Algorithms Defined by Nash Iteration: Some Implementations via Multilevel Collocation and Smoothing

Gregory E. Fasshauer*, Eugene C. Gartland, Jr.† and Joseph W. Jerome‡

Abstract

We describe the general algorithms of Nash iteration in numerical analysis. We make a particular choice of algorithm involving multilevel collocation and smoothing. Our test case is that of a linear differential equation, although the theory allows for the approximate solution of non-linear differential equations. We describe the general situation completely, and employ an adaptation involving a splitting of the inversion and the smoothing into two separate steps. We had earlier shown how these ideas apply to scattered data approximation, but in this work we are interested in the application of the ideas to the numerical solution of differential equations. We make use of approximate smoothers, involving the solution of evolution equations with calibrated time steps.

Acknowledgments: The second author is supported by the National Science Foundation under grant DMS-9870420. The third author is supported by the National Science Foundation under grant DMS-9704458. We thank the referee for a number of helpful comments.

*Department of Applied Mathematics, Illinois Institute of Technology, Chicago, IL 60616. email:fass@amadeus.math.iit.edu

†Department of Mathematics and Comp. Sc. , Kent State University, Kent, OH 44242. email: gartland@mcs.kent.edu

‡Department of Mathematics, Northwestern University, Evanston, IL 60208. email: jwj@math.nwu.edu; corresponding author.

1 Introduction

Nash iteration was introduced by J. Nash in [20] and systematized in terms of generalized implicit function theorems by J. Schwartz in [22] and by J. Moser in [18] (see also [19]). Nash had reduced the embedding problem for Riemannian manifolds to the problem of existence of a solution of an under-determined system of nonlinear partial differential equations for which linearization was not continuously invertible. The smoothing 'iteration' was thus intended to deal with the case when $[D_y\mathbf{F}(x_0, y_0)]^{-1}$ is unbounded in solving for $y(x)$ near (x_0, y_0) such that $\mathbf{F}(x, y(x)) = 0$. In its final form (see [21]), this work introduced regularization as a post-conditioning step in the application of Newton methods for generalized implicit function theorems. A development of these ideas in terms of current numerical methods was given, with complete convergence proofs, in [12]. In the latter article, the special case $\mathbf{F}(x) = 0$ of root determination was considered. This continues in the present work.

We will review the nonlinear theory in the remainder of this section, and suggest various implementations of the general Nash iteration scheme in Section 2. Section 3 is devoted to a discussion of an approximate smoothing operation involving the solution of evolution equations with calibrated time steps. In Section 4 we shall then specialize the general context to a linear differential equation and to a specific inversion defined by radial basis function collocation. We shall exhibit or support the following:

- A multilevel algorithm (without smoothing) which ceases to converge.
- An added smoothing step can sustain the convergence.
- The addition of a smoothing step allows the formulation of nested iteration algorithms which greatly improve the quality of the approximation (when using basis functions of comparable support sizes).

1.1 Loss of Derivatives

Suppose that $\mathbf{F} := \mathbf{\Phi}(D^\alpha) : B_{\eta,0} \subset Y_0 \to X_0$ is a differential mapping of order m, from $B_{\eta,0} = \{v \in Y_0 : \|v - v_0\|_{Y_0} \le \eta\}$ to X_0, where X_0 and Y_0 are Banach function spaces. The real Hölder spaces and the Sobolev/Besov spaces are the prototypical examples. Assume that $\mathbf{F}$ is continuously Lipschitz (Fréchet) differentiable on $B_{\eta,0}$:

$$\|\mathbf{F}'(v) - \mathbf{F}'(w)\|_{Y_0,X_0} \le 2M\|v - w\|_{Y_0}, \qquad v, w \in B_{\eta,0}, \tag{1a}$$

$$\|\mathbf{F}'(v)\|_{Y_0,X_0} \le M, \qquad v \in B_{\eta,0}. \tag{1b}$$

It is desired to determine a root of the operator equation $\mathbf{F}(u) = 0$ by a Newton iterative method, involving an approximate inverse $\mathbf{G}_h(v)$ of the map $\mathbf{F}'(v)$. We are concerned in this paper with the case when $\mathbf{G}_h$ is defined by a standard numerical method, i. e., when $\mathbf{G}_h$ is an inverse discretization operator and h is the "meshsize" of the discretization. If $\mathbf{G}(v)$ represents the actual inverse, and w the current residual, then each differentiation of

$$[\mathbf{G}(v) - \mathbf{G}_h(v)]w$$

leads to a loss of order one in the predicted convergence order. In particular, the approximation of the identity,

$$[\mathbf{F}'(v)\mathbf{G}_h(v) - I]w = [\mathbf{F}'\mathbf{G}_h(v) - \mathbf{F}'\mathbf{G}(v)]w$$

is of apparent order $O(1)$ and thus experiences a loss of derivatives. This observation is based upon 'a priori' analysis and is by no means necessary. It does not preclude the possibility that special local approximation properties can induce the validity of an $o(1)$ estimate in certain norms. For example, it appears that finite element methods can satisfy the latter estimate when measured in dual norms; this may be due to the special feature of finite element methods in locally reproducing polynomials of a given order. Classical theories, such as the Kantorovich theory, are based on use of an approximation of the identity of the order of the residual, which translates in the context of numerical methods to a polynomial function of the grid size, chosen adaptively according to the current residual. The 'a priori' analysis suggests that these theories are only selectively applicable.

For those readers who wish an analogy to the finite-dimensional case, loss of derivatives is analogous to the situation of rank-deficient inversion. As Nash originally envisioned it, the situation of concern is when the differential operator is not continuously invertible over the entirety of its natural range space. However, the point of [12] is that an analogue of this can occur even in smooth problems where the differential operator is invertible: if the deficiency is measured in terms of approximation by discretization operators. We note that it is still possible for convergence of Galerkin approximations, say, to occur without residual *norm* decay. The Babuška-Aziz inf-sup theory describes such a situation in the linear case in a variational context (see [1] and [13]).

It is worth commenting on the role of smoothing. On the one hand, the goal of superlinear convergence is not the sole motivation. Indeed, at the level of the ground space norm (L_2, for example), one could achieve this directly by the choice of sequential mesh spacing if convergence were assured. However, typically this rate does not persist with derivatives. Clarity of shape preserving, which is related to derivatives, may dissipate. Moreover, there is no clear link to computational complexity at this level of analysis. Important computational procedures such as multigrid attempt to correlate the dimension of the Galerkin subspace with the work function in a linear or log-linear manner. When this is done, residual decay is an essential component. The Nash iteration we propose combines residual decay with a superlinear rate of convergence persisting through derivatives at least the order of the differential operator. The computed approximation thus displays not only accuracy, but integrity with respect to shape and contour. Moreover, our analysis is primarily intended for the case of nonlinear problems, where the same conclusions hold.

1.2 Framework for the Postconditioning Iteration

Definition 1.1 *We are led to introduce a scale of Banach spaces X_σ and Y_σ, where the following properties hold:*

- *These spaces are continuously embedded, so that, for $\sigma' < \sigma$,*

$$X_0 \supset X_{\sigma'} \supset X_\sigma \supset X_\infty := \cap_{\sigma \geq 0} X_\sigma. \tag{2}$$

There is a similar statement for $\{Y_\sigma\}$.

- *There exists a smoothing $\mathbf{S}_t$: $X_0 \to X_\infty$ for $t \geq 1$:*

$$\|\mathbf{S}_t v - v\|_{X_0} \to 0, \qquad \text{as } t \to \infty; \tag{3a}$$

$$\|\mathbf{S}_t v\|_{X_p} \leq C_{rp} t^{p-r} \|v\|_{X_r}, \qquad 0 \leq r \leq p; \tag{3b}$$

$$\|\mathbf{S}_t v - v\|_{X_r} \leq C_{rp} t^{-(p-r)} \|v\|_{X_p}, \qquad 0 \leq r \leq p. \tag{3c}$$

There is a similar set of inequalities assumed for the spaces $\{Y_\sigma\}$.

It is necessary, finally, to express how the mapping $\mathbf{F}$, its derivatives, and approximate inverses behave relative to this scale of spaces.

Definition 1.2 *We explicitly assume the following.*

- $\mathbf{F} : B_{\eta,0} \cap Y_\sigma \to X_\sigma$ *is a well-defined map for $0 \leq \sigma \leq s$ and s sufficiently large (cf. (10c)). We assume without loss of generality that the center v_0 of $B_{\eta,0}$ is in Y_s so that it serves as the common center of all smooth balls $B_{\eta,\sigma}$ to follow.*

- *Functional substitution in Φ is essentially of linear growth (cf. [21] for a precise meaning). In other words, the X_s norm of the residual can be estimated from above by the Y_s norm, plus the s-th power of the Y_γ norm, where γ is defined immediately below as the loss of derivatives.*

- *There exist numerical inversion operators with loss of derivatives. Specifically, there is a constant $\gamma \geq 1$ and a family $\mathbf{G}_h(v) : X_\gamma \to Y_0$, of linear mappings depending on parameters h and v, and a continuous monotone increasing function $\tau : [0, b] \to [0, \infty)$, $\tau(0) = 0$, such that, for all $w \in X_\gamma$,*

$$\|[\mathbf{F}'(v)\mathbf{G}_h(v) - I]w\|_{X_0} \leq \tau(h)\|w\|_{X_\gamma},$$

$$v \in B_{1,\gamma} \subset Y_\gamma. \tag{4}$$

Here, we have used the symbol $B_{1,\gamma} \subset Y_\gamma$ to denote the ball of radius 1.

- *The maps $\{\mathbf{G}_h(v)\}$ are uniformly bounded in h and v from X_σ to $Y_{\sigma-\gamma}$ for $\gamma \leq \sigma \leq s$:*

$$\|\mathbf{G}_h(v)w\|_{Y_{\sigma-\gamma}} \leq M\|w\|_{X_\sigma}. \tag{5}$$

Remark Inequality (4) is the key inequality which distinguishes, say, Nash iteration from Kantorovich iteration. In the latter, one requires (4) to hold with X_γ replaced by X_0. The failure of (4) with X_0 and (5) constitutes the *loss* of derivatives characterization.

The following result was proved in [7]. The corresponding result for Hölder spaces was proved by Hörmander [11].

Theorem 1.1 *Let $\phi \in L_1(\mathbb{R}^d)$ be a kernel whose Fourier transform $\hat{\phi}$ is compactly supported and infinitely differentiable in $\mathbb{R}^d$ and $\hat{\phi}(\omega) = 1$ in a neighborhood of the origin. Furthermore, let $\phi_t = t^d \phi(t\cdot)$, $t \geq 1$. Then ϕ is a smoothing kernel defining smoothing operators in the above sense by convolution with ϕ_t. Thus, for $u \in W_p^k(\mathbb{R}^d)$ ($u \in B_{p,\infty}^k(\mathbb{R}^d)$) ϕ satisfies, for a and b not exceeding k,*

$$\lim_{t\to\infty} \|\phi_t * u - u\|_p = 0,$$

$$\|\phi_t * u\|_{b,p} \leq C t^{b-a} \|u\|_{a,p}, \qquad 0 \leq a \leq b,$$
$$\|\phi_t * u - u\|_{a,p} \leq C t^{-(b-a)} \|u\|_{b,p}, \qquad 0 \leq a \leq b.$$

Here the norms $\|\cdot\|_{a,p}$ and $\|\cdot\|_{b,p}$ are used to denote either Sobolev or Besov space norms depending on whether or not a, b are integers.

In principle, this result forms the basis for smoothing via a convolution equivalent to high frequency cutoff. In practice, the implementation in the spatial domain is badly conditioned because of the highly oscillatory nature of the convolution kernel.

1.3 Nash Iteration

We discuss the approximate Newton method with *postconditioning* (Nash iteration) and the fundamental parameters. We begin with u_0, the initial iterate, and identify u_0 with the center v_0 of $B_{\eta,0}$. More generally, u_0 will be the center of the closed balls $B_{c,\sigma} \subset Y_\sigma$ of radius c. We assume

$$f_0 = \mathbf{F}(u_0) \in X_s, \qquad u_0 \in Y_s,$$
$$\|f_0\|_{X_0} \leq \rho^{-\lambda}, \tag{6}$$

where $\rho > 0$ and $1 < \lambda < s$. We need a parameter β, satisfying $1 < \beta < 2$, to control superlinear convergence later. In terms of these quantities, we require

$$\|f_0\|_{X_s} \leq M \rho^{(s-\lambda)\beta}. \tag{7}$$

Similarly, we select a parameter $\theta > 1$ to describe acceleration of convergence, and parameters

$$t_k = \rho^{\theta\beta^k}, \qquad k \geq 1, \tag{8}$$

to serve as smoothing speeds. We assume inductively that $u_{k-1} \in Y_s \cap B_{1,\gamma}$ has been defined for $k \geq 1$. Then u_k is defined by Newton/postconditioning iteration:

$$u_k - u_{k-1} = -\mathbf{S}_{t_k} \mathbf{G}_{h_k}(u_{k-1}) \mathbf{F}(u_{k-1}). \tag{9}$$

Here the parameter h_k is subject to specification (cf. (11) below), so that the function $\tau(h_k)$ is of the order of $\|\mathbf{F}(u_{k-1})\|$ (see (11) below). The iteration is required to converge in Y_μ for specified $\mu \geq 1$. The relations among γ, s, λ, β, θ, and μ are given in the definition to follow. For ease of estimation, we assume that M is chosen so that M is sufficiently large.

Definition 1.3 *The relations satisfied by the exponents and parameters are presented here. We require*

$$1 < \beta < 2, \qquad 1 < \theta, \qquad 1 \le \gamma \le \mu < \lambda < s, \tag{10a}$$

$$\lambda \ge \max\{2\beta\gamma(2-\beta)^{-1},\ \beta(\gamma+\mu\beta)\} + \lambda_0(2-\beta)^{-1}, \tag{10b}$$

$$s \ge \max\{\theta\gamma(\theta-1)^{-1},\ \lambda+\theta\gamma(\beta-1)^{-1}\} + s_0\max\{(\theta-1)^{-1},\ (\beta-1)^{-1}\}, \tag{10c}$$

for arbitrary $\lambda_0 > 0$, $s_0 > 0$. *The number* ρ *is required to be sufficiently large (see the remark at the end of this section). The numbers* h_k *are selected so that*

$$\tau(h_k) \le M^5 \|\mathbf{F}(u_{k-1})\|_{X_\gamma}, \qquad k = 1, \ldots. \tag{11}$$

The choices $\theta = 1.16$, $\lambda = \frac{29}{6}$, $s = \frac{47}{6}$ and $\beta = 1.4$, for $\mu = \gamma = 1$, satisfy the necessary inequalities (10 a-c) with $\lambda_0 = 0.1$ and $s_0 = 0.016$. They are restrictive upon ρ, however, and should eventually be replaced by other choices.

1.4 The Superlinear Convergence Theorem

Theorem 1.2 *Suppose the hypotheses of the Definitions are satisfied, and that the initial iterate* $u_0 \in B_{\eta,0} \cap Y_s$ *satisfies* (6) *and* (7). *Then the Newton iterates* $\{u_k\}$, *defined by the adaptive, postconditioning procedure* (9), *converge to a root* u *of* $\mathbf{F}$ *in* $B_{\eta,0} \cap B_{1,\mu}$. *The superlinear convergence in* Y_0 *is described by the estimate*

$$\|u - u_k\|_{Y_0} \le M^4 \rho^{-(\lambda-\beta\gamma)\beta^{k-1}} / \beta^{k-1},$$

for $k = 1, 2, \ldots$. *The superlinear convergence in* Y_μ *is described by*

$$\|u - u_k\|_{Y_\mu} \le M^5 \rho^{-\lambda_0\beta^{k-1}} / \beta^{k-1},$$

for $k = 1, 2, \ldots$.

The proof, described fully in [15], proceeds by establishing the following statements recursively for $k \ge 0$:

$$\|\mathbf{F}(u_k)\|_{X_0} \le \rho^{-\lambda\beta^k},$$

$$\|\mathbf{F}(u_k)\|_{X_s} \le M\rho^{(s-\lambda)\beta^{k+1}},$$

$$\|\mathbf{F}(u_k)\|_{X_\gamma} \le M^2\rho^{-(\lambda-\beta\gamma)\beta^k},$$

$$\|u_{k+1} - u_k\|_{Y_s} \le \tfrac{1}{2}\rho^{(s-\lambda)\beta^{k+2}},$$

$$\|u_{k+1} - u_0\|_{Y_s} \le \rho^{(s-\lambda)\beta^{k+2}},$$

$$\|u_{k+1} - u_k\|_{Y_0} \le M^4\rho^{-(\lambda-\beta\gamma)\beta^k},$$

$$\|u_{k+1} - u_0\|_{Y_0} \le \eta,$$

$$\|u_{k+1} - u_k\|_{Y_\mu} \le M^5\rho^{-\lambda_0\beta^k},$$

$$\|u_{k+1} - u_0\|_{Y_\mu} \le 1.$$

Remark The preceding theorem can serve as an existence theorem as well as an approximation theorem. The reader may inquire as to what fails in the absence of an operator root. The most likely source of failure is the breakdown in selecting a compatible pair ρ, u_0, such that ρ is sufficiently large and the starting iterate u_0 satisfies (6, 7) in relationship to ρ. By sufficiently large here we mean the technical comparisons described in (2.20 a–e) of [12] (or (7.21 a–e) of [15]) which restrict the sizes of certain ratios of powers of M to powers of ρ or certain classical functions of ρ. The remaining restrictions are detailed in (1a, b) and Definition 1.2. The number η which appears in the induction inequalities above is defined in (2.20 d) of [12] (or (7.21 d) of [15]) as a bound for the ratio of M^4 to a classical function of ρ. As such, it is not a restriction on this ratio, but a measure of it.

2 The Connection to Multilevel Iteration

In a previous paper [7] we described the connection between multilevel scattered data interpolation as suggested by Floater and Iske [8] and approximate Newton iteration. In this section we will elaborate on those ideas as applied to the solution of nonlinear partial differential equations.

In the following we will denote a generic nonlinear PDE with

$$\mathcal{L}u = f.$$

Here $f \in X_\sigma$ and $u \in Y_\sigma$ with X_σ and Y_σ appropriate Sobolev or Besov spaces (see [7]). We denote the problem linearized at u_{k-1} by

$$L_{u_{k-1}} v = f - \mathcal{L}u_{k-1},$$

which suggests that we consider the mapping $\mathbf{F}$ to be of the form

$$\mathbf{F}(u) = \mathcal{L}u - f.$$

For the algorithms discussed in this section the other mappings introduced earlier, the numerical inversion $\mathbf{G}_h$ and the smoothing $\mathbf{S}_t$, remain arbitrary (subject to the conditions stated above). In our numerical experiments later on we will represent $\mathbf{G}_h$ by collocation with locally supported radial basis functions on a grid $\mathcal{X}$ with "meshsize" h, and approximate $\mathbf{S}_t$ by a forward Euler discretization of certain evolution equations based on the (bi)-harmonic operator (see Section 3 for details).

An approximate Newton scheme with postconditioning is then given by (9). Using the notation just introduced this is equivalent to

$$\begin{aligned} L_{u_{k-1}} v &= f - \mathcal{L}u_{k-1} \\ u_k &= u_{k-1} + \mathbf{S}_{t_k} v \end{aligned} \tag{12}$$

It is possible to interpret this basic algorithm in various different ways. We now describe several possibilities.

2.1 The "Simple" Algorithm

The specific choice of numerical inversion $\mathbf{G}_h$ prescribes via (11) that in each iteration the meshsize h_k be coupled to the size of the residual $\|\mathbf{F}(u_{k-1})\| = \|f - \mathcal{L}u_{k-1}\|$. Since this coupling is very difficult to monitor in practice, in our experiments we work with a hierarchy of computational grids $\{\mathcal{X}_k\}$ whose meshsizes change regularly (i.e. $h_k = h_{k-1}/2$). The connection between this finite-dimensional computational framework and the infinite-dimensional function spaces used in the previous section is either done as in the classical finite element or finite difference framework, where the approximate solutions u_k are represented by their nodal values on the grids, or – as in our experiments – as finite-dimensional linear spaces spanned by the translates (about the mesh points) of an (everywhere defined) radial basis function.

The first algorithm is a direct analog of the Floater-Iske method which we have applied previously to scattered data approximation and linear differential equations (see [4], [5], [6], [7]). The correction v is computed with one single computational step on the next finer grid $\mathcal{X}_k$. Thus the "simple" algorithm can be given as

Algorithm 2.1 *("Simple Algorithm")*

1. *Let* $u_0 = 0$.
2. *For* $k = 1, 2, 3, \ldots$
   (a) *Solve* $L_{u_{k-1}}v = f - \mathcal{L}u_{k-1}$ *on* $\mathcal{X}_k$.
   (b) *Smooth the residual and update:* $u_k = u_{k-1} + \mathbf{S}_{t_k}v$.

Step (a) corresponds to an application of $\mathbf{G}_{h_k}(u_{k-1})$ to the residual $\mathbf{F}(u_{k-1})$, and step (b) describes the postconditioning via $\mathbf{S}_{t_k}$ as defined in (9).

For $\mathcal{L} = L$ and $\mathcal{L} = L = I$, respectively, this covers the cases studied earlier in the references listed above.

2.2 Nested Iteration

In this section we offer two refinements of Algorithm 2.1. The first variation is to add an "inner" iteration to the "outer" Newton iteration. One can view this as Nash iteration where the numerical inversion $\mathbf{G}_h$ is implemented by an *iterative* numerical method. We fix a number N of nested computational grids $\mathcal{X}_1 \subset \mathcal{X}_2 \subset \cdots \subset \mathcal{X}_N$. It is the finest grid, $\mathcal{X}_N$, on which we want to know the Newton approximation. Thus, the meshsize h_k is *fixed* in this version. The hierarchy of grids is used to compute the update v by solving the linearized problem on these grids. The algorithm for this case is

Algorithm 2.2 *(Nested Iteration I)*

1. *Let* $u_0 = 0$.
2. *For* $k = 1, 2, 3, \ldots$
   (a) *Let* $v_0 = 0$.

(b) *For* $\ell = 1, 2, \ldots, N$

i. *Solve the linearized problem*

$$L_{u_{k-1}} w_\ell = f - \mathcal{L} u_{k-1} - L_{u_{k-1}} v_{\ell-1}$$

on $\mathcal{X}_\ell$.

ii. *Update* $v_\ell = v_{\ell-1} + w_\ell$.

(c) *Now* $L_{u_{k-1}} v_N = f - \mathcal{L} u_{k-1}$, *and so*
Smooth the correction and do Newton update:

$$u_k = u_{k-1} + \mathbf{S}_{t_k} v_N.$$

The motivation for this version of the algorithm is efficient computation of the Newton update v_N, i.e., instead of computing v_N on $\mathcal{X}_N$ directly, this is done in an efficient multilevel procedure using the coarser grids (which is of course the same philosophy used in multigrid methods).

In theory the meshsize of the finest mesh, $\mathcal{X}_N$, would have to change from one iteration to the next (as explained for the "simple" algorithm). Thus, a more accurate notation for the computational grids would be $\{\mathcal{X}_\ell^k\}_{\ell=1}^N$. In principle, N could also vary with k.

In our implementation we keep the computational grids fixed, i.e., we obviously violate the dependence of the meshsize on the size of the residual (as prescribed by (11)), and simply return to the coarsest grid to start the computation of the next Newton update. Without any added smoothing Algorithm 2.2 was already suggested by Wendland [24]. His description, however, was limited to the case in which $\mathbf{G}_h$ is given by a multilevel Galerkin approach using compactly supported radial basis functions. The motivation for the return to the coarsest grid in [24] was given by the fact that the "simple" algorithm ceased to converge at some level – but a return to the coarsest grid led to a procedure with better than linear convergence. In the multigrid literature this process is also known as Kaczmarz smoothing [16].

A second variation of Algorithm 2.1 is given by

Algorithm 2.3 *(Nested Iteration II)*

1. *Let* $u_0 = 0$.
2. *For* $k = 1, 2, 3, \ldots$

(a) *Let* $v_0 = 0$.

(b) *For* $\ell = 1, 2, \ldots, k$

i. *Solve the linearized problem*

$$L_{u_{k-1}} w_\ell = f - \mathcal{L} u_{k-1} - L_{u_{k-1}} v_{\ell-1}$$

on $\mathcal{X}_\ell$.

ii. *Update* $v_\ell = v_{\ell-1} + w_\ell$.

(c) *Now* $L_{u_{k-1}} v_k = f - \mathcal{L} u_{k-1}$, *and so*
Smooth the correction and do Newton update:

$$u_k = u_{k-1} + \mathbf{S}_{t_k} v_k.$$

The only difference to Algorithm 2.2 is that now the finest grid changes from one Newton iteration to the next, i.e., u_k is the Newton approximation on $\mathcal{X}_k$. This is more in the spirit of the theory (variable meshsize). It also has as a consequence that the number of iterations in the inner loop is no longer fixed.

There are, of course, many other possible implementations of the basic Nash iteration scheme (12) similar in flavor to the many different strategies for cycling through multigrid. Moreover, adaptive mesh refinement is certainly one avenue which needs to be explored. Another question related to the nested iterations is whether it is appropriate to employ further smoothing steps during the inner iterations. One could then interpret the resulting procedure as one for which the linearized problem also is solved by a Newton method – via an application of the "simple" algorithm.

3 Approximate Smoothing via Explicit Time Stepping

At its heart, Nash iteration is based upon a framework of smooth function spaces (cf. the estimates for s and λ following Definition 1.3). In particular, we must be able to infer the inequalities of Theorem 1.1 for $a, b \leq s$. Actually implementing this is quite rigorous for $s = 8$, for example. This is because of saturation encountered with positive kernels such as the Gaussian kernel. In order to obtain higher order convergence, oscillatory kernels must be employed if one is using direct kernel convolution. In order to circumvent such computational difficulty, it is customary to use approximate smoothing, based on the solution of initial value problems. It is not our intent in this paper to present a complete mathematical exposition of approximate smoothing. However, for the reader's interest, we wish to summarize the mathematical foundation upon which it rests. We describe this now.

It has been known for some time via semigroup methods and the resolvent calculus that smoothing can be defined involving limits of the latter (see [2], Section 4.2, for elaboration). What makes this possible is the fundamental property of the resolvent calculus given by

$$\lim_{\lambda \to \infty} \lambda R(\lambda, A) f = f,$$

for all f in the semigroup domain. Here, R denotes the resolvent of A:

$$R(\lambda, A) = (\lambda I - A)^{-1}.$$

It is this property which is made precise in [2] in terms of saturation. In general, the smoother the domain of A, the more rapid the convergence in terms of classical rates. If strict correlation with the smoothing requirements on S_t were maintained, one would require that A be of order at least s.

It is possible to construct the resolvent convergence approximately via an implicit time stepping method. Indeed, by properly identifying λ with $1/\Delta t$, one deduces the convergence of the implicit or backward

Euler method (the Trotter-Kato theorem is a rigorous statement of this fact, sometimes called the method of horizontal lines; see [14]). One can actually verify directly the convergence of the explicit method (forward Euler method), and thus deduce the smoothing properties of the latter, via the identification with the implicit limit. In what follows, we employ the explicit method rather than the implicit method. However, we employ only the Laplacian and its operator square for the approximate smoothing (see (15, 16)). We briefly discuss the time stepping now.

We consider this in the context of the numerical experiments described below, which involve a two-point boundary value problem with homogeneous Dirichlet boundary conditions on $(0,1)$. Given the Newton correction v, we take as the smoothed correction, $\mathbf{S}_t v$, the solution at time t of one of the following problems:

$$\begin{gathered} s_t = s_{xx}\,, \qquad 0 < x < 1\,, \quad 0 < t \\ s(0,t) = s(1,t) = 0 \\ s(x,0) = v(x) \end{gathered} \tag{13}$$

or

$$\begin{gathered} s_t = -s_{xxxx}\,, \qquad 0 < x < 1\,, \quad 0 < t \\ s(0,t) = s(1,t) = s_{xx}(0,t) = s_{xx}(1,t) = 0 \\ s(x,0) = v(x)\,. \end{gathered} \tag{14}$$

The first of these is the classical heat equation in one spatial dimension, for which the time-dependent Green's function (or "heat kernel") is precisely the Gauss-Weierstrass kernel, here convolved with the odd-periodic extension of v (as utilized in the smoothing context in [5] and [7]).

The second problem above gives a similar evolution but for the biharmonic operator. Approximations to the actions of these smoothers can be efficiently realized using finite differences and explicit time stepping.

In the earlier notation, $As = s_{xx}$ in the harmonic case, and $As = -s_{xxxx}$ in the biharmonic case. Since the order of A serves as a saturation index, the biharmonic smoother is inherently superior in relation to the general theory.

On a uniform mesh ($x_i = i\Delta x$, $i = 0,\ldots,n$, $\Delta x = 1/n$), denote the standard central difference approximations

$$s_{xx}(x_i) \approx \delta_x^2 s_i := \frac{s_{i+1} - 2s_i + s_{i-1}}{\Delta x^2}$$

and

$$s_{xxxx}(x_i) \approx \delta_x^4 s_i := \delta_x^2 \delta_x^2 s_i = \frac{s_{i+2} - 4s_{i+1} + 6s_i - 4s_{i-1} + s_{i-2}}{\Delta x^4}\,.$$

Then we approximate (13) by

$$\frac{s_{m+1} - s_m}{\Delta t} = \delta_x^2 s_m\,, \quad m = 0,1,\ldots$$

or

$$s_{m+1} = \left(I + \Delta t\, \delta_x^2\right) s_m =: G_1 s_m\,, \tag{15}$$

and we similarly approximate (14) by

$$s_{m+1} = \left(I - \Delta t\, \delta_x^4\right) s_m =: G_2 s_m \,. \tag{16}$$

Here s_m denotes the approximate solution on time level $t_m := m\Delta t$.

Now the eigenfunctions of the spatial differential operators d^2/dx^2 and d^4/dx^4 and their finite-difference counterparts δ_x^2 and δ_x^4 are all the same, namely, the Fourier sine modes $\sin(k\pi x)$. From this one can obtain explicitly the eigenvalues of the iteration matrices G_1 and G_2:

$$\lambda_k(G_1) = 1 - \frac{4\Delta t}{\Delta x^2}\sin^2\left(\frac{k\pi\Delta x}{2}\right) , \quad \lambda_k(G_2) = 1 - \frac{16\Delta t}{\Delta x^4}\sin^4\left(\frac{k\pi\Delta x}{2}\right) ,$$

$k = 1, \ldots, n-1$. The behavior of these damping factors as functions of the wavenumber k for different values of the ratios $\Delta t/\Delta x^2$ and $\Delta t/\Delta x^4$ is depicted in Figure 1.

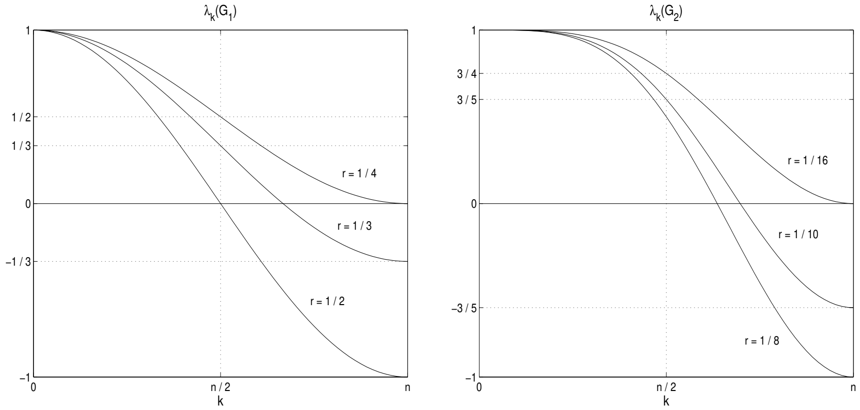

Figure 1: Damping factors for explicit time stepping with heat kernel (left) and biharmonic operator (right) for different values of the ratio $r := \Delta t/\Delta x^2$ (left) and $r := \Delta t/\Delta x^4$ (right).

Thus for the case of smoothing with the heat kernel, we obtain the following distinguished values of the ratio $r := \Delta t/\Delta x^2$: $r = 1/2$ (maximum stable time step), $r = 1/3$ (maximum smoothing of the high-frequency modes $n/2 < k < n$)—this corresponds to the "damped Jacobi" smoother utilized in some multigrid algorithms [10]—and $r = 1/4$ (maximum smoothing of high-frequency modes subject to the restriction $\lambda_k > 0$). For the biharmonic smoother, the corresponding ratios of $r := \Delta t/\Delta x^4$ are $r = 1/8$, $1/10$, and $1/16$. Aspects of these approximate smoothers can be controlled by choosing different values of these ratios along with varying the number of time steps taken.

4 Numerical Experiments

We will now illustrate the Nash iteration scheme with implementations of the algorithms of Section 2 for a linear problem. All examples will be based on the following simple linear two-point boundary value problem:

$$-u''(x) + \pi^2 u(x) = 2\pi^2 \sin \pi x, \qquad x \in (0,1), \tag{17}$$

where $u(0) = u(1) = 0$. The unique solution to this problem is given by $u(x) = \sin \pi x$.

Since the differential equation in our test case is linear, $\mathbf{G}_h(u)$ does not depend on u, and so the Newton scheme of (12) is simply

$$\begin{aligned} Lv &= f - Lu_{k-1} \\ u_k &= u_{k-1} + \mathbf{S}_{t_k} v \end{aligned} \tag{18}$$

beginning with $u_0 = 0$.

4.1 Radial Basis Functions

We implement the approximate inverse $\mathbf{G}_h$ using radial basis function (RBF) collocation. The radial functions $\phi_{\ell,\kappa}$ we use come from a class of functions introduced by Wendland [23]. These functions are compactly supported and built of polynomial pieces. Simple direct formulas for computing the $\phi_{\ell,\kappa}$ are (see [4])

$$\begin{aligned} \phi_{\ell,0}(r) &\doteq (1-r)_+^{\ell}, \\ \phi_{\ell,1}(r) &\doteq (1-r)_+^{\ell+1} \left[(\ell+1)r + 1\right], \\ \phi_{\ell,2}(r) &\doteq (1-r)_+^{\ell+2} \left[(\ell^2+4\ell+3)r^2 + (3\ell+6)r + 3\right], \\ \phi_{\ell,3}(r) &\doteq (1-r)_+^{\ell+3} \left[(\ell^3+9\ell^2+23\ell+15)r^3 \right. \\ & \quad \left. + \, (6\ell^2+36\ell+45)r^2 + (15\ell+45)r + 15\right]. \end{aligned}$$

Here $\doteq$ denotes equality up to a constant factor, and $r = \|x\|$, where $x \in \mathbb{R}^d$, so that the composition of the univariate function $\phi_{\ell,\kappa}$ with the norm indeed yields a radial function. Note that the cut-off function $(\cdot)_+$ ensures the compact support of these functions. The two indices ℓ and κ are related to the space dimension d, and to the smoothness of the basis functions. More precisely, if one intends to work in $\mathbb{R}^d$, then one should take $\ell = \lfloor \frac{d}{2} \rfloor + \kappa + 1$, where κ ensures that $\phi_{\ell,\kappa} \in C^{2\kappa}$. With this choice the functions are positive definite, and thus the associated collocation matrix is assured to be invertible.

Solving the boundary value problem by collocation corresponds to satisfying the differential equation and the boundary conditions pointwise on some given set of points $\mathcal{X} = \{x_1, \ldots, x_n\}$. One of the advantages of radial basis functions is that these points do not have to be on any kind of (regular) grid. Thus, radial basis functions can be considered as a *meshless* method. Furthermore, many different types of boundary conditions can be handled very easily by the collocation method. The expansion we use for the solution of our one-dimensional example (17) is of the form

$$u(x) = c_1 \phi(|x|) + c_n \phi(|x-1|) + \sum_{j=2}^{n-1} c_j L^{(2)} \phi(|x - x_j|), \quad x \in \mathbb{R}. \tag{19}$$

Here the collocation points $x_1 = 0$ and $x_n = 1$ lie on the boundary, and the remaining x_j are located in the interior of the interval. The superscript $^{(2)}$ indicates that L acts on ϕ as a function of the second variable, i.e., the center location. Expansion (19) is motivated by a connection to scattered Hermite interpolation which ensures the invertibility of the related collocation matrix (for more details see [3], [5] or [9]). We point out that the collocation points and the centers are both chosen to coincide with the mesh points x_k. We follow this widely used approach since this is the situation for which the theoretical foundation is sound. For the more general situation not much is known theoretically.

We use the general form of expansion (19) on all computational grids $\mathcal{X}_k$. The basis functions ϕ differ from mesh to mesh by a change in their support size. We point out that even though the computational grids $\mathcal{X}_k$ are nested, the approximation spaces

$$\mathcal{S}_k = \operatorname{span}\{\phi(\|(\cdot - x_j^{(k)})/\delta_k\|),\ x_j^{(k)} \in \mathcal{X}_k\}$$

are not. Here δ_k denotes the scale for the support size of ϕ on $\mathcal{X}_k$.

In order to get a rough idea for the loss of derivatives (see (4)) to be expected when using radial basis function collocation we refer to the only known convergence estimates for collocation with radial basis functions. In [9] upper bounds for L_∞ convergence rates of the solution of an m-th order elliptic partial differential equation via symmetric radial basis function collocation on a single fixed grid were given as

$$\|u - u_h\|_\infty = \mathcal{O}(h^{2\kappa - m - d/2}). \tag{20}$$

For this result the authors required that the smoothness of the basis satisfies $2\kappa > m + d/2$ and the smoothness of the solution $u \in H^\sigma(\Omega)$ satisfies $\sigma > m + d/2$. According to a remark in [9] one can expect to pick up an additional factor of $h^{d/2}$ for the L_2 convergence rate. The approximate solution u_h in (20) is a linear combination of basis functions which have a fixed support size throughout. Convergence is achieved by using more and more centers, thereby decreasing the "meshsize" h.

For our problem we have $d = 1$ and $m = 2$, so that the basis functions have to be chosen to be at least in $C^{2.5}$. The expected convergence order follows from any additional smoothness built into the basis. In our experiments we used the function $\phi_{5,3}$ which is C^6 and positive definite in $\mathbb{R}^3$. Therefore the L_2 convergence rate one should expect for using this function for collocation on a single grid is at least $\mathcal{O}(h^4)$, and we should therefore expect γ to be 4 at least. (In practice, on a single grid, we have observed rates which are almost $\mathcal{O}(h^8)$.)

As indicated in Section 2 we make use of a hierarchy of computational grids $\{\mathcal{X}_k\}$. In our experiments below we will always take $\mathcal{X}_1$ to consist of 5 equally spaced points in $[0, 1]$. The finer grids are obtained by halving the meshsize of the previous grid.

The linear systems arising at each level are solved with the conjugate gradient method and Jacobi preconditioning. In order to have a meaningful initial approximation the support of the basis functions for the coarsest grid is chosen so large that the matrix on the coarsest grid is a dense one.

For the subsequent levels we use two different strategies to determine the support size of the basis functions. In one group of tests we insist on keeping the bandwidth of the collocation matrices fixed, i.e., the support size of the basis functions is halved from one level to the next. This guarantees that the computational cost remains proportional to the problem size. However, as can be seen by comparing Tables 1 and 3 the convergence properties suffer. Therefore, in a second group of experiments, we let the bandwidth increase slightly from one level to the next. This has beneficial effects on the convergence, but at an added computational expense.

The errors and residuals in the tables below are computed either on the next finer grid, or on a common very fine grid. The convergence rates are determined as

$$\text{rate} = \ln\left(\frac{w_{k-1}}{w_k}\right) / \ln 2, \tag{21}$$

where w_k is either the ℓ_2 error or the ℓ_2 norm of the residual at level k.

In order to judge how well the various implementations fit into the theory we offer another interpretation of equation (11), the coupling between the meshsize and the norm of the residual. If we consider $\tau(h_k)$ to be of the form $\tau(h_k) = h_k^\gamma$, and assume that the mesh refinement is done uniformly for all levels such that $h_{k+1} = h_k/\alpha$ for some $\alpha > 1$, then taking the ratio of

$$h_k^\gamma = M^5 \|\mathbf{F}(u_{k-1})\|$$

(corresponding to the maximal allowable meshsize) for two consecutive values of k implies

$$\alpha^\gamma = \frac{\|\mathbf{F}(u_{k-1})\|}{\|\mathbf{F}(u_k)\|}.$$

Using $\alpha = 2$ and equation (21) we see that ideally we should have rate $= \gamma$. This, however, is almost never the case in our numerical simulations, and superlinear convergence is not achieved.

In fact, for any value of $\alpha > 1$, the choice of the maximum allowable meshsize yields

$$\gamma = \ln\left(\frac{\|\mathbf{F}(u_{k-1})\|}{\|\mathbf{F}(u_k)\|}\right) / \ln \alpha.$$

This formula is useful for monitoring whether the mesh refinement strategy agrees with the theory, but it is of no help in adjusting the meshsize, since the ratio $\|\mathbf{F}(u_{k-1})\|/\|\mathbf{F}(u_k)\|$ depends on α.

4.2 RBF Collocation without Smoothing

For comparison purposes we start off with the results of two experiments using Newton iteration without an added smoothing step. The only way of performing collocation on a hierarchy of nested grids is by using the "simple" algorithm. A return to the coarsest grid is not possible, since the residual is already zero there from the previous computations.

Example 1. The bandwidth B of the collocation matrices is kept fixed (see the second rightmost column of Table 1). In the last few iterations the matrices – even though they are of dimension 1025×1025 and larger – are very sparse (cf. the percentage of nonzero elements listed in the rightmost column). This strategy, however, has a serious (and possibly surprising) drawback: after a few steps the algorithm ceases to converge. The residual $\mathbf{F}(u)$ is still being reduced, but at an increasingly slower rate.

mesh	ℓ_2-error	rate	ℓ_2-residual	rate	B	$\% \neq 0$
5	$4.178144\,10^{-4}$		1.560313		17	100
9	$2.033295\,10^{-5}$	4.36	$5.158786\,10^{-1}$	1.60	17	100
17	$3.157809\,10^{-6}$	2.69	$2.275439\,10^{-1}$	1.18	17	80.6
33	$2.144409\,10^{-6}$	0.56	$1.117274\,10^{-1}$	1.03	17	49.3
65	$2.078499\,10^{-6}$	0.05	$5.685074\,10^{-2}$	0.97	17	27.1
129	$2.074232\,10^{-6}$	0.00	$2.965916\,10^{-2}$	0.94	17	14.2
257	$2.073959\,10^{-6}$	0.00	$1.605716\,10^{-2}$	0.89	17	7.26
513	$2.073942\,10^{-6}$	0.00	$9.244562\,10^{-3}$	0.80	17	3.67
1025	$2.073941\,10^{-6}$	0.00	$5.833538\,10^{-3}$	0.66	17	1.85
2049	$2.073941\,10^{-6}$	0.00	$4.126360\,10^{-3}$	0.50	17	0.93
4097	$2.073941\,10^{-6}$	0.00	$3.272172\,10^{-3}$	0.33	17	0.46

Table 1: Newton iteration without smoothing. Constant bandwidth.

Table 2 shows that for a larger (but fixed) bandwidth the algorithm also stops converging. However, this happens at a later stage (around iteration 9 instead of iteration 5). In this case, a decrease in the reduction of the residual is just starting in the last few iterations. A phenomenon similar to this was described in [17] in the context of quasi-interpolation with Gaussian kernels, and referred to as *approximate approximation* by the authors. The fact that the ℓ_2 errors are on the order of machine precision is no reason for concern, since the absolute value of the error is still on the order of 10^{-7}, and the ℓ_2 errors are computed via $1/n \sum |error|^2$.

mesh	ℓ_2-error	rate	ℓ_2-residual	rate	B	$\% \neq 0$
5	$9.292971\,10^{-5}$		$3.551851\,10^{-1}$		65	100
9	$1.076621\,10^{-6}$	6.43	$3.207832\,10^{-2}$	3.47	65	100
17	$1.351028\,10^{-8}$	6.32	$3.104383\,10^{-3}$	3.37	65	100
33	$1.893838\,10^{-10}$	6.16	$3.279753\,10^{-4}$	3.24	65	100
65	$3.196315\,10^{-12}$	5.89	$3.544461\,10^{-5}$	3.21	65	76.5
129	$1.005389\,10^{-13}$	4.99	$3.858650\,10^{-6}$	3.20	65	45.2
257	$1.593373\,10^{-14}$	2.66	$4.209222\,10^{-7}$	3.20	65	24.4
513	$1.165804\,10^{-14}$	0.45	$4.594098\,10^{-8}$	3.20	65	12.6
1025	$1.137442\,10^{-14}$	0.04	$5.020742\,10^{-9}$	3.19	65	6.43
2049	$1.135557\,10^{-14}$	0.00	$5.552718\,10^{-10}$	3.18	65	3.24
4097	$1.135435\,10^{-14}$	0.00	$6.800877\,10^{-11}$	3.03	65	1.63

Table 2: Newton iteration without smoothing.
Constant (but larger) bandwidth.

Example 2. In order to maintain a positive convergence rate throughout we now list the results of an experiment identical to Example 1, except that now the bandwidth of the collocation matrices is increased slightly from level to level. The bandwidth along with the sparsity of the matrices is again listed in the two rightmost columns of Table 3. The runtime of this algorithm is now considerably increased. The reward is an improvement of the final error by a factor of roughly 50, and the residual of more than

100. Even better results can be obtained by increasing the bandwidth even more rapidly (see Table 13).

mesh	ℓ_2-error	rate	ℓ_2-residual	rate	B	$\% \neq 0$
5	$4.178144\,10^{-4}$		1.560313		17	100
9	$1.800810\,10^{-5}$	4.54	$4.723374\,10^{-1}$	1.72	17	100
17	$1.439388\,10^{-6}$	3.65	$1.653630\,10^{-1}$	1.51	19	85.5
33	$2.884972\,10^{-7}$	2.32	$5.347573\,10^{-2}$	1.63	21	57.6
65	$1.361248\,10^{-7}$	1.08	$1.405820\,10^{-2}$	1.93	25	37.2
129	$9.618512\,10^{-8}$	0.50	$2.852467\,10^{-3}$	2.30	31	23.9
257	$7.920015\,10^{-8}$	0.28	$4.654513\,10^{-4}$	2.62	41	16.0
513	$6.901057\,10^{-8}$	0.20	$9.214398\,10^{-5}$	2.34	55	10.8
1025	$6.074316\,10^{-8}$	0.18	$4.530199\,10^{-5}$	1.02	79	7.75
2049	$5.201413\,10^{-8}$	0.22	$3.585907\,10^{-5}$	0.34	113	5.53
4097	$4.084314\,10^{-8}$	0.35	$2.798865\,10^{-5}$	0.36	171	4.18

Table 3: Newton iteration without smoothing. Increasing bandwidth.

4.3 The "Simple" Algorithm

In the next set of experiments we add a smoothing of the Newton update v based on biharmonic time stepping as indicated in (16) before moving on to the next finer grid as prescribed by (18).

Example 3. Everything here is the same as in Example 1 (with bandwidth $B = 17$), except for the smoothing for which we define the ratio $r = 1/10$, and use 9 time steps at every level. The fact that the meshsize is decreasing from one iteration to the next results in the smoothing being more and more localized, i.e., the smoothing is approaching the identity as required in the general theory of the Hörmander type smoothing (see (3a)). The choice $r = 1/10$ results in a "stencil" which includes negative coefficients. This is also consistent with the interpretation of Hörmander smoothing via convolution where the Fourier transform of the smoothing kernel needs to be C_0^∞ and identically equal to one in a neighborhood of the origin – giving rising to an oscillating smoothing kernel (see [7]). We observe that the smoothing in this case has a negative impact on the performance of the algorithm. The original algorithm performs so poorly that the smoothing – which will always start off slower – can not catch up. The slight increase in the errors here is an artifact attributable to the fact that the error at level k is computed on $\mathcal{X}_{k+1}$. The increase in the residual at the beginning is real.

mesh	ℓ_2-error	rate	ℓ_2 residual	rate
5	$8.536308\,10^{-4}$		$3.091776\,10^{-1}$	
9	$1.494368\,10^{-4}$	2.51	$8.660728\,10^{-1}$	−1.49
17	$1.462289\,10^{-5}$	3.35	$8.551890\,10^{-1}$	0.02
33	$3.489102\,10^{-6}$	2.07	$4.265696\,10^{-1}$	1.00
65	$2.959211\,10^{-6}$	0.24	$3.307111\,10^{-1}$	0.37
129	$2.938334\,10^{-6}$	0.01	$2.728114\,10^{-1}$	0.28
257	$2.941996\,10^{-6}$	−0.00	$2.168440\,10^{-1}$	0.33
513	$2.944737\,10^{-6}$	−0.00	$1.734825\,10^{-1}$	0.32
1025	$2.946166\,10^{-6}$	−0.00	$1.395395\,10^{-1}$	0.31
2049	$2.946884\,10^{-6}$	−0.00	$1.125743\,10^{-1}$	0.31
4097	$2.947244\,10^{-6}$	−0.00	$9.123804\,10^{-2}$	0.30

Table 4: Newton iteration with biharmonic smoothing.
Constant bandwidth.

Example 4. In this test we add the same smoothing as in Example 3 to the algorithm of Example 2, i.e., we use an increasing bandwidth coupled with biharmonic time stepping. In Table 5 we can clearly observe some of the benefits of the smoothing. After a slow start, beginning with the fifth iteration, the error with smoothing remains lower than without smoothing (cf. Table 3). By the final iteration the error has improved by a factor of about 570. The residual, however, is about 40 times larger. It should be pointed out, though, that the rate at which the residual is improving is slowing down towards the end in Example 2, whereas it is increasing for Example 4.

mesh	ℓ_2-error	rate	ℓ_2 residual	rate
5	$8.536308\,10^{-4}$		$3.091776\,10^{-1}$	
9	$1.532232\,10^{-4}$	2.48	$8.629300\,10^{-1}$	−1.48
17	$1.022748\,10^{-5}$	3.91	$8.206835\,10^{-1}$	0.07
33	$3.725648\,10^{-7}$	4.78	$3.508117\,10^{-1}$	1.23
65	$1.601594\,10^{-8}$	4.54	$2.055907\,10^{-1}$	0.77
129	$1.126580\,10^{-9}$	3.83	$1.150107\,10^{-1}$	0.84
257	$2.308002\,10^{-10}$	2.29	$5.533993\,10^{-2}$	1.06
513	$1.320186\,10^{-10}$	0.81	$2.374892\,10^{-2}$	1.22
1025	$1.057439\,10^{-10}$	0.32	$9.300114\,10^{-3}$	1.35
2049	$8.885222\,10^{-11}$	0.25	$3.426087\,10^{-3}$	1.44
4097	$6.959743\,10^{-11}$	0.35	$1.226064\,10^{-3}$	1.48

Table 5: Newton iteration with biharmonic smoothing.
Increasing bandwidth.

4.4 Algorithm 2.2

As already mentioned in Section 4.2, the only way to do collocation without smoothing on a hierarchy of meshes is to proceed from the coarsest to the finest grid in one sweep. If a smoothing step is added, then it is possible to return to the coarsest grid since in this case the residual has been "smeared out" by the smoothing. It will be evident from the remaining examples, which all make use of this fact in some way or other, that returning to the coarsest grid is extremely beneficial.

We start out with an application of Algorithm 2.2. This is only a slight modification of the "simple" algorithm. There is no smoothing during the "inner" iterations, but before returning to the coarsest grid one smoothing operation is added.

Example 5. The setup for this test is as follows: we use five grids (of 5, 9, 17, 33, and 65 equally spaced points) for the inner iteration. After an approximate solution on $\mathcal{X}_5$ has been computed, we smooth the Newton update using biharmonic timestepping with $r = 1/10$. Five time steps are used throughout, and the bandwidth of the matrix is kept fixedat $B = 17$.

The benefits of returning to the coarse grid are obvious. By the end of iteration 2 the error has reached $2.6\,10^{-9}$ – a value almost 1000 times smaller than what we were able to achieve on 4097 points in Example 1. The residuals are also smaller.

iteration	ℓ_2-error	rate	ℓ_2 residual	rate
1 (65)	$2.062621\,10^{-6}$		$5.685074\,10^{-2}$	
2 (65)	$2.618147\,10^{-9}$	9.62	$3.762865\,10^{-4}$	7.24
3 (65)	$1.038725\,10^{-9}$	1.33	$2.589270\,10^{-4}$	0.54
4 (65)	$1.574223\,10^{-9}$	-0.60	$2.094356\,10^{-4}$	0.31
5 (65)	$1.702751\,10^{-9}$	-0.11	$1.780998\,10^{-4}$	0.23
6 (65)	$1.586824\,10^{-9}$	0.10	$1.573126\,10^{-4}$	0.18
7 (65)	$1.423736\,10^{-9}$	0.16	$1.433483\,10^{-4}$	0.13
8 (65)	$1.261856\,10^{-9}$	0.17	$1.338644\,10^{-4}$	0.10
9 (65)	$1.107440\,10^{-9}$	0.19	$1.273977\,10^{-4}$	0.07
10 (65)	$9.614630\,10^{-10}$	0.20	$1.230218\,10^{-4}$	0.05

Table 6: Algorithm 2.2 with biharmonic smoothing. Constant bandwidth.

This algorithm is extremely efficient in its first 2 or 3 iterations. Moreover, it is computationally very efficient since it operates on very coarse grids with matrices of size at most 65×65. Therefore we can employ basis functions with a much larger support.

Example 6. Here we use basis functions with a fixed support covering the entire interval. This leads to full collocation matrices (which is tolerable as long as they stay small in size). Since only the first few iterations are significant we restrict the entries in the next table to these. A comparable accuracy was obtained with the other algorithms only at a far higher computational cost (e.g., see the results in Table 13).

iteration	ℓ_2-error	rate	ℓ_2 residual	rate
1 (65)	$5.011204\,10^{-13}$		$6.157483\,10^{-6}$	
2 (65)	$1.246917\,10^{-16}$	12.0	$1.816168,\,10^{-7}$	5.08
3 (65)	$6.243330\,10^{-15}$	-5.65	$4.793290\,10^{-9}$	5.24
4 (65)	$7.737252\,10^{-15}$	-0.31	$3.030940\,10^{-9}$	0.66

Table 7: Algorithm 2.2 with biharmonic smoothing, 5 time steps. Full matrices.

We would also like to mention that a direct solution with the basis used here on 129 points yields an error of $1.949018\,10^{-15}$, and on 257 points $2.422648\,10^{-17}$. The strategy of iterating twice on 65 points with the added smoothing step compares favorably. Again, the extremely small ℓ_2 errors are obtained by averaging the squares of the absolute errors, and thus are no reason for concern.

Example 7. If the size of the residual is more important than the magnitude of the error of the solution, then an increase in the number of smoothing steps helps. Here we have used 500 time steps instead of 5.

iteration	ℓ_2-error	rate	ℓ_2 residual	rate
1 (65)	$5.011204\,10^{-13}$		$6.157483\,10^{-6}$	
2 (65)	$2.665058\,10^{-14}$	4.23	$1.933114\,10^{-11}$	18.3
3 (65)	$2.672686\,10^{-14}$	-0.00	$5.439174\,10^{-12}$	1.83
4 (65)	$2.673897\,10^{-14}$	-0.00	$2.566985\,10^{-12}$	1.08
5 (65)	$2.673753\,10^{-14}$	0.00	$1.503205\,10^{-12}$	0.77

Table 8: Algorithm 2.2 with biharmonic smoothing, 500 time steps. Full matrices.

4.5 Algorithm 2.3

The final group of examples illustrates the performance of Algorithm 2.3. Here we have computed all errors and residuals on a very fine grid of 8193 points. The smoothing itself is also performed on the evaluation grid. In order to keep the connection to the Hörmander smoothing, the number of time steps is taken inversely proportional to the smoothing speeds defined in equation (8). Thus, the number of time steps is determined via

$$\text{steps} = \kappa\rho^{-\theta\beta^k}, \qquad k \geq 1. \tag{22}$$

The next two examples are analogous to Examples 1 and 3, i.e., the bandwidth of the matrices is kept fixed at $B = 17$.

Example 8. First we use Laplacian time stepping with $r = 1/3$. In order to determine the number of time steps for the smoothing, the values $\kappa = 15000$, $\rho = 2.5$, $\theta = 1.2$, and $\beta = 1.3$ are used in (22).

We can see that this algorithm has the "nicest" performance. After an initial decrease in the rate of convergence which is attributable to the poor convergence of the underlying algorithm (see Tables 1 and 4) the smoothing picks up, and the resulting rate of convergence is indeed better than linear. The residuals are reduced even more effectively.

iteration	steps	ℓ_2-error	rate	ℓ_2 residual	rate
1 (9)		$2.033295\,10^{-5}$		$5.158786\,10^{-1}$	
2 (17)	3592	$2.929755\,10^{-6}$	2.79	$2.269963\,10^{-1}$	1.18
3 (33)	2339	$1.740574\,10^{-6}$	0.75	$1.067171\,10^{-1}$	1.09
4 (65)	1340	$1.116929\,10^{-6}$	0.64	$1.981615\,10^{-2}$	2.43
5 (129)	649	$5.070245\,10^{-7}$	1.14	$7.856810\,10^{-4}$	4.66
6 (257)	253	$1.746176\,10^{-7}$	1.54	$1.765247\,10^{-4}$	2.15
7 (513)	74	$5.016099\,10^{-8}$	1.80	$5.050195\,10^{-5}$	1.81
8 (1025)	15	$1.619966\,10^{-8}$	1.63	$1.628521\,10^{-5}$	1.63
9 (2049)	2	$7.846139\,10^{-9}$	1.05	$7.872489\,10^{-6}$	1.05
10 (4097)	1	$1.908264\,10^{-9}$	2.04	$1.869062\,10^{-6}$	2.07

Table 9: Algorithm 2.3 with harmonic smoothing.
Constant bandwidth.

Example 9. This experiment is identical to the previous one, except that we now use biharmonic smoothing (with $r = 1/10$), and the number of time steps are computed using $\kappa = 5000$, $\rho = 2.5$, $\theta = 1.25$, and $\beta = 1.25$. The performance of the algorithm is similar to the previous one. However, it takes longer for the beneficial effects of the smoothing to show.

iteration	steps	ℓ_2-error	rate	ℓ_2 residual	rate
1 (9)		$2.033295\,10^{-5}$		$5.158786\,10^{-1}$	
2 (17)	1693	$3.157784\,10^{-6}$	2.69	$2.275439\,10^{-1}$	1.18
3 (33)	1291	$2.144208\,10^{-6}$	0.56	$1.117274\,10^{-1}$	1.03
4 (65)	921	$2.076735\,10^{-6}$	0.05	$5.684928\,10^{-2}$	0.97
5 (129)	603	$2.062944\,10^{-6}$	0.01	$2.969758\,10^{-2}$	0.94
6 (257)	355	$2.014039\,10^{-6}$	0.03	$2.037241\,10^{-2}$	0.54
7 (513)	183	$1.806686\,10^{-6}$	0.16	$7.587786\,10^{-3}$	1.42
8 (1025)	80	$1.164152\,10^{-6}$	0.63	$1.377814\,10^{-3}$	2.46
9 (2049)	29	$2.082058\,10^{-7}$	2.48	$2.377331\,10^{-4}$	2.53
10 (4097)	8	$1.189030\,10^{-8}$	4.13	$1.304149\,10^{-5}$	4.19

Table 10: Algorithm 2.3 with biharmonic smoothing.
Constant bandwidth.

Our last two examples should be compared to Examples 2 and 4, since we are now using collocation matrices with an increasing bandwidth.

Example 10. Laplacian smoothing with $r = 1/3$ and the same strategy for the choice of time steps as in Example 8 is used. As to be expected, the errors and residuals in Table 11 are smaller than those in Table 9. The errors in Tables 5 and 11 are quite comparable, but the residuals using Algorithm 2.3 are much smaller than when employing the "simple" algorithm.

iteration	steps	ℓ_2-error	rate	ℓ_2 residual	rate
1 (9)		$1.800810\ 10^{-5}$		$4.723374\ 10^{-1}$	
2 (17)	3592	$1.306465\ 10^{-6}$	3.78	$1.650572\ 10^{-1}$	1.52
3 (33)	2339	$2.213257\ 10^{-7}$	2.56	$5.043259\ 10^{-2}$	1.71
4 (65)	1340	$6.906664\ 10^{-8}$	1.68	$1.289936\ 10^{-3}$	5.29
5 (129)	649	$2.058588\ 10^{-8}$	1.75	$8.185404\ 10^{-4}$	0.66
6 (257)	253	$5.618570\ 10^{-9}$	1.87	$2.142798\ 10^{-5}$	5.26
7 (513)	74	$1.421454\ 10^{-9}$	1.98	$2.054422\ 10^{-6}$	3.38
8 (1025)	15	$3.962393\ 10^{-10}$	1.84	$8.520311\ 10^{-7}$	1.27
9 (2049)	2	$1.625181\ 10^{-10}$	1.29	$1.382370\ 10^{-7}$	2.62
10 (4097)	1	$3.016582\ 10^{-11}$	2.43	$2.504111\ 10^{-8}$	2.46

Table 11: Algorithm 2.3 with harmonic smoothing.
Increasing bandwidth.

Example 11. For our last experiment we use biharmonic time stepping as in Example 9 coupled with an increasing bandwidth. The results are consistent with our observations made in Examples 9 and 10. The biharmonic smoothing shows its benefits later than the Laplacian smoothing. In this case the choice of time steps even results in some slightly larger errors than in Example 4. Initially Algorithm 2.3 performs better than the "simple" algorithm, then it slows down before it starts to pick up speed again towards then end. The residuals are much smaller throughout.

iteration	steps	ℓ_2-error	rate	ℓ_2 residual	rate
1 (9)		$1.800810\ 10^{-5}$		$4.723374\ 10^{-1}$	
2 (17)	1693	$1.439376\ 10^{-6}$	3.65	$1.653631\ 10^{-1}$	1.51
3 (33)	1291	$2.884700\ 10^{-7}$	2.32	$5.347577\ 10^{-2}$	1.63
4 (65)	921	$1.360072\ 10^{-7}$	1.08	$1.405797\ 10^{-2}$	1.93
5 (129)	603	$9.563849\ 10^{-8}$	0.51	$2.856400\ 10^{-3}$	2.30
6 (257)	355	$7.681854\ 10^{-8}$	0.32	$1.311373\ 10^{-3}$	1.12
7 (513)	183	$5.981426\ 10^{-8}$	0.36	$5.874247\ 10^{-4}$	1.16
8 (1025)	80	$3.344160\ 10^{-8}$	0.84	$1.211105\ 10^{-4}$	2.28
9 (2049)	29	$4.910329\ 10^{-9}$	2.77	$1.658395\ 10^{-5}$	2.87
10 (4097)	8	$2.093763\ 10^{-10}$	4.55	$1.998400\ 10^{-6}$	3.05

Table 12: Algorithm 2.3 with biharmonic smoothing.
Increasing bandwidth.

5 Conclusions and Closing Remarks

All tests in this paper were designed to illustrate the effects of smoothing within the Nash iteration framework. It should be pointed out that far more accurate results for any of the algorithms can be obtained by increasing the bandwidth of the collocation matrices even more from one level to the next. Since then the resulting errors become extremely small (approaching machine accuracy), and since the matrices become quite dense, those choices do not lend themselves to illustrate the convergence behavior of the algorithms. The following table shows the dramatic improvements that can be obtained in this manner in the case of the "simple" algorithm without any kind of smoothing. The iteration was taken to 1025 points only since the solution of the linear systems (which have only about 75% zeros) and the evaluation of the approximation and the residuals at this point takes an intolerable amount of time on our desktop PC.

mesh	ℓ_2-error	rate	ℓ_2-residual	rate	B	$\% \neq 0$
5	$4.178144\,10^{-4}$		1.560313		17	100
9	$1.609356\,10^{-5}$	4.70	$4.320554\,10^{-1}$	1.85	17	100
17	$8.022206\,10^{-7}$	4.33	$1.214734\,10^{-1}$	1.83	21	89.6
33	$4.108865\,10^{-8}$	4.29	$2.674686\,10^{-2}$	2.18	27	68.6
65	$1.673436\,10^{-9}$	4.26	$3.998848\,10^{-3}$	2.74	39	53.1
129	$3.903871\,10^{-11}$	5.42	$3.945453\,10^{-4}$	3.34	57	40.5
257	$4.762770\,10^{-13}$	6.36	$2.681349\,10^{-5}$	3.88	93	33.5
513	$3.593669\,10^{-15}$	7.05	$1.360633\,10^{-6}$	4.30	157	28.6
1025	$2.043438\,10^{-17}$	7.46	$5.626762\,10^{-8}$	4.60	283	25.9

Table 13: Newton iteration without smoothing. Increasing bandwidth.

In this paper we have presented Nash iteration as a theoretical framework for the numerical solution of partial differential equations. This Newton algorithm with postconditioning can be interpreted in many different ways, some of which we suggested in Section 2. The theoretical framework is sufficiently general to cover most traditional numerical methods for the solution of partial differential equations. Our numerical examples focussed on one specific numerical inversion, namely the use of radial basis function collocation. For this choice of method several interesting observations were made: 1) A multilevel algorithm (without smoothing) ceases to converge. However, by choosing an appropriate support size of the basis functions (=bandwidth of the collocation matrix), the size of the obtainable error can be made small enough to satisfy any practical requirements (see [17] for the discussion of a similar phenomenon using (quasi-)interpolation with Gaussian kernels). 2) An added smoothing step can sustain the convergence. 3) The addition of a smoothing step allows the formulation of nested iteration algorithms which greatly improve the quality of the approximation (when using basis functions of comparable support sizes).

References

[1] I. Babuška and A.K. Aziz. Survey lectures on the mathematical foundations of the finite element method. In: *The Mathematical Foundations of the Finite Element Method with Applications to Partial Differential Equations.* Academic Press, 1972, pp. 5–359.

[2] P.L. Butzer and K. Scherer. Jackson and Bernstein-type inequalities for families of commutative operators in Banach spaces. J. Approximation Theory 5 (1972), 308–342.

[3] G.E. Fasshauer. Solving partial differential equations by collocation with radial basis functions. In: *Surface Fitting and Multiresolution Methods* A. Le Méhauté, C. Rabut and L.L. Schumaker (eds.), Vanderbilt University Press, Nashville, TN, 1997, pp. 131–138.

[4] G.E. Fasshauer. On smoothing for multilevel approximation with radial basis functions. In: *Approximation Theory IX, Volume II: Computational Aspects.* C.K. Chui and L.L. Schumaker (eds.), Vanderbilt University Press, Nashville, TN, 1999, pp. 55–62.

[5] G.E. Fasshauer. Solving differential equations with radial basis functions: multilevel methods and smoothing. Advances in Comp. Math. 11 (1999), 139–159.

[6] G.E. Fasshauer. On the numerical solution of differential equations with radial basis functions. In: *Boundary Element Technology XIII.* C.S. Chen, C.A. Brebbia and D.W. Pepper (eds.), WIT Press, Southampton, 1999, pp. 291–300.

[7] G.E. Fasshauer and J.W. Jerome. Multistep approximation algorithms: Improved convergence rates through postconditioning with smoothing kernels. Advances in Comp. Math. 10 (1999), 1–27.

[8] M.S. Floater and A. Iske. Multistep scattered data interpolation using compactly supported radial basis functions. J. Comput. Applied Math. 73 (1996), 65–78.

[9] C. Franke and R. Schaback. Convergence orders of meshless collocation methods using radial basis functions. Advances in Comp. Math. 8 (1998), 381–399.

[10] W. Hackbusch. *Multi-Grid Methods and Applications.* Springer, 1985.

[11] L. Hörmander. The boundary problems of physical geodesy. Arch. Ration. Mech. Anal. 62 (1976), 1–52.

[12] J.W. Jerome. An adaptive Newton algorithm based on numerical inversion: regularization as postconditioner. Numer. Math. 47 (1985), 123–138.

[13] J.W. Jerome. An asymptotically linear fixed point extension of the inf-sup theory of Galerkin approximation. Numer. Funct. Anal. Optim. 16 (1995), 345–361.

[14] J.W. Jerome. *Approximation of Nonlinear Evolution Systems.* Academic Press, 1983.

[15] J.W. Jerome. *Analysis of Charge Transport: A Mathematical Study of Semiconductor Devices.* Springer, 1996.

[16] S. Kaczmarz. Approximate solution of systems of linear equations (originally published as: Angenäherte Auflösung von Systemen linearer Gleichungen, Bulletin International de l'Academie Polonaise des Sciences, Lett. A, 1937, 355–357). Int. J. Control 57 (1993), 1269–1271.

[17] V. Maz'ya and G. Schmidt. On approximate approximations using Gaussian kernels. IMA J. Numer. Anal. 16 (1996), 13–29.

[18] J. Moser. A new technique for the construction of solutions of nonlinear differential equations. Proc. Nat. Acad. Sci. 47 (1961), 1824–1831.

[19] J. Moser. A rapidly convergent iteration method and nonlinear partial differential equations I. Ann. Scuola Norm. Pisa XX (1966), 265–315.

[20] J. Nash. The imbedding problem for Riemannian manifolds. Ann. Math. 63 (1956), 20–63.

[21] L. Nirenberg. *Topics in Nonlinear Functional Analysis.* Courant Institute of Mathematical Sciences, New York University, 1973.

[22] J.T. Schwartz. On Nash's implicit function theorem. Comm. Pure Appl. Math. 13 (1960), 509–530.

[23] H. Wendland. Piecewise polynomial, positive definite and compactly supported radial functions of minimal degree. Advances in Comp. Math. 4 (1995), 389–396.

[24] H. Wendland. Numerical solution of variational problems by radial basis functions. In: *Approximation Theory IX, Volume II: Computational Aspects.* C.K. Chui and L.L. Schumaker (eds.), Vanderbilt University Press, Nashville, TN, 1999, pp. 361–368.