NUMERICAL ANALYSIS IN VISUAL COMPUTING: WHAT WE CAN LEARN FROM EACH OTHER

URI M. ASCHER* AND HUI HUANG†

Abstract. Visual computing is a wide area that includes computer graphics and image processing, where the "eye-ball norm" rules.

This paper discusses two case studies involving numerical methods and analysis applied to this wide domain. The focus is on highlighting relative strengths in those intersection areas where there are many problems of interest both to numerical analysts and to researchers in the visual computing community.

The first involves motion simulation and calibration of soft objects such as cloth, plants and skin. The governing elastodynamics PDE system, discretized in space already at the variational level using co-rotated FEM, leads to a large, expensive to assemble, dynamical system in time, where the damped motion may mask highly oscillatory stiffness. Geometric integration and exponential time differencing ideas are making their way into visual computing research these days in search for more quantitative computations that are required for applications such as control and 3D-printing.

The other case study involves some image processing problems where there is a premium for local approaches that do not necessarily use underlying PDEs. The popular paradigm of solving a data fitting problem with a penalty term involving the gradient or higher derivatives of a function approximating the sought surface is employed in the visual community much less often than in the numerical analysis community. We explain why. Concepts are demonstrated and discussed.

Key words. visual computing, numerical analysis, image processing, elastodynamics, computer graphics

1. Introduction. The purpose of this paper is to examine the relationship and domain intersections between two communities, namely, visual computing and numerical analysis/applied mathematics. Our hope, and occasional conviction, is that learning from each other can lead to developing more theoretically solid methods and techniques for visual computing tasks that, importantly, also perform competitively in practice. By *visual computing* we mean a rough union of significant subsets of computer graphics and image processing.

Consider for instance a rug spread on our living room floor. Someone has accidentally stepped on it sideways, creating a ripple as in Figure 6.8 of [2]. Now, regarding the height of the rug above the floor as an "error" that we wish to extinguish by flattening the rug, it is easy to imagine scenarios where the L_1 norm of this error over the rug domain is rather small, and yet the ripple is (irritatingly) pronounced and distinguishable. In the *eye-ball norm* this error is very different from a random perturbation of the same L_1 norm value distributed over the entire rug.

The usual types of error that numerical analysts consider allow the development of mathematically solid techniques to approximate a given process in the aim of reducing such error sufficiently. In contrast, visual computing computational results are often measured, or appreciated, by their pleasing visual effect. And yet, there are application domains where there is a mutual interest and exchange of fruitful knowhow between the communities. An important subdomain of visual computing concerns physics-based computations, e.g., in creating animations, or in computing specifications for a 3D printing job, and visual computing experts have often been shopping for numerical techniques for such problems. Likewise, there is an entire area in numerical analysis that centres on, or at least is motivated by, image processing. It is in such areas that a reader occasionally finds that one community is well ahead of the other in some quantitative sense, and a process of mining for knowledge may therefore ensue.

*Dept. of Computer Science, University of BC, Vancouver, Canada V6T1Z4 http://www.cs.ubc.ca/∼ascher/ ascher@cs.ubc.ca

†College of Computer Science and Software Engineering, Shenzhen University, China 518060 http://vcc.szu.edu.cn/∼huihuang/ huihuang@szu.edu.cn

This paper concentrates on two such areas, considered as wide-scope case studies, in an attempt to highlight the benefits that different points of view can occasionally yield.

The first (Section 2) involves the simulation and calibration of deformable objects such as a sofa, or a jacket, or a plant. Physics-based methods have been devised in the visual computing community that draw on numerical analysis approaches for solving differential equations, but often with an added twist. As the need for physical accuracy grows, say, because of targets such as motion control and 3D fabrication, the numerical methods in use may require adjustment and replacement: quantitative notions gradually replace qualitative ones. Numerical analysis techniques thus gain new relevance.

The second case study (Section 3) considers image and surface processing methods for problems such as denoising, deblurring, inpainting and image completion that use PDE-based regularization to formulate and solve the ensuing inverse problem. Our contention is that in many situations such techniques do not produce the best visual results and do not necessarily yield the most efficient algorithms. We propose some simple and general rules for deciding when to use such techniques and when to opt for better, available ones that may not rely on PDE solutions and employ more local techniques.

2. Calibration and large-step time integration in elastodynamics. Motion simulation of structures containing flexible soft objects is ubiquitous in current computer graphics and robotics research. High quality simulations obtained by using fine meshes in space and time can be very expensive. Furthermore, the model typically requires calibration, e.g., specifying Young's modulus and damping properties. These are expressed as (distributed) parameters in the elastodynamics differential equations governing the motion. For control and fabrication tasks one may require more accurate simulations than before.

The elastodynamics equations under consideration may be viewed for theoretical purposes as a system of PDEs in time and three space variables describing the deformation motion of a flexible object [13]. The rather demanding, time-varying object shape invites a finite element discretization in space, and this is routinely done already at the variational level, without forming the PDE [33, 36, 7, 6, 37]. Thus we have a tetrahydral (say) *3D mesh moving in time*. Denote the nodes (or vertices) of the mesh by $\mathbf{q}(t)$.

The semi-discretized equations for the motion of these nodes describe Newton's second law of motion. Thus, masses times accelerations equal forces ($\mathbf{v} = \dot{\mathbf{q}}$)

$$M\ddot{\mathbf{q}}(t) = \mathbf{f}_{\rm els}(\mathbf{q}) + \mathbf{f}_{\rm dmp}(\mathbf{q}, \mathbf{v}) + \mathbf{f}_{\rm ext}, \tag{2.1}$$

with the elastic and damping forces

$$\mathbf{f}_{\rm els}(\mathbf{q}) = -\frac{\partial}{\partial \mathbf{q}} W(\mathbf{q}(t)), \quad \mathbf{f}_{\rm dmp}(\mathbf{q}, \mathbf{v}) = -D\mathbf{v}(t),$$

where $W(\mathbf{q}(t))$ is the elastic potential of the corresponding model. In a linear elasticity model, this elastic potential is quadratic.

Next, we rewrite (2.1) at some time $t = t_n = t_{n-1} + h$ as $\dot{\mathbf{u}}(t) = \mathbf{b}(\mathbf{u}(t))$:

$$\begin{aligned} \dot{\mathbf{u}}(t) \equiv \begin{pmatrix} \dot{\mathbf{q}}(t) \\ \dot{\mathbf{v}}(t) \end{pmatrix} &= \begin{pmatrix} \mathbf{v} \\ M^{-1}\mathbf{f}_{\rm tot}(\mathbf{q}, \mathbf{v}) \end{pmatrix} \\ &= \begin{pmatrix} 0 & I \\ -M^{-1}K & -M^{-1}D \end{pmatrix} \begin{pmatrix} \mathbf{q}(t) \\ \mathbf{v}(t) \end{pmatrix} + \begin{pmatrix} \mathbf{0} \\ \mathbf{g}(\mathbf{u}(t)) \end{pmatrix}, \end{aligned} \tag{2.2}$$

where $K = -\frac{\partial}{\partial \mathbf{q}} \mathbf{f}_{\rm els}(\mathbf{q})$ is the tangent *stiffness matrix* at $\mathbf{q} = \mathbf{q}(t)$.

Often there is *highly oscillatory stiffness*, even though the observed motion is damped and does not vibrate rapidly. This happens when the scale of the simulation is large, and/or the material stiffens under a large deformation.

Another instance of this sort, leading to an ODE system of the form (2.1) or (2.2), is *cloth simulation*, where the modelling technique often used involves a *mass-spring* system rather than a PDE.

We therefore want to discretize this large ODE system using a time step size h that is commensurate with the damped motion rather than with the invisible high oscillation. Thus, we can't use explicit Runge-Kutta (RK) discretization, for instance. Moreover, implicit RK requires solution of a nonlinear algebraic system at each step: this can be nasty if the step size h is large. By far the most popular method in use to date is a semi-implicit (SI) method, i.e., backward Euler (BE) with only one Newton iteration at each time step starting from $\mathbf{u}_n \approx \mathbf{u}(t_n)$ [3]. However, heavy step-size dependent artificial damping is introduced: this is not easy for an artist to work with, and it affects different materials differently. Indeed, one may ask, why does it work at all in the presence of such large pointwise solution errors?

In addition to BE and SI (also in stabilized form for extremely stiff materials [35]), it is possible to consider various other alternatives, as described in [11]. See also the short video at

https://www.youtube.com/watch?v=oRGuC9GMm8w

From this video it is clear that the large error in BE and SI is not arbitrary or random. Rather, the animation reflects a less energetic but generally similar motion to that produced by other methods using the same step size. This is a rather specific way in which "physics-based" differs from "physics".

Conservative methods, including symplectic and energy conserving variants, are not expected to produce any artificial damping. Such methods have been used in the present context; see, e.g., [10, 25]. But some damping is of course useful if really large steps are to be taken. This can be achieved by introducing an artificial Rayleigh damping component into $\mathbf{f}_{\rm dmp}(\mathbf{q}, \mathbf{v})$, or by using a numerical method with a controlled amount of damping. We proceed to consider the latter using a very simple scalar ODE model.

2.1. Analysis for the simplest case. The simplest case of (2.1) involves a quadratic elastic energy with a constant symmetric positive definite (SPD) stiffness matrix K, a constant SPD matrix M and a Rayleigh damping model $\mathbf{f}_{\rm dmp}(\mathbf{q}, \mathbf{v}) = (\alpha M + \beta K)\mathbf{v}$ with $\alpha, \beta \geq 0$ given parameters. This allows for an orthogonal eigen-decomposition, so we next concentrate on an eigen-pair, for which we have a scalar ODE

$$\ddot{q} + d\dot{q} + \omega^2 q = 0, \tag{2.3a}$$

which in first order form reads

$$\begin{pmatrix} \dot{q} \\ \dot{v} \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ -\omega^2 & -d \end{pmatrix} \begin{pmatrix} q \\ v \end{pmatrix}, \tag{2.3b}$$

where we assume that $\omega > d/2$. See, e.g., [12].

Setting $q(t) = \exp(\lambda t)$ in (2.3a) we obtain the quadratic equation $\lambda^2 + d\lambda + \omega^2 = 0$, so

$$\lambda = \frac{1}{2}[-d \pm \sqrt{d^2 - 4\omega^2}]. \tag{2.4a}$$

In particular, for the undamped case $d = 0$, the modes are $\exp(\lambda t)$ with $\lambda = \pm \imath\omega$, i.e., these are oscillatory, undamped Fourier-type modes. Furthermore, by assumption $d^2 < 4\omega^2$, so we

have for both eigenvalues in (2.4a) that $Re(\lambda) = -d/2$. Hence the magnitude of the mode is

$$|\exp(\lambda t)| = \exp\left(Re(\lambda)t\right) = \exp\left(-\frac{d}{2}t\right). \tag{2.4b}$$

The BE (here $\equiv$ SI) method gives

$$\begin{pmatrix} 1 & -h \\ h\omega^2 & 1+hd \end{pmatrix} \begin{pmatrix} q_n \\ v_n \end{pmatrix} = \begin{pmatrix} q_{n-1} \\ v_{n-1} \end{pmatrix},$$

which can be written equivalently as

$$\begin{pmatrix} q_n \\ v_n \end{pmatrix} = \frac{1}{1+hd+h^2\omega^2} \begin{pmatrix} 1+hd & h \\ -h\omega^2 & 1 \end{pmatrix} \begin{pmatrix} q_{n-1} \\ v_{n-1} \end{pmatrix} \equiv T \begin{pmatrix} q_{n-1} \\ v_{n-1} \end{pmatrix}. \tag{2.5}$$

Thus, for this simple ODE, given initial values q_0, v_0,

$$\begin{pmatrix} q_n \\ v_n \end{pmatrix} = T^n \begin{pmatrix} q_0 \\ v_0 \end{pmatrix}$$

for any positive integer n. Continuing with the undamped case, we set $d = 0$ in (2.5), obtaining

$$\begin{pmatrix} q_n \\ v_n \end{pmatrix} = \frac{1}{1+h^2\omega^2} \begin{pmatrix} 1 & h \\ -h\omega^2 & 1 \end{pmatrix} \begin{pmatrix} q_{n-1} \\ v_{n-1} \end{pmatrix}. \tag{2.6}$$

The eigenvalues $\hat{\mu}$ of the matrix appearing in (2.6) satisfy

$$(1-\hat{\mu})^2 = -(h\omega)^2, \quad \text{hence } \hat{\mu} = 1 \pm \imath h\omega.$$

This gives the eigenvalues for the propagator T of (2.6) as

$$\mu = \frac{1 \pm \imath h\omega}{1+h^2\omega^2} = \frac{1}{1 \pm \imath h\omega}. \tag{2.7}$$

Obviously, for both eigenvalues, $|\mu| = 1/\sqrt{1+(h\omega)^2} < 1$ for any $h > 0$. This is therefore the spectral radius

$$\rho = \max|\mu| = 1/\sqrt{1+(h\omega)^2} \tag{2.8}$$

of the discrete operator T.

Next, comparing the damping effect of one step of BE/SI applied to the undamped ODE problem to that of a similar damped problem of the form (2.3a) with $d = d^{BE} > 0$, we equate that mode magnitudes using (2.4b) and (2.8). This gives

$$\exp\left(-\frac{d^{BE}}{2}nh\right) = \rho^n = \left(1+(h\omega)^2\right)^{-n/2}.$$

To find d^{BE} satisfying this equality, we take the natural logarithm, obtaining upon cancellation of $-n/2$ that the effect of the BE (SI) method is to introduce the artificial damping level

$$d^{BE} = \frac{1}{h}\ln(1+(h\omega)^2). \tag{2.9}$$

See Figures 1–3 in [11], noting that in (2.9) d^{BE}/ω depends only on the single variable $h\omega$. The fact that, fixing h, for large $h\omega$ the artificial damping level changes only proportionally to the logarithm may save the day.

Returning to the general system case of (2.1) with the BE/SI approximation, we cannot be so specific as in the scalar constant-coefficient case. But simulations suggest a virtual extension of the results in this section to many practical situations. There is an observed, unnatural loss of energy when large-step SI is employed, although the resulting animations do not look like garbage. Rather, they look less "energetic" and more placid. This is probably why many people are still able and happy to use this method with large time steps.

Conservative methods. A similar analysis for the implicit midpoint (IM) method readily yields an amplification matrix T for which $\rho(T) = 1$, hence $\ln(\rho(T)) = 0$, and so $d^{IM} = 0$. The method of [25] coincides with IM for the simple ODE (2.3a). Indeed, for all conservative methods, $d^{method} = 0$ and there is no artificial damping.

Generalized alpha method. Newmark methods are very popular in structural mechanics and related fields. The generalized α method is one such, and it has a knob r, $0 \le r \le 1$, to control the amount of artificial diffusion per frequency [12, 26]. It is apparently used by Disney Animation Studios, and is available as part of the finite element tools ABAQUS and Comsol Multiphysics.

Considering the problem (2.1) in the form $M\ddot{\mathbf{q}} = \mathbf{f}$, with $\mathbf{f} = \mathbf{f}_{\rm tot}(\mathbf{q}, \dot{\mathbf{q}})$, let us rewrite it first as a simple semi-explicit differential-algebraic equation (DAE):

$$\dot{\mathbf{q}} = \mathbf{v}, \quad \dot{\mathbf{v}} = \mathbf{a}, \quad \mathbf{0} = M\mathbf{a} - \mathbf{f}(\mathbf{q}, \mathbf{v}). \tag{2.10}$$

The method has coefficients to play with: $\alpha_m \ne 1, \alpha_f, \beta$, and γ. Let $\alpha = \alpha_m - \alpha_f$. The time step unknowns from t_{n-1} to t_n are $\mathbf{q}_n, \mathbf{v}_n$ and $\mathbf{a}_{n+\alpha}$, i.e., it is a staggered time steppig for the acceleration if $\alpha < 0$ (cf. (2.13)). The method reads

$$\mathbf{q}_n = \mathbf{q}_{n-1} + h\mathbf{v}_{n-1} + \frac{h^2}{2}\left((1-2\beta)\mathbf{a}_{n-1+\alpha} + 2\beta\mathbf{a}_{n+\alpha}\right), \tag{2.11a}$$

$$\mathbf{v}_n = \mathbf{v}_{n-1} + h\left((1-\gamma)\mathbf{a}_{n-1+\alpha} + \gamma\mathbf{a}_{n+\alpha}\right), \tag{2.11b}$$

$$\begin{aligned}(1-\alpha_m)M\mathbf{a}_{n+\alpha} + \alpha_m M\mathbf{a}_{n-1+\alpha} \\ = (1-\alpha_f)\mathbf{f}(\mathbf{q}_n, \mathbf{v}_n) + \alpha_f\mathbf{f}(\mathbf{q}_{n-1}, \mathbf{v}_{n-1}).\end{aligned} \tag{2.11c}$$

With the staggered interpretation, this is a second order method for $\mathbf{q}$, $\mathbf{v}$ and $\mathbf{a}$. It is a one-step multivalue (and not RK) method.

Since we can express $\mathbf{v}_n$, using (2.11b), in terms of $\mathbf{a}_{n+\alpha}$ and known stuff, and since we can further express $\mathbf{q}_n$, using (2.11a), in terms of $\mathbf{a}_{n+\alpha}$ and known stuff, the nonlinear system at time level n is only for $\mathbf{a}_{n+\alpha}$, i.e., the algebraic nonlinear iteration at the current step has a minimal number of unknowns, like for BDF methods. Calculating the Jacobian is slightly more painful than for BE and BDF2, but it is obviously possible, so the method is efficient in this sense.

In the generalized α method, we choose the four coefficients based on one parameter r, where $r = 0$ for maximum damping. The coefficients are then

$$\alpha_m = \frac{2r-1}{1+r}, \quad \alpha_f = \frac{r}{1+r}, \quad \beta = \frac{(1-\alpha)^2}{4}, \quad \gamma = \frac{1}{2} - \alpha. \tag{2.12}$$

Therefore,

$$-1 \le \alpha = \frac{r-1}{r+1} \le 0. \tag{2.13}$$

For the test equation $\ddot{q} + \omega^2 q = 0$ we have $M = 1$ and $f = -\omega^2 q$. So we get (2.11) in the form

$$\begin{pmatrix} 1 & 0 & -h^2\beta \\ 0 & 1 & -h\gamma \\ (1-\alpha_f)\omega^2 & 0 & (1-\alpha_m) \end{pmatrix} \begin{pmatrix} q_n \\ v_n \\ a_{n+\alpha} \end{pmatrix} = \begin{pmatrix} 1 & h & \frac{h^2}{2}(1-2\beta) \\ 0 & 1 & h(1-\gamma) \\ -\alpha_f\omega^2 & 0 & -\alpha_m \end{pmatrix} \begin{pmatrix} q_{n-1} \\ v_{n-1} \\ a_{n-1+\alpha} \end{pmatrix}.$$

Thus, we require the spectral radius of the matrix

$$T = \begin{pmatrix} 1 & 0 & -h^2\beta \\ 0 & 1 & -h\gamma \\ (1-\alpha_f)\omega^2 & 0 & (1-\alpha_m) \end{pmatrix}^{-1} \begin{pmatrix} 1 & h & \frac{h^2}{2}(1-2\beta) \\ 0 & 1 & h(1-\gamma) \\ -\alpha_f\omega^2 & 0 & -\alpha_m \end{pmatrix}. \tag{2.14}$$

A Matlab script can be quickly written to obtain this numerically. Note that the eigenvalues of T again depend only on the product $h\omega$, not on step size or frequency independently. Damping curves for various values of r are plotted in Figure 2.1 (cf. Figure 4 of [11]). Below

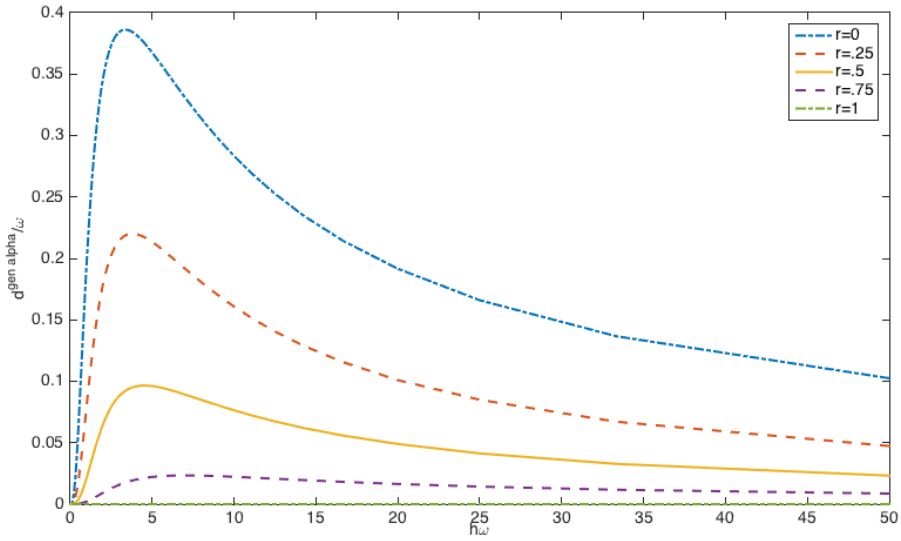

Fig. 2.1: The generalized α curves $d^{gen\ alpha}/\omega$ as a function of $h\omega$, for $r = 0 : .25 : 1$. The smaller r, the higher the curve (i.e., more damping).

we compare them with those of the method described next and given in Figure 2.2.

Theta methods. The following second order method (2.15), unlike the preceding ones, is not treated in [11] or elsewhere to our knowledge. Let us first recall that, for the ODE $\dot{\mathbf{u}} = \mathbf{f}(\mathbf{u})$, the simple θ method, depending on a parameter θ, $0 \le \theta \le 1$, reads

$$\mathbf{u}_n = \mathbf{u}_{n-1} + h((1-\theta)\mathbf{f}(\mathbf{u}_{n-1}) + \theta\mathbf{f}(\mathbf{u}_n)).$$

Considering the range $0.5 \le \theta \le 1$, the value $\theta = 1$ gives BE while the value $\theta = 1/2$ gives the trapezoidal method. So we expect the damping property to be between BE and midpoint/trapezoidal, depending on the choice of θ (see, e.g., [2]).

But this method is only first order accurate for $\theta \neq 0.5$. Instead, we can mix the trapezoid with BDF2. Letting $0 \leq \theta \leq 1$, we get

$$\mathbf{u}_n = \mathbf{u}_{n-1} + (1-\theta)\frac{h}{2}[\mathbf{f}(\mathbf{u}_{n-1}) + \mathbf{f}(\mathbf{u}_n)] + \frac{\theta}{3}[\mathbf{u}_{n-1} - \mathbf{u}_{n-2} + 2h\mathbf{f}(\mathbf{u}_n)]. \tag{2.15}$$

It is easy to verify that this method is second order accurate for all θ values on the unit interval.

For the problem $\ddot{q} + \omega^2 q = 0$ with $v = \dot{q}$, we have

$$\begin{pmatrix} q_n \\ v_n \end{pmatrix} - \left[\frac{2h\theta}{3} + (1-\theta)\frac{h}{2}\right] \begin{pmatrix} 0 & 1 \\ -\omega^2 & -d \end{pmatrix} \begin{pmatrix} q_n \\ v_n \end{pmatrix} = \begin{pmatrix} q_{n-1} \\ v_{n-1} \end{pmatrix} + (1-\theta)\frac{h}{2} \begin{pmatrix} 0 & 1 \\ -\omega^2 & -d \end{pmatrix} \begin{pmatrix} q_{n-1} \\ v_{n-1} \end{pmatrix} + \frac{\theta}{3} \begin{pmatrix} q_{n-1} \\ v_{n-1} \end{pmatrix} - \frac{\theta}{3} \begin{pmatrix} q_{n-2} \\ v_{n-2} \end{pmatrix}.$$

We can define $\mathbf{y}_n = (q_n, v_n, q_{n-1}, v_{n-1})^T$. Then $B\mathbf{y}_n = C\mathbf{y}_{n-1}$, where (assuming $d = 0$ to reduce notation)

$$B = \begin{pmatrix} 1 & -c & & \\ c\omega^2 & 1 & & \\ & & 1 & \\ & & & 1 \end{pmatrix}, \quad C = \begin{pmatrix} 1+\theta/3 & (1-\theta)h/2 & -\theta/3 & \\ -(1-\theta)h/2\omega^2 & 1+\theta/3 & & -\theta/3 \\ 1 & & & \\ & 1 & & \end{pmatrix},$$

with $c = 2h\theta/3 + (1-\theta)h/2$.

Next we find $\rho(T)$ numerically, knowing that it depends on $h\omega$ but not on h or ω separately, and use the formula $d^\theta = -\frac{2}{h}\ln(\rho(T))$ for the artificial damping coefficient. Damping

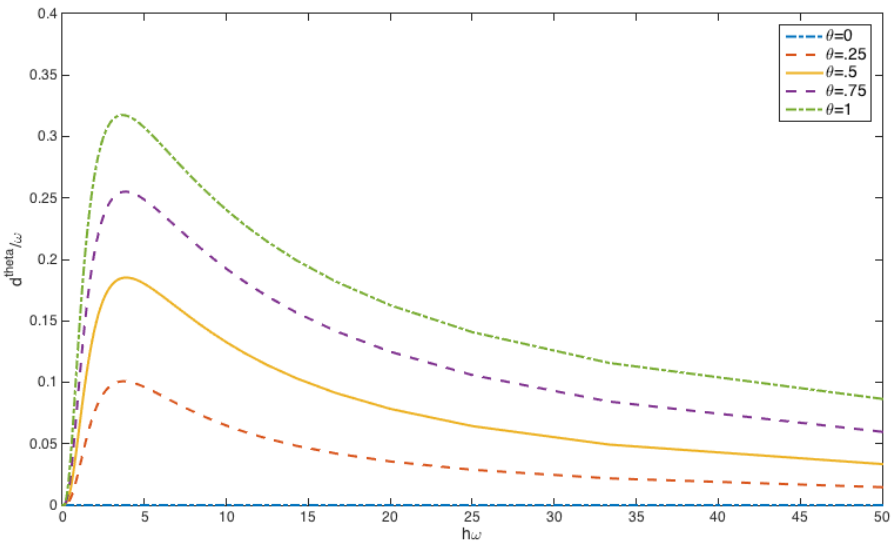

Fig. 2.2: The second order θ method curves d^{theta}/ω as a function of $h\omega$, where h is the time step, for $\theta = 0 : .25 : 1$. The larger θ, the higher the curve (i.e., more artificial damping).

curves for various values of θ are given in Figure 2.2.

Comparing the θ method just constructed to the generalized α method we find similarities. The two (families of) methods have a similar range of artificial damping behaviour as a

function of a knob $\theta = 1 - r$, although the damping amount increases faster initially as we raise θ in Figure 2.2 than when we lower r in Figure 2.1. They are both second order accurate for any value of θ or r in the given range. At each time step, they both require (upon some rewriting) the solution of an algebraic nonlinear system of the size of the original ODE system (i.e., like BE, and not larger). An advantage of generalized α is that it is a one-step method, so local step size changes are easier and it requires no extra initialization. This appears to be the winning argument in some engineering applications; however, it is less important in computer graphics. An advantage of the θ method given in (2.15) and analyzed above is its simplicity.

Before we close this subsection, let us reiterate that there is a growing movement in the visual computing community to not tolerate any method-dependent artificial damping, controlling damping instead by introduing a simple damping force as described above. At the time of this writing the jury may still be out on which approach is best; but the very discussion in a community that until recently believed only in SI is heartening.

Summary. In conclusion of this section, note that we have seen a case study where the need to move from qualitative to quantitative has caused the computer graphics community to get closer to the works of numerical analysts and applied mathematicians, for instance in geometric integration.

3. Image and surface processing. In this case study we consider common image and surface processing problems such as denoising, deblurring, inpainting, completion, registration, and salient feature detection. In the case of an image there is a given data array, say $b \in \mathbb{R}^{m \times n}$, and we typically wish to suitably upgrade it to a cleaner, sharper or more complete image $u \in \mathbb{R}^{m \times n}$. We also consider situations where the data b represent a surface of a 3D object, given as a point cloud or a triangle mesh. The output u of our algorithm would then be a similar, "cleaned" set of points (or particles) in 3D.

Many researchers have considered for this purpose regularization methods that lead to problems involving PDEs. Many consider such a regularization where *diffusion* or *anisotropic diffusion* is employed. See [29, 30, 9, 31, 28, 8, 15, 16, 21] for but a few related leading articles and books. Our own papers in this context include [1, 17] and several others.

Let us write the corresponding problem, which we shall henceforth call "*the paradigm*", as that of solving the inverse problem

$$\min_u \quad \frac{1}{2}\|f(u) - b\|^2 + \beta R(u), \tag{3.1a}$$

$$R(u) = \int_\Omega \rho(|\operatorname{grad} u|). \tag{3.1b}$$

In (3.1a), $f = f(u)$ is the *forward operator* which predicts the data (for instance, f is the identity in case of denoising, but it can be far nastier in other applications); R is the regularization operator, or *prior*, given in (3.1b) and discussed further below; and $\beta \geq 0$ is a parameter that depends on the perceived relative importance of the prior. A boundary value PDE arises upon taking the Euler-Lagrange equations of (3.1).

For the popular and simple diffusion prior, i.e., ℓ_2 on the (discrete) gradient of the function u in (3.1b), set $\rho(s) = s^2$. For the also very popular anisotropic diffusion, set $\rho(s) = s$: this corresponds to *total variation*, using ℓ_1 on the gradient (see [17] and references therein). A combination of these two using the Huber switching function was considered in [1] and elsewhere.

In some applications more complex differential operators are used in the paradigm (e.g., [5, 19]). Also, for problems involving surfaces the Laplace-Beltrami operator makes an entrance. We include all such variants in the generic discussion that follows.

3.1. To use or not to use the paradigm? That is the question. We now come to the key issue considered in the present Section 3. For many numerical analysts who dabble in image processing the answer to the question posed in the title of the present subsection is "but of course we use the paradigm! what else is worth considering?" However, many leading visual computing experts would insist that, on the contrary, this paradigm should be used rarely, if ever!

Indeed, our own experience in the past fifteen years or so indicates that the visual computing experts have a point here, as we were time and again confronted with problems that were best solved without the paradigm, achieving higher image or surface quality at a lower computational cost using often simpler computer codes. What we do next, therefore, is to discuss when the paradigm is useful and when it's best avoided. We believe that this discussion is novel.

We have crystallized the issues into two complaints.

1. The paradigm is *indirect*: we (i) start with the given discrete image b; (ii) go up to infinite function spaces; (iii) then manipulate the extended data there, possibly changing the corresponding Sobolev space; (iv) only to return an image u which is in precisely the same discrete and finite space as b. This suggests that the introduction of PDEs here is optional, not mandatory. Quality aside, it may well be that algorithms that visit data locations directly and do not require any preconditioning could have optimal complexity, unlike those involving inversion of artificial PDE problems.
2. The paradigm uses a *global*, not *local* prior. This is very unlike Photoshop and other image manipulation software. For instance, consider a photo of a particular person (it could be you!) on the beach. Manipulating the way the person looks would typically take priority over manipulating the look of the sand: but in (3.1b), any square centimeter of the given image has the same importance as any other.

The essential *advantage* in using the paradigm is that one can often obtain a more solid theoretical backing to algorithms from this global and generic point of view.

The essential *disadvantage*, however, is that this approach leads to algorithms that may be outperformed by more brute force techniques, as described above. This is so especially if the forward operator f is *simple* and the data b is of *high quality*.

These straightforward observations and arguments therefore lead to simple rules for using the paradigm. For example, if f itself involves a solution of some PDE (see, e.g., [1, 20]), then it is natural to use the paradigm. In many medical imaging problems the paradigm is likewise likely to be the way to go. Also, in [32] the authors consider (semi-)blind deconvolution using time-of-flight data, and this again creates a situation where there is not much quality to lose, hence the more solid theory that comes with the paradigm gives it the nod.

3.2. Examples where the paradigm underperforms. Below we use examples from papers we have co-authored, because we know "what is under the hood and what went on in the kitchen" in these particular works. One outcome of using published work, however, is that there are very few images in the following, and we will refer to the actual other papers for much more.

3.2.1. Surface mesh denoising [22, 23]. Consider the task of denoising a surface, rather than an image. The surface of a 3D object, such as a statue, is described by a triangle mesh: there is a vertex set V and an edge set E depicting which vertices are nodes of the same triangle. But the nodes $\mathbf{q}_i \in V$ are noisy, and we wish to remove this noise as much as we can without destroying (e.g., by over-smoothing) the actual object's apparent structure.

There is a lot of literature on denoising an image, some already mentioned before, but the start of the work described in the present subsection was in the papers [34, 27]. The authors

addressed the case of denoising images with texture, where one wished to be extra careful not to over-smooth, and used a decomposition theorem of BV spaces to propose multiscale algorithms that gradually increase fidelity to the (noisy) data.

We had wished, back in 2005, to extend these algorithms to the surface mesh case, but some months later concluded that this approach would not work. The only part adopted successfully was the gradual multiscale idea. In fact, there are several differences from image processing. The most important one is that here there is no separation between mesh locations (pixels in an image) and intensity heights. As such, the "mesh locations" are also noisy and there is the danger of vertex drift. Moreover, there is mesh sampling irregularity and potential volume shrinkage (which is visually observed in case of over-smoothing).

In fact, experience shows that the best denoising algorithms for surface meshes apply smoothing adjustments at each node essentially in the direction normal to the surface at that point. The normal direction, in turn, is determined *locally, not globally*! For each $\mathbf{q}_i \in V$, define the one-ring neighborhood $\mathcal{N}(i) \equiv \{j \mid \mathbf{e}_{i,j} = \mathbf{q}_j - \mathbf{q}_i \in E\}$. Then our vertex normal $\mathbf{n}_i$ is calculated as the average of neighbouring (triangle) face normals.[1]

A denoising iteration is derived as updating each vertex $\mathbf{q}_i$ by

$$\mathbf{q}_i \longleftarrow \mathbf{q}_i + \tau\Delta\mathbf{q}_i + \lambda_i(\hat{\mathbf{q}}_i - \mathbf{q}_i),$$

where $\{\hat{\mathbf{q}}_i;\ i = 1, \ldots, N\}$ are the given noisy data. The nonnegative parameters τ and λ_i are discussed further below. We choose

$$\Delta\mathbf{q}_i = \sum_{j\in\mathcal{N}(i)} W_{i,j}\mathbf{e}_{i,j}$$
$$W_{i,j} = w_{i,j}\mathbf{n}_i\mathbf{n}_i^T.$$

In this way, all sum contributions are proportional to the normal $\mathbf{n}_i$. Our *anisotropic Laplacian* (AL) algorithm computes $\mathbf{h}_i = \{h_{i,j} = \mathbf{e}_{i,j}^T\mathbf{n}_i \mid j \in \mathcal{N}(i)\}$ and defines

$$\Delta\mathbf{q}_i = \frac{1}{C_i}\left(\sum_{j\in\mathcal{N}(i)} g(h_{i,j})h_{i,j}\right)\mathbf{n}_i.$$

See [22] for details on determining the "edge stopping" function g, and the robust local scaling factor. This method can be seen as a simplification of *bilateral filtering with Gaussian splitting*. Note the optimal algorithm complexity (namely, $\mathcal{O}(N)$ operations: this would be hard to match using any version of the paradigm, because the latter involves inverting a surface PDE).

The AL algorithm is effective so long as there are no excessive texture or sharp edges; see examples in [22]. Sharp edges are handled in [23]. For problems with intrinsic texture we have the multiscale AL (MSAL) algorithm briefly described as follows:

For $k = 0, 1, 2, \ldots,$

$$\mathbf{q}_i \longleftarrow \mathbf{q}_i + \xi^k\Delta\mathbf{q}_i + \lambda_i(\hat{\mathbf{q}}_i - \mathbf{q}_i), \quad i = 1, 2, \ldots, N.$$

We set the step size $\tau = \xi^k, 0 < \xi < 1$, so as to reduce the effect of smoothing gradually. We also set $0 < \lambda_i \leq 1$ to accumulate smoothing contributions monotonically. Specifically,

[1] If there is no triangle mesh, just a point cloud, then a similar process ensues to determine the normal at vertex i using principal component analysis (PCA). The normal is then determined as the singular vector corresponding to the smallest singular value, the other two spanning the tangent space at this vertex or node.

$\lambda_i = \sigma_i/\bar{\sigma}, \quad \bar{\sigma} = \max\{\sigma_j;\ j = 1, \ldots, N\}$, since we want more data fidelity where there is more fine scale action.

As $k \to \infty$ the process converges at steady state to the given data $\hat{\mathbf{q}}$. Of course in practice we stop the iteration much sooner (typically, up to 4 iterations suffice). We suggest, immodestly, that the algorithm just described, which requires astonishingly few Matlab lines to code, would be hard to beat both in terms of quality of results and in terms of efficiency by any algorithm based on the paradigm.

3.2.2. Salient features and tele-registration [24]. Suppose we are given an image of several broken pieces of a plate, and we wish to "put it together", in the sense of producing an output image depicting an approximation of the original, unbroken plate (Figure 5 in [24]). Or we want to clone a mermaid from photos of a girl and a fish (Figure 14 there). Or we are given a photo of a statue occluded in part, and we wish to use texture from the non-occluded part of the statue to obtain a more complete image in a sense (Figure 6 there). Another application is where the input is a photo of an existing ancient mural, where the block pieces have moved over the centuries with respect to each other (Figure 8 in [24] and Figure 3.1 below). They should be realigned before an image completion and inpainting job is applied. These examples all call for an algorithm involving tele-registration to align the pieces, followed by a structure-driven image completion to fill in the gaps.

The algorithm proposed in [24] has the following outline:

1. Detect salient curves inside each image piece.
2. Attempt to find for each salient curve a matching curve from an adjacent piece, across the gap.
3. Use this to construct an ambient vector field surrounding all the pieces.
4. Transform (translation, rotation and scaling) each piece so salient curves line up.
5. Construct smooth bridging curves that connect such pairs across gaps.
6. Fill the gaps using structure-driven synthesis, while any remaining inside/outside holes are completed using standard inpainting tools [4, 14].

We will not dwell further on our previously published paper. Let us instead add the following:

- While working on this paper we were tempted to use our existing code based on the method described in [1] for finding the salient curves, as this sort of task is what total variation is famed for (see, e.g., [17]). However, experiments frustratingly revealed that the method of [18], which is not based on total variation, performs better for this task.
- Figure 3.1 was not included in [24] because it was deemed to be less impressive than other figures there. However, its processing steps can be found in the online project page related to this work.

Summary. In conclusion of this section, note that we have seen a case study containing several examples where better quantitative results are obtained using more localized techniques that do not involve solving PDEs. This is often (though not always) the case in visual computing tasks.

4. Conclusions. We conclude with the following comments and observations:

- Incorporating more mathematically sound techniques into methods and algorithms for computer graphics and image processing can be a rather worthwhile cause.
- Significant practical advantages have been gained in visual computing using techniques involving physics-based simulation and data-driven model calibration.
- One may occasionally be able to use mathematical analysis to obtain solid justification of algorithms: both on when they work and on when they won't.

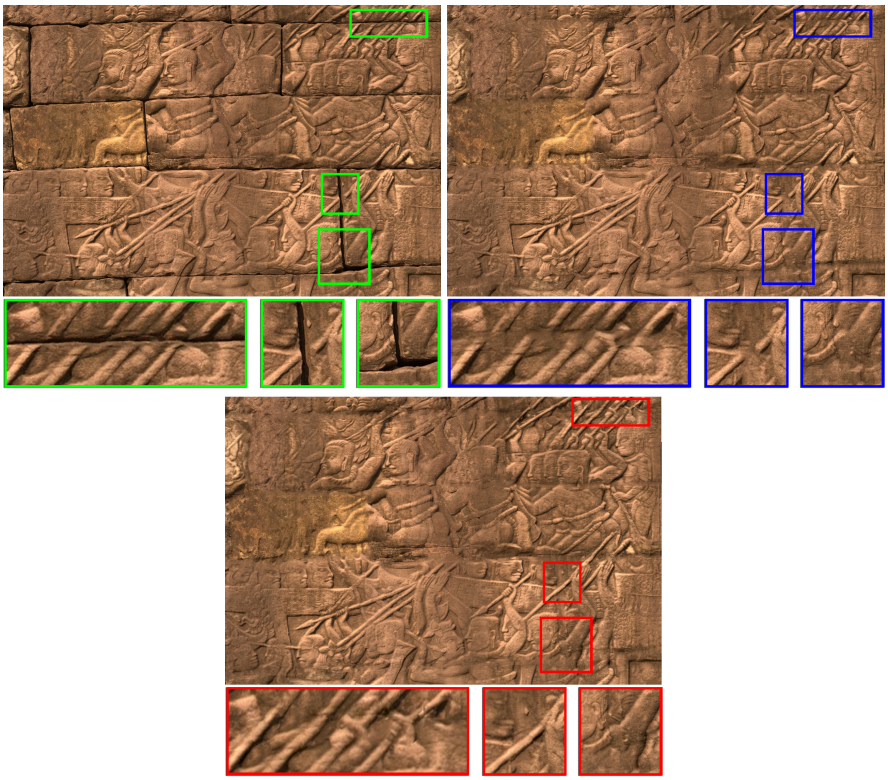

Fig. 3.1: Visual archaeology: a photograph (top left) of a wall in the archaeological park of Banteay Chhmar, Cambodia. The blocks making up the wall have moved over the centuries, and the relief correspondingly requires adjustment to make sense (observe in particular the highlighted neighborhoods). The recovery result of our algorithm is shown in the bottom subfigure. The blocks have been shifted using tele-registration, and inpainting was subsequently applied. The quality is clearly higher than direct inpainting (top right) can achieve.

- Occasionally it is even possible to bridge the gap between qualitative and quantitative (which is our secret dream).
- However, at the same time, our advice to our younger colleagues is not to get swayed by sheer mathematical prowess.
- Watch out for situations where the gap between physics and physics-based appears to be too wide (e.g., in finding fluid viscosity or damping of a soft body).
- Insisting on solving differential equations or satisfying mathematical topology theorems might lead to inferior algorithms for visual tasks: let us remember to keep our eyes on the ball, so to speak.

REFERENCES

[1] Ascher, U., Haber, E., Huang, H.: On effective methods for implicit piecewise smooth surface recovery. SIAM J. Sci. Comput. **28**, 339–358 (2006)
[2] Ascher, U.M.: Numerical Methods for Evolutionary Differential Equations. SIAM (2008)
[3] Baraff, D., Witkin, A.: Large steps in cloth simulation. In: Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH '98, pp. 43–54. ACM, New York, NY, USA (1998). DOI 10.1145/280814.280821

[4] Barnes, C., Shechtman, E., Finkelstein, A., Goldman, D.: Patchmatch: a randomized correspondence algorithm for structural image editing. ACM Transactions on Graphics (Proc. of SIGGRAPH) **28**(3), 24:1?24:11 (2009)

[5] Bertozzi, A., Schnlieb, C.B.: Unconditionally stable schemes for higher order inpainting. Communications in Mathematical Sciences **9**(2) (2011)

[6] Bonet, J., Wood, R.: Nonlinear Continuum Mechanics for Finite Element Analysis. Cambridge University Press (2008)

[7] Braess, D.: Finite Elements: Theory, Fast Solvers, and Applications in Solid Mechanics. Cambridge University Press (2007)

[8] Chan, T., Shen, J.: Image Processing and Analysis: Variational, PDE, Wavelet and Stochastic Methods. SIAM (2005)

[9] Chan, T.F., Golub, G.H., Mulet, P.: A nonlinear primal-dual method for total variation-based image restoration. SIAM J. Scient. Comput. **20**, 1964–1977 (1999)

[10] Chen, D., Levin, D., Matusic, L., Kaufman, D.: Dynamics-aware numerical coarsening for fabrication design. ACM Trans. Graphics **36(4)** (2017)

[11] Chen, Y.J., Ascher, U., Pai, D.: Exponential rosenbrock-euler integrators for elastodynamic simulation. IEEE Trans. Computer Graphics (2017)

[12] Chung, J., Hulbert, G.: A time integration algorithm for structural dynamics with improved numerical dissipation: the generalized-α method. J. Applied Mech. **60**, 371–375 (1993)

[13] Ciarlet, P.G.: Mathematical Elasticity: Three-Dimensional Elasticity, vol. 1. Elsevier (1993)

[14] Darabi, S., Shechtman, E., Barnes, C., Goldman, D., Sen, P.: Image melding: Combining inconsistent images using patch-based synthesis. ACM Transactions on Graphics (Proc. of SIGGRAPH) **31**(4), 82:1?82:10 (2012)

[15] Desbrun, M., Meyer, M., Schröder, P., Barr, A.: Implicit fairing of irregular meshes using diffusion and curvature flow. In: Proceedings SIGGRAPH, pp. 317–324 (1999)

[16] Desbrun, M., Meyer, M., Schröder, P., Barr, A.: Anisotropic feature–preserving denoising of height fields and bivariate data. Graphics Interface pp. 145–152 (2000)

[17] Doel, K.v.d., Ascher, U., Haber, E.: 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)

[18] Farbman, Z., Fattal, R., Lischinsky, D., Szeliski, R.: Edge-preserving decompositions for multi-scale tone and detail manipulation. ACM Transactions on Graphics (Proc. of SIGGRAPH) **27**(3), 67:1?67:10 (2008)

[19] de Goes, F., James, D.: Regularized kelvinlets: Sculpting brushes based on fundamental solutions of elasticity. ACM Transactions on Graphics(Proc. of SIGGRAPH 2017) **36**(4) (2017)

[20] Haber, E., Heldmann, S., Ascher, U.: Adaptive finite volume method for distributed non-smooth parameter identification. Inverse Problems **23**, 1659–1676 (2007)

[21] Hildebrandt, K., Polthier, K.: Anisotropic filtering of non-linear surface features. EUROGRAPHICS **23(3)**, 391–400 (2004)

[22] Huang, H., Ascher, U.: Fast denoising of surface meshes with intrinsic texture. Inverse Problems **24 (3)**, 034,003 (2008)

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

[24] Huang, H., Yin, K., Gong, M., Lischinski, D., Cohen-Or, D., Ascher, U., Chen, B.: "mind the gap": tele-registration for structure-driven image completion. ACM Transactions on Graphics **32**(6), 174:1–174:10 (2013)

[25] Kane, C., Marsden, J., Ortiz, M., West, M.: Variational integrators and the newmark algorithm for conservative and dissipative mechanical systems. Int. J. Numer. Meth. Eng. **49(10)**, 1295–1325 (2000)

[26] Kobis, M., Arnold, M.: Convergence of generalized-α time integration for nonlinear systems with stiff potential forces. Multibody Syst Dyn **37**, 107–125 (2016)

[27] Osher, S., Burger, M., Goldfarb, D., Xu, J., Yin, W.: An iterative regularization method for total variation based image restoration. SIAM J. multiscale model. simul. **4**, 460–489 (2005)

[28] Osher, S., Fedkiw, R.: Level Set Methods and Dynamic Implicit Surfaces. Springer (2003)

[29] Perona, P., Malik, J.: Scale-space and edge detection using anisotropic diffusion. IEEE Transactions on Pattern Analysis and Machine Intelligence **12**(7), 629–639 (1990)

[30] Rudin, L., Osher, S., Fatemi, E.: Nonlinear total variation based noise removal algorithms. Physica D **60**, 259–268 (1992)

[31] Sapiro, G.: Geometric Partial Differential Equations and Image Analysis. Cambridge (2001)

[32] Shrestha, S., Heide, F., Heidrich, W., Wetzstein, G.: Computational imaging with multi-camera time-of-flight systems. ACM Transactions on Graphics(Proc. of SIGGRAPH 2016) **35**(4) (2016)

[33] Sifakis, E., Barbic, J.: FEM simulation of 3D deformable solids: a practitioner's guide to theory, discretization and model reduction. In: ACM SIGGRAPH 2012 Courses, p. 20. ACM (2012)

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

[35] Tournier, M., Nesme, M., Gilles, B., Faure, F.: Stable constrained dynamics. ACM Trans. Graphics **34(4)** (2015)
[36] Wang, B., Wu, L., Yin, K., Ascher, U., Liu, L., Huang, H.: Deformation capture and modeling of soft objects. ACM Transactions on Graphics (Proc. of SIGGRAPH) **34**(4), 94:1–94:12 (2015)
[37] Xu, H., Sin, F., Zhu, Y., Barbič, J.: Nonlinear material design using principal stretches. ACM Transactions on Graphics (TOG) **34**(4), 75 (2015)