DISCRETE PROCESSES AND THEIR CONTINUOUS LIMITS

URI M. ASCHER

Abstract. The possibility that a discrete process can be fruitfully approximated by a continuous one, with the latter involving a differential system, is fascinating. Important theoretical insights, as well as significant computational efficiency gains may lie in store. A great success story in this regard are the Navier-Stokes equations, which model many phenomena in fluid flow rather well. Recent years saw many attempts to formulate more such continuous limits, and thus harvest theoretical and practical advantages, in diverse areas including mathematical biology, economics, computational optimization, image processing, game theory, and machine learning.

Caution must be applied as well, however. In fact, it is often the case that the given discrete process is richer in possibilities than its continuous differential system limit, and that a further study of the discrete process is practically rewarding. Furthermore, there are situations where the continuous limit process may provide important qualitative, but not quantitative, information about the actual discrete process. This paper considers several case studies of such continuous limits and demonstrates success as well as cause for caution. Consequences are discussed.

Key words. numerical methods, inverse problems, differential equations, optimization, regularization

AMS subject classifications. Primary: 65K05, 65L04; Secondary: 68U10

1. Introduction. The quest for continuous models that would govern in some fruitful sense families of discrete processes is age-old and continuing to be fascinating [60]. Indeed, there is much to be gained if a complicated discrete process can be adequately approximated by a continuous one in some limit sense:

• A simpler structure more amenable to analysis may be considered, adding to general phenomenological understanding.
• Efficient numerical approaches may be constructed for the discrete problem by going through the continuous one.
• Perhaps the more complicated and less elegant discrete structure may not even have to be considered.

A great success story in this regard are the Navier-Stokes equations: for many fluid flow phenomena these partial differential equations capture the observed flow well enough that we can forget about their derivation while concentrating on understanding their solutions and on approximating them numerically [23, 22]. At the same time we note, ironically, that the discrete, molecular description of fluid flow involves a huge number of particles, and that not all fluid phenomena are covered by this continuous model.

A lot of exciting work that is relevant in our context has been carried out in recent years. Here is but a partial list of areas and instances of interest:

• Artificial time regularization. We return to this below, in Sections 2, 3, and 5; see [6, 62] for a wider survey.
• One-step iterative methods posed as a first order ODE discretization. This is a special case of artificial time continuation. Indeed, any iterative algorithm of the form

\[ x_{k+1} = x_k + \alpha_k g(x_k) \]
which updates a current iterate \( x_k \) (for a nonnegative iteration counter \( k \)) using a step size \( \alpha_k > 0 \) to obtain the next approximation \( x_{k+1} \) can be viewed as a forward Euler discretization of the ordinary differential equation (ODE)

\[
\frac{dx}{dt} = g(x(t)).
\]

This differential equation is obtained from the given discrete process by letting \( \alpha_k \to 0 \), though, while in practice we may well want to keep the step size quite far from zero. Herein lies a difference between qualitative and quantitative information.

- **Two-step iterative methods posed as a second order ODE discretization.** There has been a lot of recent activity in this regard in attempts to better understand accelerated gradient descent methods for unconstrained optimization [51, 14, 56, 57, 61, 15, 63, 26]. We will not dwell further in this article on this mushrooming area of research, beyond what is exposed in Section 5. We do believe nonetheless that the present paper is relevant in this context.

- **Regularizing image processing problems using a penalty on the discretized gradient of an image \( u \) in its continuous limit \( \int_\Omega |\nabla u|^s \), with \( s = 2 \) or \( s = 1 \).** See, for instance, [20, 19, 10]. We return to this below, in Section 6.

- **Continuation methods for nonlinear equations; homotopy path, etc.** See, for instance, [25, 6] and references therein. Such methods can be viewed as artificial time integration, but again we focus on other directions here.

- **Semiconductor equations.** See, for instance, the text [46]. A significant volume of work preceded and followed this book. In our present context we observe that, unlike the case for fluid flow, more than one continuous process has been derived and the practical importance of the underlying particle process has not diminished over the years.

- **Hamilton-Jacobi and mean field in game theory.** See, for instance, [44, 45] and many following articles such as [32, 16, 31] and others. Here we do not focus on this large and relatively recent volume of work.

- **Deep learning.** The tremendous potential in machine learning and in particular deep learning (DL) techniques has riveted scientists and engineers in recent years [33]. Noting that the connection between consecutive layers in a DL network resembles a finite difference scheme, models depending on corresponding limit differential equations and algebraic multigrid methods have been proposed in order to better design such neural networks. See, for instance, [54]. Interest in two-step optimization algorithms mentioned above is also related to optimization methods in neural networks.

With all the promise and excitement of the continuous-limit models, it is also important to emphasize that care should be taken to ensure that, for the given task, all important and interesting properties of the discrete process are captured by the continuous one. Otherwise, there is the possibility of being restricted to a non-optimal path. Furthermore, it is important, and occasionally crucial, to distinguish between qualitative and quantitative relationships between the discrete and the continuous. Undoubtedly, many researchers have arrived at such a practical conclusion in specific circumstances (for instance in collective dynamics), with graduate students who are responsible for the implementation of the relevant algorithms often being the first to realize this.
Below we describe several short case studies that aim to highlight various aspects of the issues involved. We start in Section 2 with a simple example which demonstrates that viewing a discrete algorithm as a discretization of a continuous process may simply be a matter of taste, depending on what one is used to and feels comfortable with. Then in Section 3 we quickly derive a useful and well-known class of regularization methods for inverse problems following the artificial time reasoning, so here is a case where the continuous limit offers a useful and somewhat different perspective. In Section 4 we then turn the heat up a notch and discuss the potential, if rare, instability of the classical implicit midpoint method for stiff ODEs. Here we see an instance where, on one hand, the meaningful continuous limit is not what comes to mind without thought, and on the other hand, the correct limit leads to a rigorous characterization of unstable scenarios. In the longer Section 5 we develop a scenario where a discretization of the simplest heat equation with constant time step size exhibits the usual stability limit on that step, arising from a semi-discretization in space; but with the freedom of choosing a variable time step size such a stability connection to spatial discretization does not necessarily hold. We then connect this scenario to the gradient descent method of Section 2 and demonstrate that the lagged steepest descent method [13, 28] produces such large step sizes. In Section 6 we give a quick overview of our past efforts to apply a Tikhonov-type regularization involving differential equations, as mentioned above, to image and surface processing applications. These efforts, and many others, have only been partially successful, and we explain why. Section 7 seals the paper by offering a few general conclusions.

In each of our case studies the discrete process has a step size parameter $h$, and we examine a limit process for it. If the step size depends on the stage or iteration $k$, denoted $h_k$, then $h = \max_k h_k$.

**Remark.** The author’s curiosity about the subject of the present work was first aroused before 1980. We were constructing the general-purpose software package COLSYS [3] for boundary value ODE systems [7]. Such discretized nonlinear problems often lead to systems of nonlinear algebraic equations that are difficult to solve, and we used the damped Newton method with an automatic choice for the damping, or step size, parameter. The ideal choice for this parameter is $\alpha = 1$, where a quadratic convergence rate is achieved for Newton’s method. However, often in applications the Newton direction was poor, and a rather small step size was required to obtain decrease is some monitor objective function. The essential difficulty was that upon moving the iterate by such a small step size the next Newton direction was also poor, causing a repeated similar difficulty! We ended up providing means in COLSYS to switch to other methods in such situations. But the lesson relevant here was that, in practice, letting the step size go to 0 was not desirable at all, as this limit process is not “practically smooth”.

2. **Gradient descent stability bound.** Consider the problem of finding the minimum of a convex $C^1$ function of $n$ variables $f(x)$, and denote

$$x^* = \arg\min_{x \in \mathbb{R}^n} f(x).$$

The gradient descent iterative method evaluates at the current iterate $x_k$ the gradient $\nabla f(x_k)$ and sets the next iterate as

$$x_{k+1} = x_k - \alpha_k \nabla f(x_k),$$

3
where $\alpha_k$ is the (positive) step size and $k$ is the iteration counter, $k = 0, 1, \ldots$ See, e.g., [49].

Let us next restrict attention to the quadratic case

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

(2.3)

where $A$ is a given $n \times n$ real symmetric positive definite (SPD) matrix and $b$ is a given real inhomogeneity vector. Here grad $f(x) = A x - b$, and defining (for historical reasons) the residual $r = -\text{grad} f(x) = A (x^* - x)$ we get the iteration

$$x_{k+1} = x_k + \alpha_k r_k.$$  

(2.4)

Next we fix the step size $\alpha_k = \alpha$ (called “learning rate” in DL parlance) and ask, what is the upper stability limit on $\alpha$?

There are two methods to answer this simple question:

1. Observe that for the error $e_k = x_k - x^*$ we have the recursion

$$e_{k+1} = (I - \alpha A)e_k,$$

so $\alpha$ is restricted by $\|I - \alpha A\|_2 \leq 1$. Since $A$ can be diagonalized by an orthogonal similarity transformation, this bound translates to $1 - \alpha \max \lambda_i \geq -1$, where $\lambda_i > 0$ are the eigenvalues of $A$ arranged in decreasing order. This leads to the stability bound

$$\alpha \leq 2/\lambda_1.$$  

(2.5)

2. Another way to see this is by introducing the limit ODE $\frac{dx}{dt} = r(x(t))$, where $t \geq 0$ is artificial time. Discretizing this ODE using the forward Euler method [8] clearly gives the gradient descent method (2.4), where the time step size is $h_k = \alpha_k$. But now, we know the absolute stability bound for forward Euler! It is the bound (2.5).

Conclusion. Which method (for arriving at the same result) is better? This depends on one’s “comfort zone”, or what one is used to. In fact, the second method is “obvious” if you are used to numerical ODEs; but it is the unnecessarily longer route otherwise.

3. Regularizing ill-posed problems. In our second case study, let us consider for simplicity the linear problem

$$A x = b,$$

(3.1)

where $A$ and $b$ are as described in Section 2, but now $A$ is also large, sparse and ill-conditioned. In terms of the SVD [4]

$$A = U \Sigma U^T,$$

(3.2a)

where $U$ is an orthogonal matrix and $\Sigma$ is diagonal, containing the singular values $s_i$ in decreasing order on its main diagonal, the solution is strictly given by $x = U y$, where the components of the vector $y$ are

$$y_i = s_i^{-1} (U^T b)_i \quad i = 1, 2, \ldots, n.$$  

(3.2b)
But if $s_{i_0} \geq 0$ is extremely small (representing a perturbation of 0 due to noise, say) for some $i_0 \leq n$, then we wish to avoid (3.2b) for all $i \geq i_0$. The solution process then has to be regularized somehow.

The truncated SVD is one possible method, where we set $y_i = 0$ for all $i \geq i_0$. But the SVD transformation involves large, full matrices, and we want to take advantage of the sparsity of $A$. An appropriate Tikhonov regularization is possible, see, e.g., [29]; we will not describe this further here.

An alternative to Tikhonov’s method is to consider the artificial time formulation from Section 2, written as

$$\frac{dx}{dt} = b - Ax. \quad (3.3)$$

This well-known method has been called exponential filtering in [18, 17]. For the artificial time ODE, at time $t$ the effect of the reciprocal singular value $s_i^{-1}$ is replaced by $\omega(s^2_i)s_i^{-1}$, where

$$\omega(s) = 1 - e^{-ts}. \quad (3.4)$$

Thus, the effect of small singular values gets dampened while large ones remain almost intact for an appropriate finite time instance $t$. This artificial finite time serves as the regularization parameter [6], gauging the amount by which inverse singular values are dampened. Next, we can use forward Euler, or some other time discretization hopefully with large step sizes $h_k$, to roughly integrate the ODE.

The resulting filter function, as it turns out, behaves quite well and not so differently from the Tikhonov filter with regularization weight $\beta$ given by

$$\omega_T(s) = \frac{s}{s + \beta}. \quad (3.5)$$

For the choice of artificial time $t_{\beta} = \frac{1}{2\beta}$ the curves in Figure 3.1 are quite similar.

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{fig3.png}
\caption{Exponential filter $\omega(s) = 1 - e^{-ts}$ and Tikhonov filter $\omega_T(s) = \frac{s}{s + \beta}$ for $t_{\beta} = 1/2$.}
\end{figure}

Some relevant questions, discussed in [6], are:
1. Using an explicit discretization method, can we take large steps even if the ODE is stiff?

2. Is the resulting regularization effective? (Note that we can view it as a continuation method: we want to get to $t = t_\beta$ quickly, without approximating the ODE trajectory accurately.)

**Conclusion.** We will not dwell on this further, since this case study is documented in a similar context elsewhere. But let us observe that here the artificial ODE yields a practically useful regularizer! At the same time, the task of approximating the ODE (3.4) should not be taken too zealously: in fact, using a higher order method such as explicit two-stage Runge-Kutta [8] does not prove more useful than forward Euler for the given purpose. Moreover, this entire regularization approach does have competition: for instance, our own favourite has often been a subspace regularization method, applying just a few conjugate gradient iterations to a singular linearized system [27, 53].

4. **Stability of the midpoint method.** In our third case study we start with a given ODE system (so the independent variable $t$ is not artificial) and consider another, implicit discretization. Let us denote $\dot{u} = \frac{du}{dt}$. For the ODE system

$$\dot{u} = f(t, u), \quad 0 \leq t \leq T,$$

(4.1)

the well-known implicit midpoint method reads

$$\frac{u_{k+1} - u_k}{h} = f \left( \frac{t_{k+1} + t_k}{2}, \frac{u_{k+1} + u_k}{2} \right).$$

(4.2)

Here $h = h_k$ is a step size from time $t_k$ (where for an initial value problem $u_k$ is known) to time $t_{k+1}$ (where $u_{k+1}$ is unknown).

This method is implicit, second order accurate, A-stable, AN-stable (meaning it is stable for $f = \lambda(t)u$ for any $\Re(\lambda) \leq 0$), algebraically stable, symplectic for Hamiltonian systems, symmetric (reversible), and it conserves quadratic invariants. Collocation at Gaussian points generalizes midpoint to higher order methods and ODE systems [7]. The method has been applied in the context of boundary value ODEs [7], geometric integration [35] and stiff initial value ODEs (IVODEs). We continue with the latter.

Despite all its good properties (and perhaps because of some of them), the implicit midpoint method is not always stable for well-posed, stiff IVODE systems! Our task is next to analyze this situation. To recall, a stiff ODE typically has more than one time scale, and we aim to employ a time step $h$ that is commensurate with the slow scale but not with the fast one [36, 11, 21]. In such a constellation, though, the _continuous limit of the discrete structure_ (4.2) _is not the original problem_ (4.1). This is so, because to get from (4.2) to (4.1) we have to let $h \to 0$, and once we do that the problem is no longer stiff in the sense that the fast scale is eventually also resolved by the small step size. Instead, we have to resort to numerical singular perturbation techniques.

To study this further, consider the linear non-autonomous ODE system

$$\dot{u} = \omega A(t)u,$$

(4.3)

where $\omega$ is a parameter that is allowed to grow large (so we must consider $\omega \gg 1$), and $\|A\| = \mathcal{O}(1)$ and nonsingular. It is possible to generalize (4.3) to the case where
$A = A(t; \omega)$. In (4.3) we are essentially studying a linearized version of (4.1) that governs the propagation of an error. Assume that the IVODE for (4.3) is well-posed (or “stable”). The midpoint method reads

$$\frac{u_{k+1} - u_k}{h} = \omega A \left( \frac{t_{k+1} + t_k}{2} \right) \frac{u_{k+1} + u_k}{2},$$

(4.4)

but we cannot conclude much when

$$\gamma = \frac{1}{4} \omega h^2$$

(4.5)

is fixed and not small, because we can’t take the limit of $h \to 0$ without also changing $\omega$ and thus the ODE (4.3), as explained above. Specifically, what we are looking at instead is the limit process of $(h \to 0, \omega \to \infty)$, such that $\gamma$ stays constant (e.g., $\gamma = 1$).

We next introduce an old but not necessarily well-known trick [43]. For analysis purposes, let us define for each integer $k$

$$v_k = (-1)^k u_k.$$  

(4.6)

Obviously, the boundedness properties of the sequences $\{v_k\}$ and $\{u_k\}$ are the same:

$$|u_k| = |v_k| \forall k.$$  

Substituting (4.6) in (4.4) and cancelling out $(-1)^k$ we obtain

$$\frac{v_{k+1} + v_k}{h} = \omega A \left( \frac{t_{k+1} + t_k}{2} \right) \frac{v_{k+1} - v_k}{2}.$$  

Multiplying throughout by $2/h$ then yields an approximation for $\dot{v}$ at the right hand side. Multiplying further by $(\omega A)^{-1}$ we can rewrite this as

$$\frac{v_{k+1} - v_k}{h} = \frac{1}{\gamma} A^{-1} \left( \frac{t_{k+1} + t_k}{2} \right) \frac{v_{k+1} + v_k}{2}.$$  

(4.7)

Now, for (4.7) with $\gamma$ fixed we can finally take the limit $h \to 0$, obtaining the ghost ODE

$$\dot{v} = \gamma^{-1} A^{-1}(t)v.$$  

(4.8)

The stability of the midpoint method therefore depends on the stability of the ghost ODE (4.8), not on that of the given ODE (4.3).

The exercise to design a stable IVODE problem (4.3) such that the IVODE for (4.8) is unstable is solved in [2] for the limit case $1/\omega = 0$. It can be extended to a large but finite $\omega$. Then the computed solution using the implicit midpoint method applied to a well-posed problem can be made to blow up; see [2].

The above analysis can be summarized as follows:

Theorem 4.1. Assume that the given IVODE for (4.3) is well-posed, and consider the midpoint discretization (4.4) with $u_0 = u(0)$ for $k = 0, 1, \ldots, N - 1$ and $Nh = T$.  

Further, although it is not strictly a matter of stability, we urge the interested reader to see Example 2.1 in [9], where the analysis trick (4.6) is used to show that the symplectic midpoint method could produce consistently wrong results for highly oscillatory non-autonomous Hamiltonian systems.
Then this method is stable along the ray \((h \to 0, \, \omega \to \infty)\) such that \(\gamma\) defined in (4.5) is held fixed, iff the IVODE for the ghost ODE (4.8) is well-posed.

If \(A\) is a constant matrix, then the stability of the midpoint method is guaranteed. But in the more general case it is not. Fortunately, such midpoint instability is rather rare in practice, and this is important especially for boundary value ODEs, where general-purpose codes typically implement symmetric schemes.

**Conclusion.** In this case study we have learned two lessons. The first is that just because a differential equation looks like a plausible continuous limit does not mean that it is the correct continuous limit for a discrete process in the context of a given task. The second lesson is that with some extra care and ingenuity it may be possible to use a continuous limit to prove interesting theoretical results that are of practical importance.

5. **Forward Euler for the heat equation and chaotic descent.** In this case study we obtain some unexpected and perhaps counter-intuitive results by considering time-stepping with highly variable step sizes.

Consider the simple heat equation on a unit square in space and time

\[
\frac{\partial u}{\partial t} = \Delta u + b, \quad 0 < x, y < 1, \quad t \geq 0
\]

with homogeneous boundary conditions (BC) and initial condition \(u(0, x, y) = u_0(x, y)\) that satisfies the BC. Here \(\Delta u = u_{xx} + u_{yy}\) is the Laplacian, and \(b > 0\) is a constant for simplicity. Integrating in time to steady state, we obtain the model Poisson problem via a continuation method in the time variable

\[-\Delta u = b, \quad 0 < x, y < 1\]

subject to the same BC.\(^2\)

A forward Euler discretization of (5.1) in time gives the recursion

\[
u_{k+1} = u_k + h_k \Delta u_k + h_k b, \quad k = 0, 1, \ldots
\]

This is clearly unstable for any series of positive time step sizes \(h_k\). The unconditionally unstable semi-discretization (5.3) is then our continuous model.

Next, we discretize the partial differential equation (PDE) (5.1) in space first, with constant spacing \(\xi = 1/\sqrt{n}\) in both \(x\) and \(y\) directions. Reshaping the 2D array of unknowns at spatial mesh points into a vector \(\mathbf{u}\) of length \(n\) we obtain an ODE system

\[
\frac{d\mathbf{u}}{dt} = -A\mathbf{u} + \mathbf{b},
\]

where the \(n \times n\) matrix \(A\) is SPD. Using straightforward centred differences for the Laplacian, \(A\) has 5 nonconsecutive nonzero diagonals (see, e.g., Section 7.1 of [4]).

Our discrete model is obtained upon applying the same forward Euler method as for the continuous model to (5.4), obtaining

\[
\mathbf{u}_{k+1} = \mathbf{u}_k + h_k (\mathbf{b} - A\mathbf{u}_k).
\]

\(^2\)A time-honoured continuation method for finding an approximate solution for a tough nonlinear elliptic PDE \(\psi(u) = 0\) is to numerically integrate the parabolic PDE \(u_t = \psi(u)\), starting from an initial guess \(u_0\), until a suitable error tolerance is satisfied. For our purpose here we consider the simplest linear version of this.
This recursion is the same as (2.4) in Section 2. Evaluating eigenvalues, the stability restriction (2.5) translates here to the condition

\[ h \leq \frac{1}{4} \xi^2, \text{ where } h = \max_k h_k. \]  

(5.6)

So, when we let \( \xi \to 0 \) the stability region for \( h \) shrinks like \( \xi^2 \). The unconditional instability as above is effectively reached in the continuous limit for (5.3). For a uniform step size, then, the continuous model governs the discrete process well.

But next we ask, is this shrinking stability limit on the maximum step size necessary even if we allow \( h_k \) to vary? The answer turns out to be negative! Indeed, consider a sequence of several small step sizes \( h_k \) effectively reducing the amplitude of high eigenvalue modes of the residual, followed by a large step reducing low mode amplitudes while inflaming the high modes to some extent. The latter are reduced again using further small steps. The largest step size \( h \) depends on the low modes rather than the high modes, and as such it can be independent of \( \xi \). This possibility is not available for the (semi-) continuous limit scheme, only the fully discrete one!

**Remark.** Before proceeding to see an actual method which produces large step sizes, let us note that it is \( h \) and not \( \xi \) which is in our focus of interest here. Indeed, the entire PDE setup can be replaced by a time-dependent linear ODE system discretized as in (5.5), where the constant SPD matrix \( A \) has eigenvalues that grow larger and larger and the question is whether \( h \) must then tend to zero for convergence to steady state.

5.1. Faster gradient descent. To actually obtain such a sequence of time steps as described above, we use the interpretation of the forward Euler discretization as that of gradient descent, as in Section 2. Since we are switching from numerical PDEs to optimization, let us change notation slightly to conform to different standards by letting \( x \leftarrow u \), \( \alpha_k \leftarrow h_k \). Then our forward Euler for the discretized heat equation becomes gradient descent for the quadratic objective function (2.3)

\[ x_{k+1} = x_k + \alpha_k r_k, \quad r_k = b - Ax_k. \]

We next concentrate on choosing the step size \( \alpha_k = h_k \).

The steepest descent (SD) choice is

\[ \alpha_k^{SD} = \frac{r_k^T r_k}{r_k^T A r_k} = \frac{(r_k, r_k)}{(r_k, A r_k)}. \]  

(5.7)

This is the “greedy choice”, obtained upon performing exact line search. It is known to perform well for the first few steps but to become slower later on [1]. Asymptotically SD performs as well as the best constant step choice, for which the bound (5.6) holds.

The lagged steepest descent (LSD) step size [13] uses the same expression evaluated at the previous iterate:

\[ \alpha_k^{LSD} = \frac{r_{k-1}^T r_{k-1}}{r_{k-1}^T A r_{k-1}} = \frac{(r_{k-1}, r_{k-1})}{(r_{k-1}, A r_{k-1})}. \]  

(5.8)

This then is a two-step method, using information about both \( x_k \) and \( x_{k-1} \) to define \( x_{k+1} \).

\[ \text{In [28] we found experimentally that the overall best faster gradient descent methods are the two-step ones.} \]
Example. Returning to our heat equation with $b = 1$, we apply the scheme (5.5) with $h_k = \alpha_k^{SD}$ to advance to steady state, stopping when $\|r_k\| \leq 10^{-6}\|r_0\|$. The maximum step size $h$ is recorded in Table 5.1 for a decreasing sequence of spatial steps $\xi$. Evidently, the maximum step size $h$ does not shrink as a function of $\xi$ in

| $\xi$   | $2^{-3}$ | $2^{-6}$ | $2^{-1}$ | $2^{-8}$ |
|--------|----------|----------|----------|----------|
| $h$    | .05      | .039     | .043     | .035     |

*Table 5.1: Maximum step size $h$ for the heat-to-Poisson process as a function of spatial step $\xi$."

this example.

Continuing our discussion of the LSD method, it is obvious from Table 5.1 that the maximum step size $h$ does not satisfy the bound (5.6). In other words, some step sizes $h_k = \alpha_k$ disobey the fixed-step stability limit which relates to the continuous PDE. This causes a chaotic effect in the resulting method which nonetheless converges [28].

There are many other faster gradient descent variants; our favourite among those is the one where the SD step (5.7) is simply updated only at every second iteration [52]. In experiments its efficiency is comparable to that of LSD.

Figure 5.1 displays the step sizes observed in the LSD iteration. Notice how much larger the largest step sizes are from the maximum constant step size.

![Figure 5.1: Gradient descent with step sizes by (5.8) for the discretized heat equation with $n = 63^2$. The calculated step sizes are displayed vs iteration counter $k$. The stability limit for a constant step size gives the straight blue line.](image)

The actual residual norm $\|r_k\|_2$ and objective function error $f(x_k) - f(x^*)$ with $f$ given by (2.3) are given in Figures 5.2 and 5.3, respectively. In these figures we observe a rather non-monotone convergence, and yet the overall convergence rate does not appear to be slow.

As mentioned in Section 1, there has been a lot of recent interest in two-step optimization methods that accelerate gradient descent. We will not get into related
ODE formulations, and instead mention the most popular of these. The celebrated Nesterov’s method [47] for the problem (2.1), i.e., the unconstrained minimization of a function $f(x)$, is given by

$$y_{k+1} = x_k + \beta_k(x_k - x_{k-1}), \quad (5.9a)$$

$$x_{k+1} = y_{k+1} - \alpha_k \text{grad } f(y_{k+1}). \quad (5.9b)$$
Note that, unlike LSD, the gradient is not evaluated at \( x_k \), but rather at its modification \( y_{k+1} \). Hence, the search direction is not the residual \( r_k \).

It can be shown that under suitable conditions this method satisfies
\[
f(x_k) - f(x^*) = \mathcal{O}(k^{-2}).
\] (5.10)

Furthermore, there is an oracle that says that such a rate is optimal. Note that SD and constant-step gradient descent do not achieve (5.10) [48]. It is therefore interesting to compare the method (5.9) to the faster gradient descent LSD for the case where \( f \) is convex quadratic as in (2.3).

However, the non-monotonic convergence displayed in Figures 5.2 and 5.3 makes such a comparison difficult. Fortunately, these errors can be “monotonized” by ditching the heat equation interpretation and evaluating, following the \( k \)th iteration,
\[
\ell_k = \arg \min_{1 \leq l \leq k} \| \nabla f(x_l) \|.
\] (5.11a)

Note that storage and CPU time can be saved by evaluating this at each step \( k > 0 \) as
\[
\ell_k = \begin{cases} 
  k & \text{if } \| \nabla f(x_k) \| \leq \| \nabla f(x_{\ell_{k-1}}) \| \\
  \ell_{k-1} & \text{otherwise}
\end{cases}.
\] (5.11b)

We can then report \( x_{\ell_k} \) at any real time, if required, as the “best” approximate solution at the end of iteration \( k \) (although we don’t use it for the evaluation of the next iteration), because this represents what we know at the end of step \( k \) better than \( x_k \) does, in general. So to assess the iterative process we may record also
\[
E_{\text{grad}}_k = \| \nabla f(x_{\ell_k}) \|, \quad \text{and} \quad E_f_k = f(x_{\ell_k}) - f(x^*),
\] (5.11c)
the latter still involving a generally unknown quantity.

The important point to remember is that all we need to report in practice is the computed solution upon termination. So, if our termination criterion is \( \| \nabla f(x_k) \| \leq \text{tol} \) for some error tolerance \( \text{tol} \), then upon achieving this we stop.

For the quadratic optimization problem we use the notation
\[
E_r_k = \| r_{\ell_k} \|.
\] (5.12)

These residual error measures are by definition monotonically non-increasing, and calculating them at the end of each iteration is straightforward. Furthermore, stopping the iteration at the first \( k \) such that \( \| r_k \| \leq \text{tol} \| b \| \) is the same as using \( E_r_k \leq \text{tol} \| b \| \) for termination.

For the method (5.9) there is the question of determining appropriate step sizes \( \beta_k \) and \( \alpha_k \). If we keep both step sizes constant, then the iteration is stationary and standard techniques (involving the largest and perhaps also smallest eigenvalues of the Hessian) apply. For the calculations in Figures 5.4 and 5.5 we fixed the momentum parameter \( \beta = 0.95 \) (after some trial-and-error) and calculated \( \alpha_k \) by the SD formula (5.7) with \( r_k \) replaced by \( b - Ay_{k+1} \).

In the present convex quadratic optimization context, all four methods depicted in these figures are Krylov-subspace methods generating iterates in the same subspace at each \( k \). So, the superiority of CG is a foregone conclusion, as it is optimal
Fig. 5.4. Convergence behaviour of monotonized LSD (LSDm) as well as SD, conjugate gradient (CG) and Nesterov’s (Nes) for the model Poisson problem with $n = 63^2$. The errors $\|r_k\|$ are displayed as a function of iteration counter $k$.

on each such subspace (see, e.g., [34]). More interestingly, note the closeness of the LSD method and the accelerated gradient descent method (5.9), both being far better than SD and not much worse than CG. This suggests that the monotonized lagged steepest descent method also satisfies the “optimal” estimate (5.10).

Conclusion. This longer case study, which involves some new work, demonstrates a situation where the continuous differential equation limit (5.3) misses out on some interesting and unusual action that occurs at the discrete process level.

6. Gradient-based penalties in image processing. In this case study we consider image and surface processing problems, where the given problem is entirely discrete. Instances include image and surface denoising, deblurring and inpainting, data consolidation, morphing, and much more. We can often view this as an instance of data fitting: the given data $d$ relates to an image or object surface (e.g., it can be a point cloud), the required operation such as denoising is described by a forward operator $\phi$, and we seek another image or surface $u$ in 2D or 3D, respectively, such that $\phi(u)$ fits the data $d$ to a desired extent. Often, however, a differential term is added in an attempt to regularize and control the desired process, and the resulting differential problem is then subsequently discretized in order to compute a solution.

Note that we are deliberately using the notation $u$ both for a discrete image/object and for this image/object interpolated in such a way that its gradient in two or three space variables, denoted $\nabla u$, may be considered and differentiated. It is then discretized (or restricted) back to the space where the data $d$ lives. The step size $h$ is a parameter of such a discretization, say the distance between neighbouring image data locations. Here we examine the utility of such a continuous model, which the computer never “sees”.
To be concrete, let us consider the problem

\[
\min_u f(u), \quad \text{where } f(u) = \|\phi(u) - d\|_2^2 + \beta \int_\Omega |\nabla u|^s,
\]

which we will call the paradigm, for short. Here \(\Omega\) is the domain of the image or the 3D object and \(\beta > 0\) is a fixed, noise-dependent regularization weight as in Figure 3.1. Indeed, (6.1) is in the typical form of Tikhonov regularization for an inverse problem [59, 29, 42]. However, we stress that our concern here is with problems where \(\phi\) is not a discretization of a continuous structure, and the only source of differential terms, whether explicit or inverted, is the regularization term. As such there is some cause for minor mathematical discomfort, because the leading (or, highest order) term of the PDE in the boundary value problem that arises in the variational form comes from the regularization rather than the data fitting term in (6.1).

There is a huge volume of literature on such methods; see, for instance, [55, 20, 50, 19], to name but a few. The parameter \(s\) typically equals 2, representing significant smoothing (diffusion), or 1 for total variation (TV), which typically yields piecewise smoothing and is also referred to as anisotropic diffusion. Other values such as \(0 < s < 1\) are not our concern here. The Euler-Lagrange equations for (6.1) give a boundary value PDE, and the problem actually solved is a discretization of such a PDE. The gradient in (6.1) can be replaced by a higher order differential operator, which is again not the focus here. If \(s = 1\), then some further regularization of the regularizer is required, and in [5] we proposed a method for controlling a Huber switch between \(s = 1\) and \(s = 2\) for this purpose.

There are lots of situations where the paradigm is the thing to consider! Indeed, it introduces additional information which the discrete \(\phi(u)\) as such does not have...
that relates the values of the \( u \)-unknowns to their neighbouring values. This in turn potentially enables a significantly simpler optimization process for \( f(u) \).

However, already around the turn of the 21st century researchers found experimentally that the differential term in (6.1) can introduce side effects that pollute the visual quality of the obtained result. A “social bifurcation” has formed, where researchers and practitioners in computer graphics and image processing have turned more towards purely data driven methods, while mathematicians and numerical analysts have continued to explore the paradigm and other such methods due to their more coherent structure that in turn enables development of better and more complete theory.

Around the year 2005, armed with the code from [5], we started exploring ways to introduce the paradigm for various image processing applications in computer graphics. Here the visual quality of the results is what matters, and no theory can compensate for lack in this regard. We considered surface triangle mesh denoising in applications that involve intrinsic texture and sharp corners [37, 38]. Together with computer graphics and vision experts we then considered consolidation of unorganized point clouds to allow quality surface reconstruction [39], point-resampling in the presence of edges [40], and image tele-registration and structure-driven completion [41]. Unfortunately, while working on these projects our version of the paradigm was consistently beaten by other, discrete approaches, and there is no continuous model left in the final, published version of any of [37, 38, 39, 40, 41]!

Let us add a few more details regarding the research just mentioned.

- A noisy triangle mesh representing a surface has noisy triangle vertices, which are vectors in 3D. Several researchers have considered forms of anisotropic Laplace-Beltramy PDE for denoising surface triangle meshes. But in [37] we learned (not before failing to extend the image denoising method of [58] to surfaces) that the most effective way to denoise such a vertex is to find the normal direction to the surface there and denoise in that specific direction. To find such a normal, the standard technique is to perform a principal component analysis (PCA) on a group of neighboring vertices. This process, described in detail in [37], is necessarily geometrically local, in marked difference to the global nature of an expression such as the penalty integral in (6.1). The obtained visual results are better, the complexity of the algorithm is linear in the number of unknowns(!), and the corresponding MATLAB code has less than 50 lines. We feel that, unfortunately, one just can’t practically do all that with anisotropic Laplace-Beltramy.

- In [40] there is a construction of normals for a consolidated point cloud that represents a surface with an edge. Our algorithm is shown to perform better than applying an \( \ell_1 \) regularization to the crude PCA normals (which is related to TV).

- The tele-registration algorithm in [41] involves a step to find salient curves in an image. This is a task for TV regularization, but the algorithm of [30] proved to be more robust. Further, towards the end of the image completion procedure there the algorithm and software of [12, 24], which do not employ any direct version of the paradigm, were used for inpainting.

In [10] we have summarized the pros and cons of the continuous paradigm (6.1) vs discrete processes of data-driven reconstruction.

**Conclusion.** The essential advantage in using the paradigm is that one can often obtain a more solid theoretical backing to algorithms that are occasionally
more clearly derived based on solid principles from this global and generic point of view. The essential disadvantage, however, is that this approach leads to algorithms that could be outperformed by more brute force techniques, as described for the examples above. This is so especially if (i) the forward operator $\phi$ is simple, and (ii) the data $d$ is of high visual quality.

7. Conclusions. Recent years have seen a welcome development where knowledge and expertise that have been gained over decades in modelling and computationally solving differential equations find use in the context of discrete problem areas such as optimization, game theory, graphics and image processing. The author, who has dealt with the numerical solution of differential equations most of his adult life while being a member of a computer science department, finds special reasons to rejoice. At the same time, we advocate balance and controlled euphoria. It is crucial to examine with an open mind the utility of such approaches and specific methods, from perspectives that include applicability, relative advantage, and extensions in the context of the task at hand.

We have demonstrated in a sequence of case studies that employing differential equations in this way can be at times very useful, at times limiting, and at times just a matter of comfort. One example of such “comfort”, not mentioned hitherto, is the use of global properties of dynamical systems, such as conservation laws, symplecticity and reversibility, at the discrete level without requiring extreme closeness of discrete and continuous model solutions. We advocate to become involved in this fascinating research avenue, as we ourselves are, while exercising practical caution.

We have deliberately avoided any significant discussion of stochastic optimization and stochastic differential equation methods and models, as well as randomized algorithms. Our point is not to ignore these exciting areas, but rather to emphasize that significant issues arise more basically even in their absence.

Acknowledgment. I thank Drs Jelena Diakonikolas, Arieh Iserles, Alexander Madureira, Fred Roosta, Lars Ruthotto, Mark Schmidt and Jorge Zubelli for fruitful discussions. During several preparation stages of this paper I enjoyed the hospitality of IMPA, Rio de Janeiro.

REFERENCES

[1] 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.
[2] U. Ascher. On symmetric schemes and differential-algebraic equations. SIAM J. Scient. Comput., 10:937–949, 1989.
[3] U. Ascher, J. Christiansen, and R.D. Russell. Collocation software for boundary-value ODEs. ACM Trans. Math. Software, 7(2):209–222, 1981.
[4] U. Ascher and C. Greif. A First Course in Numerical Methods. SIAM, Philadelphia, PA, 2011.
[5] U. Ascher, E. Haber, and H. Huang. On effective methods for implicit piecewise smooth surface recovery. SIAM J. Sci. Comput., 28:339–358, 2006.
[6] U. Ascher, H. Huang, and K. van den Doel. Artificial time integration. BIT, 47:3–25, 2007.
[7] U. Ascher, R. Mattheij, and R. Russell. Numerical Solution of Boundary Value Problems for Ordinary Differential Equations. SIAM, Philadelphia, 1995.
[8] U. Ascher and L. Petzold. Computer Methods for Ordinary Differential Equations and Differential-Algebraic Equations. SIAM, Philadelphia, PA, 1998.
[9] U. Ascher and S. Reich. The midpoint scheme and variants for hamiltonian systems: advantages and pitfalls. SIAM J. Scient. Comput., 21:1045–1065, 1999.
[10] U.M. Ascher and H. Huang. Numerical analysis in visual computing: What we can learn from each other. *Vietnam J. Math.*, pages DOI: 10.1007/s10013–018–0299–6, 2018.

[11] D. Baraff and A. Witkin. Large steps in cloth simulation. In *SIGGRAPH*, pages 43–54. ACM, 1998.

[12] C. Barnes, E. Shechtman, A. Finkelstein, and D. Goldman. Patchmatch: a randomized correspondence algorithm for structural image editing. *ACM Transactions on Graphics (SIGGRAPH)*, 27(3):67:1–67:10, 2009.

[13] J. Barzilai and J. Borwein. Two point step size gradient methods. *IMA J. Num. Anal.*, 8:141–148, 1988.

[14] D. Calvetti and L. Reichel. Lanczos-based exponential filtering for discrete ill-posed problems. *Numer. Algorithms*, 29:45–65, 2002.

[15] M. Betancourt, M. Jordan, and A. Wilson. On symplectic optimization. In *IEEE Conference on Decision and Control*, volume 52, pages 3128–3133, 2003.

[16] D. Calvetti and L. Reichel. Lanczos-based exponential filtering for discrete ill-posed problems. *Numer. Algorithms*, 29(4):671–6710, 2009.

[17] K. van den Doel and U. Ascher. The chaotic nature of faster gradient descent methods. *J. Scient. Comput.*, 51:560–581, 2011.

[18] H. W. Engl, M. Hanke, and A. Neubauer. *Regularization of Inverse Problems*. Kluwer, 1996.

[19] I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*. MIT Press, 2016. http://www.deeplearningbook.org.

[20] A. Greenbaum. *Iterative Methods for Solving Linear Systems*. SIAM, Philadelphia, PA, 1997.

[21] H. Huang and U. Ascher. Surface mesh smoothing, regularization and feature detection. *SIAM J. Scient. Comput.*, 31:74–93, 2008.

[22] H. Huang, D. Li, R. Zhang, U. Ascher, and D. Cohen-Or. Consolidation of unorganized point clouds for surface reconstruction. *ACM TOG (SIGGRAPH Asia)*, 29(5), 2009.

[23] H. Huang, S. Wu, M. Gong, D. Cohen-Or, U. Ascher, and H. Zhang. Edge-aware point set resampling. *ACM TOG*, 32(1), 2013.
[41] H Huang, K Yin, M Gong, D Lischinski, D Cohen-Or, UM Ascher, and B Chen. “mind the gap”: tele-registration for structure-driven image completion. *ACM Transactions on Graphics*, 32(6):174:1–174:10, 2013.

[42] J. Kaipo and E. Somersalo. *Statistical and Computational Inverse Problems*. Springer, 2005.

[43] H.-O. Kreiss. Centered difference approximation to singular systems of odes. *Symposia Mathematica X*, 1972. Inst. Nazionale di Alta Math.

[44] J. Lasry and P. Lions. Mean field games. *Jpn. J. Math.*, 2(1):229–260, 2007.

[45] R. Malhame M. Huang and P. Caines. Large population stochastic dynamic games: Closed-loop mckean?vlasov systems and the nash certainty equivalence principle. *Comm. Info. Sys.*, 6(3):221–252, 2006.

[46] M. Markowich, C. Ringhofer, and C. Schmeiser. *Semi-conductor equations*. Springer, 1990.

[47] Y. Nesterov. A method of solving a convex programming problem with convergence rate o(1/k²). In *Doklady Akademisi Nauk*, pages 543–547, 1983.

[48] Y. Nesterov. Lectures on convex optimization. Springer, 2018.

[49] J. Nocedal and S. Wright. *Numerical Optimization*. New York: Springer, 1999.

[50] S. Osher and R. Fedkiw. *Level Set Methods and Dynamic Implicit Surfaces*. Springer, 2003.

[51] B. T. Polyak. Some methods of speeding up the convergence of iteration methods. *USSR Comput. Math. & Math. Phys.*, 4:1–17, 1964.

[52] M. Raydan and B. Svaiter. Relaxed steepest descent and Cauchy-Barzilai-Borwein method. *Comp. Optimization Applic.*, 21:155–167, 2002.

[53] F. Roosta-Khorasani, K. van den Doel, and U. Ascher. Stochastic algorithms for inverse problems involving PDEs and many measurements. *SIAM J. Scient. Comput.*, 2014.

[54] L. Ruthotto and E. Haber. Deep neural networks motivated by partial differential equations. *J. Mathematical Imaging and Vision*, 2019.

[55] G. Sapiro. *Geometric Partial Differential Equations and Image Analysis*. Cambridge, 2001.

[56] W. Su, S. Boyd, and E. Candes. A differential equation for modelling nesterov’s accelerated gradient method. *Advances in Neural Information Processing Systems (NIPS)*, 27, 2014.

[57] W. Su, S. Boyd, and E. Candes. A differential equation for modelling nesterov’s accelerated gradient method: Theory and insights. *J. Machine Learning Research*, 17(153):1–43, 2016.

[58] E. Tadmor, S. Nezzar, and L. Vese. A multiscale image representation using hierarchical (BV,L²) decompositions. *SIAM J. Multiscale Model. Simul.*, 2:554–579, 2004.

[59] A. N. Tikhonov and V. Ya. Arsenin. *Methods for Solving Ill-posed Problems*. John Wiley and Sons, Inc., 1977.

[60] G. Wanner. Kepler, Newton and numerical analysis. *Acta Numerica*, 19:561–598, 2010.

[61] A. Wibisono, A. Wilson, and M. Jordan. A variational perspective on accelerated methods in optimization. *Proc. Nat. Academy Science*, 113(47):E7351–E7358, 2016.

[62] Willard I. Zangwill. *Nonlinear programming: a unified approach*. Prentice-Hall, Inc., Englewood Cliffs, N.J., 1969.

[63] J. Zhang, A. Mokhtari, S. Sra, and A. Jadbabai. Direct runge-kutta discretization achieves acceleration. *NeurIPS*, 2018. arXiv:1805.00521.