ITERATIVE METHODS FOR NONLINEAR ELLIPTIC EQUATIONS

LONG CHEN

CONTENTS

1. Introduction 1
1.1. Basic Iterative Methods 1
1.2. Nonlinear Elliptic Equations 3
2. Newton-type Methods 4
2.1. Convergence analysis 5
2.2. Damped Newton's method 6
2.3. Inexact Newton's method 6
2.4. Jacobian-free Newton method 7
3. Two-grid Methods 7
3.1. Coarse-Grid and Fixed point iteration 8
3.2. Coarse-grid and Newton's iteration 8
4. FAS: Nonlinear Multigrid 9
4.1. Smoother: Nonlinear Gauss-Seidel Iteration 9
4.2. Coarse grid problem 10
References 11

1. INTRODUCTION

We consider the nonlinear equation posed on $\mathcal{V}_h$,

$$\mathcal{L}(u_h) = 0, \qquad u_h \in \mathcal{V}_h. \tag{1}$$

Starting from an initial guess u_0, we seek a better approximation u_{k+1} from the current iterate u_k for $k = 0, 1, 2, \ldots$. For clarity, we omit the subscript h in the iteration.

An important case is $\mathcal{L}(v) = E'(v)$, where the equation is the Euler–Lagrange condition of the optimization problem $\min_{v\in\mathcal{V}} E(v)$.

1.1. Basic Iterative Methods.

A generic iterative method contains three steps:

(1) form the residual $r_k = -\mathcal{L}(u_k)$;
(2) compute an approximate correction $e_k = Br_k$;
(3) update $u_{k+1} = u_k + \alpha\, e_k$.

The parameter α can be absorbed into B, but we keep it explicit when needed. The operator B may be nonlinear; we first discuss the linear case.

For a linear problem $Au = f$, the ideal choice $B = A^{-1}$ follows from the fact that the error $e_k = u - u_k$ satisfies $Ae_k = r_k$. For nonlinear equations this relation does not hold:

$$\mathcal{L}(u - u_k) \neq \mathcal{L}(u) - \mathcal{L}(u_k) = r_k.$$

Date: 11/20/2013, Latest update: December 4, 2025.

By the mean value theorem,

$$\mathcal{L}(u) - \mathcal{L}(u_k) = \mathcal{L}'(\xi)\,(u - u_k),$$

for some $\xi = tu_k + (1-t)u$, $t \in (0,1)$. Hence the ideal residual equation is

$$J(\xi)e_k = r_k, \tag{2}$$

where $J(\xi) = \mathcal{L}'(\xi)$ is the Jacobian at ξ. Thus the ideal operator is $B = J(\xi)^{-1}$, which is not practical since ξ is unknown.

Different iterative methods follow from different choices of B. Several examples are listed in the following table.

Method	Nonlinear Eqn	Optimization
Gradient descent (GD)	$u_{k+1} = u_k - \alpha\,\mathcal{L}(u_k)$	$u_{k+1} = u_k - \alpha\,E'(u_k)$
Preconditioned GD	$u_{k+1} = u_k - A^{-1}\mathcal{L}(u_k)$	$u_{k+1} = u_k - A^{-1}E'(u_k)$
Newton's method	$u_{k+1} = u_k - J^{-1}(u_k)\mathcal{L}(u_k)$	$u_{k+1} = u_k - (E''(u_k))^{-1}E'(u_k)$

We use the term gradient descent for $B = \alpha I$ since it matches the classical method when $\mathcal{L} = E'$. The choice $B = A^{-1}$, for a fixed SPD operator A, can be viewed as gradient descent in the metric defined by A. This reduces the condition number and improves the convergence. For nonlinear problems, a better metric should reflect the local behavior of the function. Newton's method uses $B = J(u_k)$, which incorporates such local information through the Jacobian.

We write derivatives and equations in the calculus style. A more rigorous formulation uses the calculus of variations. Given the current approximation u_k, we perturb it as $u_k + \varepsilon e$, where $\varepsilon \in \mathbb{R}$ and $e \in H_0^1$, and take the derivative with respect to ε:

$$\frac{d}{d\varepsilon}\,\mathcal{L}(u_k + \varepsilon e)\Big|_{\varepsilon=0} = \mathcal{L}'(u_k)\,e.$$

The residual equation $\mathcal{L}'(u_k)e = r_k$, posed in $\mathcal{V}'$, is understood in the weak sense:

$$(\mathcal{L}'(u_k)e, \phi) = (r_k, \phi) \qquad \forall\, \phi \in \mathcal{V}. \tag{3}$$

Note that even if u has a nonzero Dirichlet boundary condition, the correction e in the residual equation must vanish on the Dirichlet boundary. From the viewpoint of the calculus of variations, the perturbation $u_k + \varepsilon e$ must satisfy the boundary condition for every ε, hence e must be zero on the Dirichlet boundary. Thus the residual equation is posed only on the free nodes, and the linear algebraic system is solved for those nodes.

The boundary condition of u is included when computing the residual $\mathcal{L}(u)$. For example, let `A` be the full stiffness matrix and `u` the vector of nodal values. Then the operation `(A*u)(freeNode)` carries the contribution from the boundary nodes to the interior.

A common splitting method writes the equation as $\mathcal{L}(u) = Mu + N(u)$ and updates u_{k+1} by

$$Mu_{k+1} = -N(u_k).$$

This follows from the fixed point iteration, where M is treated as a linear operator and $N(u)$ accounts for the nonlinear part.

1.2. **Nonlinear Elliptic Equations.** We focus on the finite element discretization of semi-linear elliptic equations of the form: find $u \in H_0^1(\Omega)$ such that

$$-\Delta u = f(x,u) \qquad \text{in } H^{-1}(\Omega). \tag{4}$$

In most places we drop the dependence on x in f to emphasize the nonlinearity in u. Let $A : H_0^1(\Omega) \to H^{-1}(\Omega)$ be the elliptic operator defined by $Au = -\Delta u$, and set $\|v\|_A^2 = \langle Av, v\rangle$. Then (4) can be written as

$$\mathcal{L}(u) := Au - f(u) = 0.$$

The presence of the Laplacian introduces additional structure, especially the separation of scales among H_0^1, L^2, and H^{-1}, and gives a natural splitting $M = A$ and $N(u) = -f(u)$.

Let $\mathcal{V}_h \subset H_0^1(\Omega)$ be a linear finite element space associated with a quasi-uniform mesh $\mathcal{T}_h$. The nonlinear Galerkin method seeks $u_h \in \mathcal{V}_h$ such that

$$Au_h = f(u_h) \qquad \text{in } \mathcal{V}_h'. \tag{5}$$

We are interested in iterative methods to compute u_h or an accurate approximation of it. We drop the subscript h when the mesh size does not affect the discussion.

We assume f is sufficiently smooth; for example, f is Lipschitz with respect to u:

$$\|f(u) - f(v)\|_{-1} \le L\,\|u - v\|_A.$$

We also assume the existence and local uniqueness of the solution to (4). It is known (see [14] and the references therein) that, if h is small enough, the discrete problem (5) has a locally unique solution u_h that satisfies the error estimate

$$\|u - u_h\|_{0,p} + h\,\|u - u_h\|_{1,p} \lesssim h^2\,\|u\|_{2,p}, \qquad \text{for all } 2 \le p < \infty. \tag{6}$$

For $\mathcal{L}(u) = Au - f(u)$, the Jacobian is $\mathcal{L}'(u_k) = A - f'(u_k)$, and equation (3) becomes

$$(Ae, \phi) + (f'(u_k)e, \phi) = (r_k, \phi) \qquad \forall\, \phi \in \mathbb{V}.$$

In addition to the stiffness matrix, one must assemble a weighted mass matrix corresponding to the weighted L^2 inner product $(f'(u_k)e, \phi)$, where the coefficient $f'(u_k)$ varies over the domain. For linear finite elements, this can be approximated by the three–vertex quadrature rule

$$\int_\tau g(x)\,dx \approx \frac{1}{3}\sum_{i=1}^{3} g(x_i)\,|\tau|.$$

With this quadrature, the mass matrix becomes diagonal, $M = \operatorname{diag}(m_1, \ldots, m_N)$, a procedure known as mass lumping. Taking $\phi = \phi_i$, the basis function associated with vertex i, we obtain at each vertex

$$\sum_{j=1}^{N} a_{ij} e_j \;+\; m_i\, f'(u_k)_i\, e_i \;=\; m_i\, r_{k,i}. \tag{7}$$

For two–dimensional uniform grids with Dirichlet boundary conditions, $m_i = h^2$ and (7) can be interpreted as a rescaled finite difference discretization of the Jacobian equation at each vertex.

Algorithm 1.1 (Fixed point / Picard iteration)**.** *We choose $B = A^{-1}$ and solve*

$$Au_{k+1} = f(u_k). \tag{8}$$

If the sequence $\{u_k\}$ converges, the limit is the solution u. From

$$\|u - u_{k+1}\|_1 = \|f(u) - f(u_k)\|_{-1} \leq L \|u - u_k\|_A,$$

the iteration converges when $L < 1$, which restricts its applicability.

Algorithm 1.2 (Gradient flow)**.** *We choose $B = \alpha I$ and update*

$$u_{k+1} = u_k + \alpha (f(u_k) - Au_k). \tag{9}$$

This iteration can be viewed as the forward Euler method applied to the nonlinear parabolic equation

$$u_t + Au = f(u), \tag{10}$$

and seeks its steady state.

Each iteration of (11) is cheap since it only requires applying A rather than A^{-1}. However, the method is not recommended for large problems because the step size α must be small (of order h^2 even for the linear case). As a result, many iterations are needed to reach the equilibrium, and the total cost becomes large.

More accurate or stable time-stepping methods, such as Runge–Kutta schemes or implicit integrators, may be used to solve (10) with larger time steps. For example, an implicit-explicit (IMEX) discretization of (10) is

$$u_{k+1} = u_k + \alpha (f(u_k) - Au_{k+1}), \quad u_{k+1} = (I + \alpha A)^{-1}(u_k + \alpha f(u_k)). \tag{11}$$

Since we are interested in the equilibrium rather than the full trajectory, one may also rescale the pseudo-time variable or modify the underlying PDE to speed up convergence. These ideas will be discussed in a later lecture note.

Algorithm 1.3 (Gradient flow in the H^{-1} inner product)**.** *We choose $B = \alpha A^{-1}$*

$$u_{k+1} = u_k + \alpha A^{-1}\big(f(u_k) - Au_k\big) = (1 - \alpha)\, u_k + \alpha\, A^{-1} f(u_k). \tag{12}$$

This iteration can be interpreted as a gradient flow

$$u_t = f(u) - Au,$$

but posed in the H^{-1} inner product, represented by A^{-1}. The second expression in (12) shows that it is also a relaxed fixed point method: a weighted average between the current iterate and the Picard update $A^{-1} f(u_k)$.

Exercise 1.4. Write the error equation and determine a range of α such that the iteration (12) is a contraction. Conclude that for such α, the true solution u is a fixed point of (12).

2. Newton-type Methods

In this section, we consider Newton's method and its variants for solving nonlinear elliptic equation. The Newton's method is also refereed as Newton-Raphson method; see [1] on 'who is Raphson'?

Algorithm 2.1 (Newton's method)**.** *We choose $B = J_k := A - f'(u_k)$ and update*

$$u_{k+1} = u_k + J_k^{-1}\big(f(u_k) - Au_k\big).$$

2.1. **Convergence analysis.** We first analyze the convergence of Newton's method for the nonlinear elliptic equation. At iteration k, Newton's method computes u_{k+1} from the linear approximation of $\mathcal{L}(u) = 0$ at u_k:

$$\mathcal{L}(u_k) + \mathcal{L}'(u_k)(u_{k+1} - u_k) = 0.$$

For $\mathcal{L}(u) = Au - f(u)$, we write the Newton update and the Taylor expansion of the exact solution u:

$$\begin{aligned} Au_{k+1} &= f(u_k) + f'(u_k)(u_{k+1} - u_k), \\ Au &= f(u_k) + f'(u_k)(u - u_k) + \tfrac{1}{2} f''(\xi)\,(u - u_k)^2, \end{aligned}$$

where $\xi = t\,u_k + (1-t)u$ for some $t \in (0,1)$. Subtracting the two equalities gives

$$J_k(u - u_{k+1}) = \tfrac{1}{2} f''(\xi)\,(u - u_k)^2,$$

where $J_k = A - f'(u_k)$. Assume $J_k : H_0^1(\Omega) \to H^{-1}(\Omega)$ is an isomorphism and f'' is uniformly bounded. Then

$$\|u - u_{k+1}\|_1 \le C_k\, \|(u - u_k)^2\|_{-1}.$$

Notice that $\|(u-u_k)^2\| \neq \|u - u_k\|^2$. We estimate its H^{-1} norm using Hölder's inequality and the Sobolev embedding $H^1(\Omega) \hookrightarrow L^6(\Omega)$ for $d \le 3$:

$$\left|((u-u_k)^2, v)\right| \le \|(u-u_k)^2\|_{L^{6/5}} \|v\|_{L^6} = \|u - u_k\|_{L^{12/5}}^2 \|v\|_{L^6} \lesssim \|u - u_k\|_1^2 \|v\|_1.$$

Hence

$$\|(u-u_k)^2\|_{-1} \lesssim \|u - u_k\|_1^2,$$

and therefore

$$\|u - u_{k+1}\|_1 \le C\,\|u - u_k\|_1^2 \le C^{2(k+1)} \|u - u_0\|_1^{2(k+1)}. \tag{13}$$

These estimates reflect the scaling between Sobolev spaces and do not appear in the standard convergence analysis for Newton's method in optimization methods.

Thus, if the initial guess is sufficiently close to the true solution u, Newton's method converges quadratically. If the initial error is not small, the method may diverge. For example, with $C = 10$ and $\|u - u_0\|_1 = 0.1$, the bound (13) is always 0.1; small perturbations around this threshold can lead either to rapid convergence or divergence.

Theorem 2.2 (Local quadratic convergence of Newton's method)**.** *Let u solve $Au = f(u)$. Assume:*

(1) $J_k = A - f'(u_k)$ is uniformly elliptic for all iterates in a neighborhood of u;
(2) f' is Lipschitz (equivalently f'' is uniformly bounded).

Then there exists $\delta > 0$ such that if $\|u_0 - u\|_1 < \delta$, Newton's method is well-defined and satisfies

$$\|u - u_{k+1}\|_1 \;\le\; C\,\|u - u_k\|_1^2.$$

Hence the convergence is quadratic.

The assumption that the initial guess is sufficiently close to u is essential. This is usually referred to as *local* quadratic convergence. For a general formulation and proof, see the classical Kantorovich theorem [6].

The limitations of Newton's method include:

(1) It requires a good initial guess and is sensitive to the choice of the starting point.
(2) Each iteration involves solving a large linear system, which is expensive.
(3) It needs explicit information about the Jacobian matrix, so the method is not easily transferable across different problems.

(4) It requires sufficient smoothness of the nonlinear term; if the nonlinearity is not smooth, the Newton step may be large or unstable.

To address some of these difficulties, we consider the following variants of Newton's method:

(1) Damped Newton: enforce a decrease of the residual at each step.
(2) Inexact Newton: solve the Jacobian system only approximately.
(3) Jacobian-free Newton: approximate the Jacobian numerically.

The semi-smooth Newton method (SSN) for non-smooth problems is not included here, as it requires subdifferential operators and would need a more extensive introduction.

2.2. **Damped Newton's method.** In Newton's method, the update can be too large in the search direction e; see the figure below for a simple example. To control the step, we introduce a step size α and enforce a decrease of the residual in a chosen norm. The optimal α may be found by a line search. Here we adopt a simple bisection strategy, taking $\alpha = 1, 0.5, 0.25, \ldots$. For more advanced line search techniques, we refer to [5, 10].

Algorithm: $u = \mathrm{update}(u_{\mathrm{old}}, e)$
$\alpha = 1;\ r_{\mathrm{old}} = \|\mathcal{L}(u_{\mathrm{old}})\|;\ r = r_{\mathrm{old}} + 1;\ k = 1;$
while $r > r_{\mathrm{old}}$ *and* $k < maxIt$ **do**
  $u = u_{\mathrm{old}} + \alpha\, e;$
  $r = \|\mathcal{L}(u)\|;$
  $\alpha = \alpha/2;$
  $k = k + 1;$
end

Algorithm 1: Damped Newton Update

2.3. **Inexact Newton's method.** In each step of Newton's method, the most time consuming part is solving the Jacobian equation $Je = r$. Since J is elliptic, the most efficient approach is to use iterative solvers with multilevel preconditioners. We refer to this solver as the *inner iteration*, and we allow a dynamic tolerance that changes with each outer Newton step.

Algorithm: $e = \mathrm{inexact}(J, r, B, \theta)$
$e^0 = 0;\ r_{\mathrm{in}} = r;$
while $\|r_{\mathrm{in}}\| > \theta\|r\|$ **do**
  $e^i = e^{i-1} + Br;$
  $r_{\mathrm{in}} = r - Je^i;$
end

Algorithm 2: Inner iteration of inexact Newton method

In the inner iteration, the Jacobian equation is solved only approximately by an operator B, with the iteration stopped once a prescribed tolerance is reached. This tolerance may depend on the progress of the outer Newton iteration through a parameter θ. During the early Newton steps, there is no benefit in solving the Jacobian equation to high accuracy, and a practical choice is to use a decreasing sequence such as $\theta = 10^{-2}, 10^{-3}, \ldots, 10^{-6}$ until the smallest tolerance is reached. We refer to [6] for a detailed discussion.

2.4. **Jacobian-free Newton method.** In Krylov methods for linear systems, the Jacobian matrix does not need to be formed explicitly; only the action Jv on a given vector v is required. A first–order finite difference approximation gives

$$J_u v \approx \frac{\mathcal{L}(u+\epsilon v)-\mathcal{L}(u)}{\epsilon} = Av - \frac{f(u+\epsilon v)-f(u)}{\epsilon}.$$

The truncation error is $\mathcal{O}(\epsilon)$. The choice of ϵ is sensitive to the scaling of u and v: if ϵ is too large, the derivative approximation is poor; if it is too small, the finite difference is affected by floating–point roundoff, since we divide by a small number. A practical choice is

$$\epsilon = \sqrt{\epsilon_{\text{mach}}}\left(\frac{1}{N\|v\|_{l_2}}\sum_{i=1}^{N}|u_i|+1\right),$$

where ϵ_{mach} is the machine roundoff. See [7] for further discussion. We summarize these variants in the following algorithm. For Jacobian–free Newton, replace the matrix J by a subroutine `J(u,v)` that evaluates the finite difference approximation of Jv.

Algorithm: $u = \text{NewtonMethod}(u_0, \text{tol})$
$u = u_0; r_0 = f(u_0) - Au_0; r = r_0;$
while $\|r\|/\|r_0\| > \text{tol}$ **do**
  $r = f(u) - Au;$
  $J = Au - f'(u);$
  $e = \text{inexact}(J, r, B, \theta);$
  $u_0 = u;$
  $u = \text{update}(u_0, e);$
  $\theta = \theta/10;$
end

3. Two-grid Methods

The two–grid method of Xu [14] computes a coarse approximation, where the nonlinearity is resolved, and uses it as the initial guess on the fine grid, where only one linear iteration is needed because of scale separation. We follow [14] and refer to [9, 12, 13] for further applications.

Let $\mathcal{V}_h$ and $\mathcal{V}_H$ be finite element spaces on quasi-uniform meshes $\mathcal{T}_h$ and $\mathcal{T}_H$ with $H > h$. Assume we have computed $u_H \in \mathcal{V}_H$ from

$$Au_H = f(u_H) \quad \text{in } \mathcal{V}_H' \tag{14}$$

by a coarse nonlinear solve (for example, gradient flow or Newton). The space $\mathcal{V}_H$ can be chosen small, so the cost of computing u_H is negligible.

The fine-grid solution $u_h \in \mathcal{V}_h$, which satisfies

$$Au_h = f(u_h) \quad \text{in } \mathcal{V}_h'. \tag{15}$$

The coarse and fine solutions satisfy the estimates in (6). In particular,

$$\|u-u_H\|_p \lesssim H^2\|u\|_{2,p}, \qquad \|u-u_h\|_1 \lesssim h\,\|u\|_2. \tag{16}$$

We then seek an approximation $u^h \in \mathcal{V}_h$ which is differrent with u_h but still have the optimal first order approximation.

3.1. **Coarse-Grid and Fixed point iteration.** We first analyze a simple two–grid strategy in which the nonlinear problem is solved only on the coarse grid and the fine–grid solution is obtained by a single fixed point update.

Algorithm 3.1 (Coarse-Grid + One Fixed Point Iteration)**.** *Compute $u^h \in \mathcal{V}_h$ such that*

$$Au^h = f(u_H) \quad \text{in } \mathcal{V}_h'. \tag{17}$$

We present the error estimate below. The key point is that the fine–grid error is measured in the H^1–norm, whereas the perturbation induced by the coarse–grid approximation is measured in the L^2–norm, which is of one order higher.

Theorem 3.2 (Coarse–Grid + One Fixed Point Iteration)**.** *Let u_H solve (14) and u_h solve (15), and u^h is obtained by (17). Assume f is Lipschitz continuous with respect to u in the L^2–norm. Then*

$$\|u_h - u^h\|_1 \lesssim H^2, \quad \|u - u^h\|_1 \lesssim h + H^2.$$

Proof. Subtracting (17) from the fine–grid equation $Au_h = f(u_h)$ gives the error equation

$$A(u_h - u^h) = f(u_h) - f(u_H) \quad \text{in } \mathcal{V}_h'. \tag{18}$$

By ellipticity of A and the Lipschitz continuity of f,

$$\|u_h - u^h\|_1 \lesssim \|f(u_h) - f(u_H)\|_{-1} \leq L\|u_h - u_H\|.$$

Using the nonlinear Galerkin estimate (6),

$$\|u_h - u_H\| \leq \|u - u_h\| + \|u - u_H\| \lesssim H^2,$$

which proves

$$\|u_h - u^h\|_1 \lesssim H^2.$$

Finally, the triangle inequality gives

$$\|u - u^h\|_1 \leq \|u - u_h\|_1 + \|u_h - u^h\|_1 \lesssim h + H^2.$$

□

Hence, to achieve an optimal H^1–error on the fine grid, it suffices to select $H = h^{1/2}$, which is much larger than h. For instance, in two dimensions, if $h = 2^{-10}$, then $\dim \mathcal{V}_h \approx 2^{20} \approx 10^6$, whereas $H = 2^{-5}$ yields $\dim \mathcal{V}_H \approx 2^{10} = 10^3$. Thus the nonlinear iterations are carried out on a space that is orders of magnitude smaller.

3.2. **Coarse-grid and Newton's iteration.** Using Newton's method on the fine grid yields a sharper correction than the fixed point update.

Algorithm 3.3 (Coarse–Grid + One Newton iteration)**.** *Compute $e^h \in \mathcal{V}_h$ from*

$$J(u_H)\, e^h = f(u_H) - Au_H \qquad \text{in } \mathcal{V}_h', \tag{19}$$

and set

$$u^h = u_H + e^h.$$

Because of the quadratic convergence of Newton's method, the error ratio improves to $h + H^4$.

Theorem 3.4 (Coarse–Grid + One Newton iteration)**.** *Let u_H solve (14), let u_h solve (15), and let u^h be obtained by Algorithm 3.3. Assume $J(u_H)$ is elliptic and f'' is uniformly bounded. Then*

$$\|u_h - u^h\|_1 \lesssim H^4, \qquad \|u - u^h\|_1 \lesssim h + H^4.$$

Proof. We compare the linearization at u_H with the Taylor expansion of u_h. The Newton correction satisfies

$$Au^h = f(u_H) + f'(u_H)(u^h - u_H),$$

while Taylor expansion of f at u_H gives

$$Au_h = f(u_H) + f'(u_H)(u_h - u_H) + \tfrac{1}{2} f''(\xi)(u_h - u_H)^2,$$

where ξ lies on the segment between u_H and u_h. Subtracting yields

$$J(u_H)(u_h - u^h) = \tfrac{1}{2} f''(\xi)\,(u_h - u_H)^2 \qquad \text{in } \mathcal{V}_h'. \tag{20}$$

Since $J(u_H) : H_0^1(\Omega) \to H^{-1}(\Omega)$ is an isomorphism and f'' is uniformly bounded, the same argument used in the convergence analysis of Newton's method and the estimate (16) gives

$$\|u_h - u^h\|_1 \;\le\; C\,\|(u_h - u_H)^2\|_{-1} \;\lesssim\; \|u_h - u_H\|_{L^{12/5}}^2 \;\lesssim\; H^4.$$

Using the triangle inequality and (16),

$$\|u - u^h\|_1 \;\le\; \|u - u_h\|_1 + \|u_h - u^h\|_1 \;\lesssim\; h + H^4.$$

□

In two dimensions, if $h = 2^{-10}$, then $\dim \mathcal{V}_h \approx 2^{20} \approx 10^6$, while choosing $H = h^{1/4} = 2^{-5}$ gives $\dim \mathcal{V}_H \approx 2^5 = 32$. Thus the cost of nonlinear iterations is negligible, yet one Newton correction still achieves optimal accuracy on the fine grid.

The main constraint of the two–grid approach is that the coarse grid must capture the essential nonlinearity so that the error estimate (16) remains valid. Hence H cannot be too large. If the coarse problem is chosen overly coarse, the nonlinear features of the solution are not resolved and the fine-grid correction deteriorates.

4. FAS: Nonlinear Multigrid

In this section we introduce the nonlinear multigrid method FAS developed by Brandt [2] for solving the nonlinear equation

$$\mathcal{L}(u) = 0, \quad u \in \mathcal{V}_h.$$

Since the iteration is defined on a fixed space, we will skip the subscript h. Recall that the success of multigrid methods relies on two ingredients:

- High-frequency errors are efficiently damped by smoothers.
- Low-frequency errors can be approximated well on a coarse grid.

We first introduce an effective smoother for nonlinear problems: the nonlinear Gauss–Seidel iteration, and then discuss the most subtle part of the nonlinear multigrid method—the perturbed coarse-grid equation used to construct the coarse-grid correction.

4.1. Smoother: Nonlinear Gauss-Seidel Iteration.

Recall that the Gauss–Seidel iteration for a linear equation $Au = b$ can be interpreted as a subspace correction method based on the canonical decomposition $\mathbb{V} = \sum_{i=1}^N \operatorname{span}\{\phi_i\}$. That is, we solve the original equation only in the one–dimensional subspace spanned by a basis function $\mathbb{V}_i = \operatorname{span}\{\phi_i\}$.

For nonlinear problems, suppose v is the current approximation. We seek a correction in the subspace $\mathbb{V}_i$ only, that is, we solve the nonlinear one–dimensional problem: find $\alpha \in \mathbb{R}$ such that

$$\mathcal{L}(v + \alpha\phi_i) = 0. \tag{21}$$

One nonlinear Gauss–Seidel (NGS) sweep loops over all degrees of freedom, $i = 1, \dots, N$, and solves N such one–dimensional nonlinear problems.

When the nonlinear equation $\mathcal{L}(u) = 0$ is the first order optimality condition of an optimization problem $\min_u E(u)$, the one–dimensional problem (21) in $\mathbb{V}_i$ is equivalent to restricting the optimization to this subspace,

$$\min_{v_i \in \mathbb{V}_i} E(v + v_i) = \min_{\alpha \in \mathbb{R}} E(v + \alpha \phi_i). \tag{22}$$

From the viewpoint of space decomposition, $\mathbb{V}_i$ can be any low–dimensional subspace, not necessarily one dimensional.

The nonlinear problem (21) is solved by Newton's method. Given the current approximation α_k, one Newton step follows from the linearization at α_k:

$$\mathcal{L}(v + \alpha_k \phi_i) + \mathcal{L}'(v + \alpha_k \phi_i)\, e_\alpha \phi_i = 0. \tag{23}$$

The corresponding weak form is

$$\big(\mathcal{L}'(v + \alpha_k \phi_i)\, e_\alpha \phi_i, \phi_i\big) = -\big(\mathcal{L}(v + \alpha_k \phi_i), \phi_i\big), \tag{24}$$

and we update $\alpha_{k+1} = \alpha_k + e_\alpha$. The residual and Jacobian must be updated when moving from i to $i + 1$, but this update is local and inexpensive in standard finite element or finite difference settings.

4.2. **Coarse grid problem.** Given the current approximation v, we solve a coarse problem to obtain a new approximation u. Introduce a coarse space $\mathcal{V}_H \subset \mathcal{V}$. Denote by I_H the inclusion (prolongation) from $\mathcal{V}_H$ to $\mathcal{V}$. Its transpose is the restriction $Q_H = I_H^{\intercal} : \mathcal{V}' \to \mathcal{V}_H'$, which restricts the residual to the coarse space, $r_H = Q_H r$. Introduce a projection $\mathcal{I}_H : \mathcal{V} \to \mathcal{V}_H$, typically the nodal interpolation, and set $v_H = \mathcal{I}_H v$ as the coarse approximation of v.

In the optimization setting (22), where $\mathcal{L}(v) = E'(v)$, the residual is $r = -E'(v)$ and $r_H = Q_H r$ is its coarse projection. We consider the τ-perturbed nonlinear coarse problem: find $u_H \in \mathcal{V}_H$ such that

$$E_H'(u_H) = \tau_H, \qquad \tau_H = E_H'(v_H) + r_H. \tag{25}$$

Equivalently,

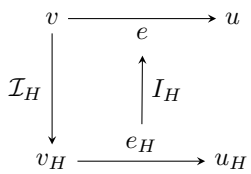
$$\min_{w \in \mathcal{V}_H} E_H(w) - (w, \tau_H).$$

For a general nonlinear operator, the corresponding coarse problem is

$$\mathcal{L}_H(u_H) = \tau_H, \qquad \tau_H = \mathcal{L}_H(\mathcal{I}_H v) - Q_H \mathcal{L}(v). \tag{26}$$

Once u_H is computed, the coarse correction and the fine-grid update are

$$e_H = u_H - v_H, \qquad u = v + I_H e_H.$$



This approach is called the *full approximation scheme* (FAS) because the coarse problem solves for the full approximation u_H rather than for a correction.

We present several remarks.

Remark 4.1. We compare the two–grid method and emphasize the role of solving the perturbed coarse equation $\mathcal{L}_H(u_H) = \tau_H$. If instead one solves $\mathcal{L}_H(u_H) = 0$ and uses u_H as an initial guess on the fine grid, the method reduces to the two–grid approach in Section 3, which requires the coarse approximation to satisfy $\|u - u_H\| \lesssim H^2$. In contrast, FAS is an iterative method in which the coarse problem $\mathcal{L}_H(u_H) = \tau_H$ is solved at every step (with updated residual), thereby ensuring that each coarse solve reflects the current residual on the fine grid.

Remark 4.2. For an optimization problem, the coarse equation can be written as

$$E'_H(u_H) - E'_H(v_H) = -\, r_H,$$

which resembles a mirror descent method. Therefore the coarse-grid problem can be generalized to an approximated problem (not necessiarly on a coarse grid) when the geometric mesh is not available. The approximated function E_H severes as a nonlinear preconditioner and its derivative E'_H is called mirror map.

Remark 4.3. A Newton step on the coarse grid computes

$$\nabla^2 E_H(v_H)\, e_H = r_H,$$

followed by the update

$$u = v + I_H e_H = v - I_H \left(\nabla^2 E_H(v_H)\right)^{-1} Q_H E'(v).$$

In effect, the full Hessian $\nabla^2 E(v)$ in Newton's method is replaced by the coarse Hessian $\nabla^2 E_H(v_H) \approx \nabla^2 E(v)$. This reduces the cost of forming and inverting the Hessian while retaining the main nonlinear structure.

Algorithm 4.4 (Two-Level Full Approximation Scheme (FAS))**.** $u = \mathrm{FAS}(v)$.

(1) Pre-smoothing: apply m steps of nonlinear Gauss–Seidel, $v = \mathrm{NGS}(v, m)$.
(2) Compute the residual: $r = -\mathcal{L}(v)$.
(3) Restrict to the coarse grid: $r_H = Q_H r$ *and* $v_H = \mathcal{I}_H v$.
(4) Solve the coarse problem:

$$\mathcal{L}_H(u_H) = \mathcal{L}_H(v_H) + r_H.$$

(5) Compute the coarse correction: $e_H = u_H - v_H$.
(6) Prolongate and update:

$$u = v + I_H e_H.$$

(7) Post-smoothing: apply m steps of nonlinear Gauss–Seidel, $u = \mathrm{NGS}(u, m)$.

The nonlinear coarse-grid problem $\mathcal{L}_H(u_H) = \mathcal{L}_H(v_H) + r_H$ can be solved recursively, which gives a standard V-cycle. In the nonlinear Gauss–Seidel smoother, there is no need to solve each local nonlinear problem exactly; one gradient or Newton update is sufficient. For convergence analysis of FAS in convex optimization, see [4].

REFERENCES

[1] N. Bićanić and K. H. Johnson. Who was "Raphson"? *Internat. J. Numer. Methods Engrg.*, 14(1):148–152, 1979.
[2] Achi Brandt. Multi-level adaptive solutions to boundary-value problems. *Math. Comp.*, 31:333–390, 1977.
[3] W. L. Briggs, V. E. Henson, and S. F. McCormick. *A Multigrid Tutorial*. SIAM Books, Philadelphia, 2000. Second edition.
[4] L. Chen, X. Hu, and S. Wise. Convergence Analysis of the Fast Subspace Descent Methods for Convex Optimization Problems. *Mathematics of Computation*. April 7, 2020.
[5] C. Kelley. *Iterative methods for linear and nonlinear equations*. Society for Industrial Mathematics, 1995.

[6] P. Knabner and L. Angermann. *Numerical Methods for Elliptic and Parabolic Partial Differential Equations*. Springer, 2003.
[7] D. Knoll and D. Keyes. Jacobian-free Newton–Krylov methods: a survey of approaches and applications. *J. Comput. Phys.*, 193(2):357–397, 2004.
[8] C. Liu and S. Atluri. A novel time integration method for solving a large system of non-linear algebraic equations. *CMES: Computer Modeling in Engineering and Sciences*, 31:71–83, 2008.
[9] M. Marion and J. Xu. Error estimates on a new nonlinear galerkin method based on two-grid finite elements. *SIAM J. Numer. Anal.*, 32(4):1170–1184, 1995.
[10] J. Nocedal and S. Wright. *Numerical Optimization*. Springer, New York, 1999.
[11] X. Tai and J. Xu. Global convergence of subspace correction methods for convex optimization problems. *Math. Comp.*, 71(237):105–124, 2002.
[12] J. Xu. A new class of iterative methods for nonselfadjoint or indefinite problems. *SIAM J. Numer. Anal.*, 29:303–319, 1992.
[13] J. Xu. A novel two-grid method for semilinear elliptic equations. *SIAM J. Sci. Comput.*, 15(1):231–237, 1994.
[14] J. Xu. Two-grid discretization techniques for linear and nonlinear PDEs. *SIAM J. Numer. Anal.*, 33(5):1759–1777, 1996.