

$$\phi(\mathbf{m}) = \sum_{i=1}^{s} \|\mathbf{f}_i(\mathbf{m}) - \mathbf{d}_i\|_2^2 = \|F(\mathbf{m}) - D\|_F^2, \tag{2.16c}$$

where the subscript F denotes the Frobenius norm. In this case, the discrepancy principle yields the stopping criterion

$$\phi(\mathbf{m}) \le \rho, \quad \text{where } \rho = \sigma^2 ls. \tag{2.16d}$$

So, as in Section 2.3, provided that all assumptions hold we have a decent idea of a stopping tolerance value for an iterative method that generates iterates $\mathbf{m}_k, k = 1, 2, \ldots$, in an attempt to decrease ϕ until (2.16d) holds with approximate equality.

However, when s is very large, since s evaluations of $G(\mathbf{m})\mathbf{q}_i$ are required just to form $F(\mathbf{m})$ for a given $\mathbf{m}$, we obtain an instance where the matrix $B = F(\mathbf{m}) - D$ can be prohibitively expensive to calculate explicitly (Section 4 provides a concrete example). Hence the matrix $A = B^T B$ is implicit symmetric positive semi-definite (SPSD) (cf. Section 2.2). Furthermore, we have

$$\phi(\mathbf{m}) = \|B\|_F^2 = tr(A) = \mathbb{E}(\|B\mathbf{w}\|_2^2), \tag{2.17}$$

where $tr(A)$ denotes the trace of the matrix A, $\mathbb{E}$ stands for expectation, and $\mathbf{w}$ stands for a random vector drawn from any distribution satisfying $\mathbb{E}(\mathbf{w}\mathbf{w}^T) = \mathbb{I}$.

Inexpensive approximate alternatives to working with all s data sets throughout an algorithm for finding a suitable $\mathbf{m}$ may be obtained by approximating the misfit function $\phi(\mathbf{m})$ in (2.16c). Specifically, the rightmost expression in (2.17) suggests a Monte Carlo sampling method, obtaining

$$\widehat{\phi}(\mathbf{m}, n) := \frac{1}{n}\sum_{j=1}^{n} \|B(\mathbf{m})\mathbf{w}_j\|_2^2 \approx \phi(\mathbf{m}). \tag{2.18}$$

Note that $\widehat{\phi}(\mathbf{m}, n)$ is an *unbiased estimator* of $\phi(\mathbf{m})$, as we have $\phi(\mathbf{m}) = \mathbb{E}(\widehat{\phi}(\mathbf{m}, n))$. For the forward operators (2.16a), if $n \ll s$ (in words, the sample size n is much smaller than the total number of data sets s) then this procedure yields a very efficient algorithm for approximating the misfit (2.16c). This is so because

$$\sum_{i=1}^{s} \mathbf{f}(\mathbf{m}, \mathbf{q}_i)w_i = \mathbf{f}(\mathbf{m}, \sum_{i=1}^{s} w_i\mathbf{q}_i),$$

which can be computed with a single evaluation of $\mathbf{f}$ per realization of the random vector $\mathbf{w} = (w_1, \ldots, w_s)^T$, so in total n (rather than s) such evaluations are required [26, 48]. An example utilizing (2.18) is considered in Section 4 below.

But now it is natural to ask, *how large should n be?* Generally speaking, the larger n is, the closer $\widehat{\phi}(\mathbf{m}, n)$ is to $\phi(\mathbf{m})$. So we are led to ask how important it is to pedantically satisfy (2.16d). This in turn brings up the question that is the common refrain of this section, namely, the extent to which the stopping criterion is really known and as such must be obeyed. Here, a higher uncertainty in ρ and (2.16) allows a more efficient solution algorithm because a smaller n will suffice. In the context of the present randomized algorithm it is therefore natural to consider satisfying the error criterion (1.1) only within some probability range. To concentrate on this aspect we isolate it by assuming that ρ in (2.16d) is given and the error model that enables this is approximately valid.

3. Probabilistic relaxation of a stopping criterion. The previous section details four different case studies which highlight the fact of life that in applications an error tolerance for stopping an algorithm is rarely known with absolute certainty. Thus, we can say that such a tolerance is only "probably" known. Yet in some situations, it is also possible to assign it a more precise meaning in terms of statistical probability. This holds true for the case study in Section 2.4, with which we proceed below. Thus, in the present section we consider a way to relax (1.1), which is more systematic and also allows for further theoretical developments. Specifically, we consider satisfying a tolerance in a probabilistic sense.

Suppose we seek an unbiased estimator $\hat{g}(x)$ to a real valued function $g(x)$ (so $\mathbb{E}(\hat{g}(x)) = g(x)$, where $\mathbb{E}$ denotes expectation). Then, given a pair of values (ε, δ), both small and positive, we require

$$Pr\big(|\hat{g}(x) - g(x)| \le \varepsilon\ |g(x)|\big) \ge 1 - \delta. \tag{3.1}$$

The parameters ε and δ relate to the relative accuracy and the probabilistic guarantee of such an estimation, respectively [1]. In Section 3.1 we consider applying this notion to the problem of trace estimation. Then in Section 3.2 we apply the results of Section 3.1 to the case study of Section 2.4. The uncertainty in the stopping tolerance ρ is quantified in terms of parameters ε and δ towards the end of this section.

3.1. Estimating the trace of an implicit matrix. As in Section 2.2 we consider an $s \times s$ matrix A that is given only implicitly, through matrix-vector products. We assume that $A = B^T B$, where B is some real rectangular matrix with s columns, and wish to approximate the trace, $tr(A) = \sum_{i=1}^{s} a_{i,i}$, without having the luxury of knowing the diagonal elements $a_{i,i}$. This task has several applications; see [8, 46]. We consider it here mainly as a preparatory step, given the appearance of the trace in (2.17).

Note that if $\mathbf{x}$ is the ith column of the $s \times s$ identity matrix then $(\mathbf{x}, A\mathbf{x}) = a_{i,i}$. Hence, $tr(A)$ can be calculated in s matrix-vector products. However, s is very large so we want a cheaper approximation. Towards that goal, observe that if $\mathbf{w}$ stands for a random vector drawn from a probability distribution $\mathcal{D}$ satisfying $\mathbb{E}[\mathbf{w}\mathbf{w}^T] = I$, then

$$tr(A) = \mathbb{E}[(\mathbf{w}, A\mathbf{w})]. \tag{3.2}$$

So, we consider the Monte Carlo approximation $tr(A) \approx tr_{\mathcal{D}}(A)$, defined by

$$tr_{\mathcal{D}}(A) := \frac{1}{n}\sum_{i=1}^{n}(\mathbf{w}_i, A\mathbf{w}_i) = \frac{1}{n}\sum_{i=1}^{n}\|B\mathbf{w}_i\|_2^2 \ , \tag{3.3}$$

where $\mathbf{w}_i$ are drawn from the distribution $\mathcal{D}$, and hopefully $n \ll s$.

The next question is, *how small can we take n to be?* In other words, how can we quantify the uncertainty in such an approximation? Here is where the pair (ε, δ) appearing in (3.1) comes in handy for deriving probabilistic necessary and sufficient conditions on the size of n. Below we state two results that were proved in [46, 47].

Let us restate (3.1) in the present context as

$$Pr\big(|tr_{\mathcal{D}}(A) - tr(A)| \le \varepsilon\ |tr(A)|\big) \ge 1 - \delta. \tag{3.4}$$

Further, given the pair (ε, δ) (both values positive and small), define the constant

$$c = c(\varepsilon, \delta) = \varepsilon^{-2}\ln(2/\delta). \tag{3.5}$$

This constant appears in all estimates of the type stated in Theorem 3.1 below. Note its strong dependence on ε and much weaker dependence on δ, in the sense of how rapidly c grows as these values shrink.

THEOREM 3.1.

- *Let $\mathcal{D}$ be the Gaussian (i.e., standard normal) probability distribution. Then the probabilistic bound (3.4) holds if*

$$n \ge 8c(\varepsilon, \delta).$$

- *Let $\mathcal{D}$ be the Rademacher probability distribution [36], namely, for each component of $\mathbf{w} = (w_1, \ldots, w_s)^T$, $Pr(w_j = 1) = Pr(w_j = -1) = 1/2$. (The components of $\mathbf{w}$ are i.i.d.) Then the probabilistic bound (3.4) holds if*

$$n \ge 6c(\varepsilon, \delta).$$

The property that stands out in these bounds is that they are independent of the matrix size s (reminiscent in this sense to a multigrid method for the Poisson example in Section 2.2) and of any property of the matrix A other than it being SPSD. The latter also has a downside, of course, in suggesting that such bounds may not be very tight.

Theorem 3.1 provides *sufficient* bounds on n, and not very sharp ones at that; however, these bounds do have a charming simplicity. Next we give better sufficient bounds, as well as *necessary* bounds, for the case where $\mathcal{D}$ is the Gaussian distribution [47]. Simplicity, however, shall have to be sacrificed. For this purpose, we write (3.4), with a minor abuse of notation, as

$$\Pr\Big(tr_n(A) \ge (1-\varepsilon)tr(A)\Big) \ge 1 - \delta, \tag{3.6a}$$

$$\Pr\Big(tr_n(A) \le (1+\varepsilon)tr(A)\Big) \ge 1 - \delta. \tag{3.6b}$$

Also, denote by $Q_n \sim \chi^2_n$ a chi-squared random variable of degree n, and set $Q(n) = \frac{Q_n}{n}$.

THEOREM 3.2 (Necessary and sufficient condition for (3.6)). *Given an SPSD matrix A of rank r and parameters (ε, δ) as above, the following hold:*

(i) Sufficient *condition for* (3.6a)*: there exists some integer* $n_0 \geq 1$ *such that*

$$\Pr\left(Q(n_0) < (1-\varepsilon)\right) \leq \delta. \tag{3.7}$$

Furthermore, (3.6a) *holds for all* $n \geq n_0$.

(ii) Sufficient *condition for* (3.6b)*: if the inequality*

$$\Pr\left(Q(n_0) \leq (1+\varepsilon)\right) \geq 1-\delta \tag{3.8}$$

is satisfied for some $n_0 > \varepsilon^{-1}$*, then* (3.6b) *holds with* $n = n_0$*. Furthermore, there is always an* $n_0 > \varepsilon^{-2}$ *such that* (3.8) *is satisfied and, for such* n_0*, it follows that* (3.6b) *holds for all* $n \geq n_0$.

(iii) Necessary *condition for* (3.6a)*: if* (3.6a) *holds for some* $n_0 \geq 1$*, then for all* $n \geq n_0$

$$\Pr\left(Q(nr) < (1-\varepsilon)\right) \leq \delta. \tag{3.9}$$

(iv) Necessary *condition for* (3.6b)*: if* (3.6b) *holds for some* $n_0 > \varepsilon^{-1}$*, then*

$$\Pr\left(Q(nr) \leq (1+\varepsilon)\right) \geq 1-\delta, \tag{3.10}$$

with $n = n_0$*. Furthermore, if* $n_0 > \varepsilon^{-2}r^{-2}$*, then* (3.10) *holds for all* $n \geq n_0$.

The necessary conditions in Theorem 3.2 indicate that the lower bound on the *smallest* "true" n that satisfies (3.4) grows as the rank of A decreases (regardless of A's size s, $s \geq r$). For $r = 1$ the necessary and sufficient bounds coincide, which indicates a form of tightness.

Are these necessary bounds "a bug or a feature"? We argue that they can be both. Examples can be easily found where Hutchinson's method [36] (employing Rademacher's distribution) performs better than Gaussian, requiring a smaller n for a small rank matrix A. On the other hand, if other factors come in (as in the application of Section 4 below) which make the practical use of the Gaussian and Hutchinson methods comparable, then the availability of both necessary and sufficient conditions for the Gaussian distribution allow a better chance to quantify the error, probabilistically, in cases where the lower and upper bounds are close. This is taken up next.

3.2. Least squares data fitting with many data sets revisited. We now return to the problem considered in Section 2.4, and apply the results of the previous section to (2.18) and (2.16d). Thus, according to (3.6), in the check for termination of our iterative algorithm at the next iterate $\mathbf{m}_{k+1}$, we consider replacing the condition

$$\phi(\mathbf{m}_{k+1}) \leq \rho \tag{3.11a}$$

by either

$$\widehat{\phi}(\mathbf{m}_{k+1}, n_t) \leq (1-\varepsilon)\rho, \quad \text{or} \tag{3.11b}$$

$$\widehat{\phi}(\mathbf{m}_{k+1}, n_t) \leq (1+\varepsilon)\rho, \tag{3.11c}$$

for a suitable $n = n_t$ that is governed by Theorem 3.2 with a prescribed pair (ε, δ). If (3.11b) holds, then it follows with a probability of at least $(1-\delta)$ that (3.11a) holds. On the other hand, if (3.11c) does *not* hold, then we can conclude with a probability of at least $(1-\delta)$ that (3.11a) is *not* satisfied. In other words, unlike (3.11b), a successful (3.11c) is only necessary and not sufficient for concluding that (3.11a) holds with the prescribed probability $1-\delta$.

What are the connections among these three parameters, ρ*,* δ *and* ε*?!* The parameter ρ is the deterministic but not necessarily too trustworthy error tolerance appearing in (3.11a), much like the tolerance in Section 2.1. Next, we can reflect the uncertainty in the value of ρ by choosing an appropriately large δ (≤ 1). Smaller values of δ reflect a higher certainty in ρ and a more rigid stopping criterion (translating into using a larger n_t). For instance, success of (3.11b) is equivalent

to making a statement on the probability that a positive "test" result will be a "true" positive. This is formally given by the conditional probability statement

$$Pr\left(\phi(\mathbf{m}_{k+1}) \le \rho \mid \widehat{\phi}(\mathbf{m}_{k+1}, n_t) \le (1-\varepsilon)\rho\right) \ge 1-\delta.$$

Note that, once the condition in this statement is given, the rest only involves ρ and δ. So the tolerance ρ is augmented by the probability parameter δ. The third parameter ε governs the false positives/negatives (i.e., the probability that the test will yield a positive/negative result, if in fact (3.11a) is false/true), where a false positive is given by

$$Pr\left(\widehat{\phi}(\mathbf{m}_{k+1}, n_t) \le (1-\varepsilon)\rho \mid \phi(\mathbf{m}_{k+1}) > \rho\right),$$

while a false negative is

$$Pr\left(\widehat{\phi}(\mathbf{m}_{k+1}, n_t) > (1-\varepsilon)\rho \mid \phi(\mathbf{m}_{k+1}) \le \rho\right).$$

We note in passing that such efficient algorithms as described above can also be obtained with some extensions of the probability distribution assumption on the noise, which lead to *weighted* least squares generalizing (2.16c).

In summary, we have demonstrated in this section an approach for quantifying the uncertainty in the stopping criterion by deriving such a procedure for the case study described in Section 2.4.

4. An inverse problem with many data sets. The purpose of this section is to demonstrate the ideas developed in Section 3 for the methods of Section 2.4 on a concrete example.

Inverse problems of the sort described in (2.16) arise frequently in practice, and applications include electromagnetic data inversion in mining exploration (e.g., [41, 19, 25, 42]), seismic data inversion in oil exploration (e.g., [21, 33, 45]), diffuse optical tomography (DOT) (e.g., [3, 11]), quantitative photo-acoustic tomography (QPAT) (e.g., [23, 54]), direct current (DC) resistivity (e.g., [50, 43, 27, 26, 17]), and electrical impedance tomography (EIT) (e.g., [12, 14, 18]). Exploiting many data sets currently appears to be particularly popular in exploration geophysics, and our example can be viewed as mimicking a DC resistivity setup.

Our entire setup is the same as in [47] (built in turn on [17, 48]), where many details that are omitted here can be found. This allows us to concentrate next on those issues that are most relevant in the present article. The PDE has the form

$$\nabla \cdot (\mu(\mathbf{x}) \operatorname{grad} u) = q(\mathbf{x}), \quad \mathbf{x} \in \Omega, \tag{4.1a}$$

where $\Omega \subset \mathbb{R}^d$, $d = 2$ or 3, and $\mu(\mathbf{x}) \ge \mu_0 > 0$ is a conductivity function which may be rough (e.g., only piecewise continuous). An appropriate transfer function ψ is selected so that, point-wise, $\mu(\mathbf{x}) = \psi(m(\mathbf{x}))$. For example, ψ can be chosen so as to ensure that the conductivity stays positive and bounded away from 0, as well as to incorporate bounds, which are often known in practice. The matrix G in (2.16a) is a discrete Green's function for (4.1a) subject to the homogeneous Neumann boundary conditions

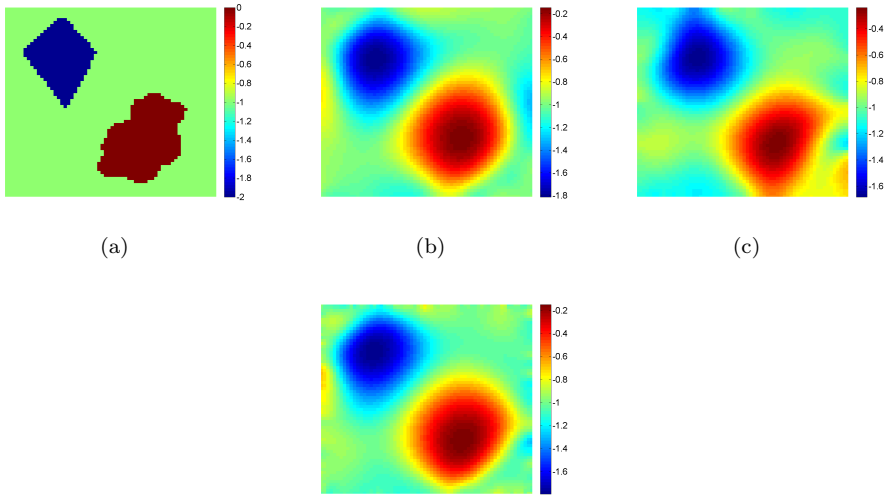
$$\frac{\partial u}{\partial n} = 0, \quad \mathbf{x} \in \partial\Omega. \tag{4.1b}$$

In other words, evaluating $\mathbf{f}_i(\mathbf{m})$ for given $\mathbf{m}$ and i requires the approximate solution of the PDE problem (4.1). Given the nonlinearity in $\mathbf{m}$ (which requires an iterative method for the nonlinear least squares problem of minimizing $\phi(\mathbf{m})$) and the number of data sets s, the PDE count can easily climb without employing the Monte Carlo approximations described in Sections 2.4 and 3.2.

The variant of our algorithm used here employs unbiased estimators of the form (2.18) on four occasions during an iteration k:

1. Use sample size n_k for an approximate stabilized Gauss-Newton (ASGN) iteration. This requires also a corresponding approximate gradient, and n_k is set heuristically from one iteration to the next as described next.
2. Use sample size n_c for a cross validation step, where we check whether
$$(1-\varepsilon)\widehat{\phi}(\mathbf{m}_{k+1}, n_c) \le (1+\varepsilon)\widehat{\phi}(\mathbf{m}_k, n_c) \tag{4.2}$$
holds. If it does not then we judge that the sample size n_k is too small for a meaningful ASGN iteration, increase it by a set factor (e.g., 2) and reiterate; otherwise, continue. We set n_c using a given pair ε_c, δ_c.
3. Use sample size n_u for an uncertainty check, asking if
$$\widehat{\phi}(\mathbf{m}_{k+1}, n_u) \le (1-\varepsilon)\rho. \tag{4.3}$$
We set n_u using a given pair ε_u, δ_u. If (4.3) holds then we next check for possible termination.
4. Use sample size n_t for a stopping criterion check, asking if (3.11c) holds. If yes then terminate the algorithm. We set n_t using a given pair ε_t, δ_t.

The last three steps allow for uncertainty quantification according to Theorem 3.2.



(a) (b) (c)

(d)

FIG. 4.1. *Plots of log-conductivity: (a) True model; (b) Vanilla recovery with $s = 3,969$; (c) Vanilla recovery with $s = 49$; (d) Monte Carlo recovery with $s = 3,969$. The vanilla recovery using only 49 measurement sets is clearly inferior, showing that a large number of measurement sets can be crucial for better reconstructions. The recovery using our algorithm, however, is comparable in quality to Vanilla with the same s. The quantifier values used in our algorithm were: $(\varepsilon_c, \delta_c) = (0.05, 0.3)$, $(\varepsilon_u, \delta_u) = (0.1, 0.3)$ and $(\varepsilon_t, \delta_t) = (0.1, 0.1)$.*

Figure 4.1 depicts an example of a piecewise constant "true model" $\mathbf{m}_{\text{true}}$ consisting of two homogeneous but different bodies in a homogeneous background (a). This is used to synthesize noisy partial-boundary data sets that are then used in turn to approximately recover the model. The large dynamical range of the conductivities, together with the fact that the data is available only on less than half of the boundary, contribute to the difficulty in obtaining good quality reconstructions. The term "Vanilla" refers to using all s available data sets for each task during the algorithm. This costs 527,877 PDE solves[7] for $s = 3,969$ (b) and 5,733 PDE solves for $s = 49$ (c). However, the quality

[7]Fortunately, the discrete Green's function G does not depend on i in (2.16a). Hence, if the problem is small enough that a direct method can be used to construct G, i.e., perform one LU decomposition at each iteration k, then the task of solving half a million PDEs just for comparison sake becomes less daunting.

of reconstruction using the smaller number of data sets is clearly inferior. On the other hand, using our algorithm yields a recovery (d) that is comparable to Vanilla but at the cost of only 5,142 PDE solves. The latter cost is about 1% that of Vanilla and is comparable in order of magnitude to that of evaluating $\phi(\mathbf{m})$ once!

4.1. TV and stochastic methods. This section, like Section 2.2.1, is not directly related to the main theme of this article, but it arises from the present discussion and has significant merit on its own. The two comments below are not strongly related to each other.

- The specific example considered above is used also in [47], except that the objective function there includes a total variation (TV) regularization. This represents usage of additional a priori information (namely, that the true model is discontinuous with otherwise flat regions), whereas here an implicit ℓ_2-based regularization has been employed without such knowledge regarding the true solution. The results in Figures 4.3(b) and 4.4(v) there correspond to our Figures 4.1(b) and 4.1(d), respectively, and as expected, they look sharper in [47]. On the other hand, a comparative glance at Figure 4.3(c) there vs the present Figure 4.1(c) reveals that the ℓ_1-based technique can be inferior to the ℓ_2-based one, even for recovering a piecewise constant solution! Essentially, even for this special solution form TV shines only with sufficiently good data, and here "sufficiently good" translates to "many data sets". This intuitively obvious observation does not appear to be well-known (see [18]).
- If we run the code used to obtain the results displayed in this section and in [47] several times (having at first settled on whether or not to use TV regularization), different solutions $\mathbf{m}$ are obtained because of the different way that the noisy data sets are sampled. The *variance* of such solutions may provide information, or indication, on the extent to which, albeit having done our best to control the misfit function $\phi(\mathbf{m})$, we can trust the quality of the reconstruction [49]. A large variance in $\mathbf{m}$ suggests a capricious dependence of the solution on the noisy data (that is given indirectly through $F(\mathbf{m})$), and it would decrease our confidence in the computed results using a particular bias such as TV regularization. Having such information is an advantage of the stochastic algorithm over a similar deterministic one.

5. Conclusions and further notes. Mathematical software packages typically offer a default option for the error tolerances used in their implementation. Users often select this default option without much further thinking, at times almost automatically. This in itself suggests that practical occasions where the practitioner does not really have a good hold of a precise tolerance value are abundant. However, since it is often convenient to assume having such a value, and convenience may breed complacency, surprises may arise. We have considered in Section 2 four case studies which highlight various aspects of this uncertainty in a tolerance value for a stopping criterion.

Recognizing that there can often be a significant uncertainty regarding the actual tolerance value and the stopping criterion, we have subsequently considered the relaxation of the setting into a probabilistic one, and demonstrated its benefit in the context of the case study of Section 2.4. The environment defined by (3.1) or (3.6), although well-known in other research areas, is relatively new (but not entirely untried) in the numerical analysis community. It allows, among other benefits, specifying an amount of trust in a given tolerance using two parameters that can be tuned, as well as the development of bounds on the sample size of certain Monte Carlo methods, as described in Section 3. In Section 4 we have then applied this setting in the context of a particular inverse problem involving the solution of many PDEs, and we have obtained some uncertainty quantification for a rather efficient algorithm solving a large scale problem.

There are several aspects of our topic that remain untouched in this article. For instance, there is no discussion of the varying nature of the error quantity that is being measured (which strongly differs across the subsections of Section 2, from solution error through residual error through data misfit error for an ill-posed problem to stochastic quantities that relate even less closely to the solution error). Also, we have not mentioned that complex algorithms often involve sub-tasks such

as solving a linear system of equations iteratively, or employing generalized cross validation (GCV) to obtain a tolerance value, or invoking some nonlinear optimization routine, which themselves require some stopping criteria: thus, several occurrences of tolerances in one solution algorithm are common. In Section 3, we have made the choice of concentrating on bounding the sample size n and not, for example, on minimizing the variance as in [36].

What we have done here is to highlight an often ignored yet rather fundamental issue, namely, the uncertainty that is often present in stopping criteria, from different angles. Subsequently, we have pointed at and demonstrated a promising approach (or direction of thought) that is not currently common in the scientific computing community, namely, treating stopping criteria probabilistically. Along the way we have also made in context several observations (Sections 2.1.1, 2.2.1, 2.3.1 and 4.1) from an array of fields of numerical computation, observations which we believe are not common knowledge.

Acknowledgments. The authors thank Eldad Haber and Arieh Iserles for several fruitful discussions.

REFERENCES

[1] D. Achlioptas. Database-friendly random projections. In *ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems, PODS 01*, volume 20, pages 274–281, 2001.
[2] H. Akaike. On a successive transformation of probability distribution and its application to the analysis of the optimum gradient method. *Ann. Inst. Stat. Math. Tokyo*, 11:1–16, 1959.
[3] S R Arridge. Optical tomography in medical imaging. *Inverse Problems*, 15(2):R41, 1999.
[4] U. Ascher and C. Greif. *First Course in Numerical Methods*. Computational Science and Engineering. SIAM, 2011.
[5] U. Ascher, R. Mattheij, and R. Russell. *Numerical Solution of Boundary Value Problems for Ordinary Differential Equations*. Classics. SIAM, 1995.
[6] U. Ascher and L. Petzold. *Computer Methods for Ordinary Differential and Differential-Algebraic Equations*. SIAM, 1998.
[7] U. Ascher and S. Reich. The midpoint scheme and variants for hamiltonian systems: advantages and pitfalls. *SIAM J. Scient. Comput.*, 21:1045–1065, 1999.
[8] H. Avron and S. Toledo. Randomized algorithms for estimating the trace of an implicit symmetric positive semi-definite matrix. *JACM*, 58(2), 2011. Article 8.
[9] J. Barzilai and J. Borwein. Two point step size gradient methods. *IMA J. Num. Anal.*, 8:141–148, 1988.
[10] E. van den Berg and M. P. Friedlander. Probing the pareto frontier for basis pursuit solutions. *SIAM J. Scient. Comput.*, 31(2):890–912, 2008.
[11] D.A. Boas, D.H. Brooks, E.L. Miller, C. A. DiMarzio, M. Kilmer, R.J. Gaudette, and Q. Zhang. Imaging the body with diffuse optical tomography. *Signal Processing Magazine, IEEE*, 18(6):57–75, 2001.
[12] L. Borcea, J. G. Berryman, and G. C. Papanicolaou. High-contrast impedance tomography. *Inverse Problems*, 12:835–858, 1996.
[13] Y. Cao and L. Petzold. A posteriori error estimation and global error control for ordinary differential equations by the adjoint method. *SIAM J. Scient. Comput.*, 26:359–374, 2004.
[14] M. Cheney, D. Isaacson, and J. C. Newell. Electrical impedance tomography. *SIAM Review*, 41:85–101, 1999.
[15] G. Dahlquist and A. Bjorck. *Numerical Methods*. Prentice-Hall, 1974.
[16] Y. Dai and R. Fletcher. Projected Barzilai-Borwein methods for large-scale box-constrained quadratic programming. *Numerische. Math.*, 100:21–47, 2005.
[17] K. van den Doel and U. Ascher. The chaotic nature of faster gradient descent methods. *J. Scient. Comput.*, 48, 2011. DOI: 10.1007/s10915-011-9521-3.
[18] K. van den Doel, U. Ascher, and E. Haber. The lost honour of ℓ_2-based regularization. *Radon Series in Computational and Applied Math*, 2013. M. Cullen, M. Freitag, S. Kindermann and R. Scheinchl (Eds).
[19] O. Dorn, E. L. Miller, and C. M. Rappaport. A shape reconstruction method for electromagnetic tomography using adjoint fields and level sets. *Inverse Problems*, 16, 2000. 1119-1156.
[20] H. W. Engl, M. Hanke, and A. Neubauer. *Regularization of Inverse Problems*. Kluwer, Dordrecht, 1996.
[21] A. Fichtner. *Full Seismic Waveform Modeling and Inversion*. Springer, 2011.
[22] A. Friedlander, J. Martinez, B. Molina, and M. Raydan. Gradient method with retard and generalizations. *SIAM J. Num. Anal.*, 36:275–289, 1999.
[23] H. Gao, S. Osher, and H. Zhao. Quantitative photoacoustic tomography. In *Mathematical Modeling in Biomedical Imaging II*, pages 131–158. Springer, 2012.
[24] A. Greenbaum. *Iterative Methods for Solving Linear Systems*. SIAM, 1997.

[25] E. Haber, U. Ascher, and D. Oldenburg. Inversion of 3D electromagnetic data in frequency and time domain using an inexact all-at-once approach. *Geophysics*, 69:1216–1228, 2004.
[26] E. Haber, M. Chung, and F. Herrmann. An effective method for parameter estimation with PDE constraints with multiple right-hand sides. *SIAM J. Optimization*, 22:739–757, 2012.
[27] E. Haber, S. Heldmann, and U. Ascher. Adaptive finite volume method for distributed non-smooth parameter identification. *Inverse Problems*, 23:1659–1676, 2007.
[28] E. Hairer, C. Lubich, and G. Wanner. *Geometric Numerical Integration*. Springer, 2002.
[29] E. Hairer, S. Norsett, and G. Wanner. *Solving Ordinary Differential Equations I*. Springer, 1993.
[30] E. Hairer and G. Wanner. *Solving Ordinary Differential Equations II*. Springer, 1996.
[31] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning*. Springer, 2001.
[32] M. Heath. *Scientific Computing, An Introductory Survey*. McGraw-Hill, 2002. 2nd Ed.
[33] F. Herrmann, Y. Erlangga, and T. Lin. Compressive simultaneous full-waveform simulation. *Geophysics*, 74:A35, 2009.
[34] Desmond J. Higham. Global error versus tolerance for explicit runge-kutta methods. *IMA J. Numer. Anal*, 11:457–480, 1991.
[35] J. Holodnak and I. Ipsen. Randomized approximation of the gram matrix: Exact computation and probabilistic bounds. *SIAM J. Matrix Anal. Applic.*, 36:110–137, 2015.
[36] M.F. Hutchinson. A stochastic estimator of the trace of the influence matrix for laplacian smoothing splines. *J. Comm. Stat. Simul.*, 19:433–450, 1990.
[37] I. Ipsen and T. Wentworth. The effect of coherence on sampling from matrices with orthonormal columns, and preconditioned least squares problems. *SIAM J. Matrix Anal. Applic.*, 35:1490–1520, 2014.
[38] J. Kaipo and E. Somersalo. *Statistical and Computational Inverse Problems*. Springer, 2005.
[39] B. Leimkuhler and S. Reich. *Simulating Hamiltonian Dynamics*. Cambridge, 2004.
[40] V. A. Morozov. *Methods for Solving Incorrectly Posed Problems*. Springer, 1984.
[41] G. A. Newman and D. L. Alumbaugh. Frequency-domain modelling of airborne electromagnetic responses using staggered finite differences. *Geophys. Prospecting*, 43:1021–1042, 1995.
[42] D. Oldenburg, E. Haber, and R. Shekhtman. Three dimensional inverseion of multi-source time domain electromagnetic data. *J. Geophysics*, 78:E47–E57, 2013.
[43] A. Pidlisecky, E. Haber, and R. Knight. RESINVM3D: A MATLAB 3D Resistivity Inversion Package. *Geophysics*, 72(2):H1–H10, 2007.
[44] M. Raydan. *Convergence Properties of the Barzilai and Borwein Gradient Method*. PhD thesis, Rice University, Houston, Texas, 1991.
[45] J. Rohmberg, R. Neelamani, C. Krohn, J. Krebs, M. Deffenbaugh, and J. Anderson. Efficient seismic forward modeling and acquisition using simultaneous random sources and sparsity. *Geophysics*, 75(6):WB15–WB27, 2010.
[46] F. Roosta-Khorasani and U. Ascher. Improved bounds on sample size for implicit matrix trace estimators. *Foundations of Comp. Math.*, 2014. DOI: 10.1007/s10208-014-9220-1.
[47] F. Roosta-Khorasani, G. Székely, and U. Ascher. Assessing stochastic algorithms for large scale nonlinear least squares problems using extremal probabilities of linear combinations of gamma random variables. *SIAM/ASA J. Uncertainty Quantification*, 3, 2015. DOI: 10.1137/14096311X.
[48] F. Roosta-Khorasani, K. van den Doel, and U. Ascher. Stochastic algorithms for inverse problems involving PDEs and many measurements. *SIAM J. Scient. Comput.*, 36:s3–s22, 2014.
[49] A. Shapiro, D. Dentcheva, and D. Ruszczynski. *Lectures on Stochastic Programming: Modeling and Theory*. Piladelphia: SIAM, 2009.
[50] N. C. Smith and K. Vozoff. Two dimensional DC resistivity inversion for dipole dipole data. *IEEE Trans. on geoscience and remote sensing*, GE 22:21–28, 1984.
[51] U. Trottenberg, C. Oosterlee, and A. Schuller. *Multigrid*. Academic Press, 2001.
[52] C. Vogel. *Computational methods for inverse problem*. SIAM, Philadelphia, 2002.
[53] J. Young and D. Ridzal. An application of random projection to parameter estimation in partial differential equations. *SIAM J. Scient. Comput.*, 34:A2344–A2365, 2012.
[54] Z. Yuan and H. Jiang. Quantitative photoacoustic tomography: Recovery of optical absorption coefficient maps of heterogeneous media. *Applied physics letters*, 88(23):231101–231101, 2006.