Chapter 4

Parameter Field Inversion

4.1 Synopsis

We are interested in recovering system parameter fields from given input and output data, i.e., a *parameter field inversion*. As a representative example, we consider an *inverse medium problem*, in which we seek the spatial distribution of material properties, such as the Lamé parameters, mass density, thermal conductivity, or dielectric permittivity, of a physical system from boundary or interior measurements of its response.

Problems of this type arise across a wide range of disciplines. In geophysics, seismic waveform inversion recovers the elastic moduli and density of subsurface formations from surface recordings of reflected and refracted waves. In medical imaging, electrical impedance tomography seeks the conductivity distribution of biological tissue from boundary voltage measurements. In structural engineering, non-destructive evaluation identifies damage or material degradation in load-bearing components by inverting vibration or guided-wave measurements for the local stiffness field.

Despite the diversity of the underlying physics, each of these problems shares the similar mathematical structure. The inverse problem is formulated as a minimization of a *misfit functional*, which quantifies the discrepancy between the measured data and the response predicted from a trial parameter field. The state field must satisfy a governing partial differential equation (PDE), which enters the formulation as a constraint enforced through a Lagrange multiplier. The resulting problem is therefore classified as a *PDE-constrained optimization* problem.

As in the previous example of image deblurring, the inverse medium problem requires the solution of a nonlinear optimization problem that, at best, exploits gradient information. Computing the derivative of the misfit functional with respect to the parameter field is therefore central to any practical inversion algorithm. Four broad strategies are available.

- **Direct sensitivity method.** One differentiates the governing equation with respect to the parameter field to obtain a *sensitivity equation* whose solution yields the required gradient analytically.
- **Finite differencing.** Each component of the gradient is approximated by perturbing the corresponding parameter entry and re-solving the forward

problem. The implementation is straightforward, but the cost again scales with the parameter dimension and the approximation is susceptible to truncation and round-off errors.

- **Adjoint method.** An *adjoint equation*, derived from the optimality conditions of the Lagrangian, is solved once to obtain the gradient with respect to the entire parameter field. The computational cost is therefore independent of the parameter dimension, making the adjoint method the workhorse of large-scale PDE-constrained optimization.
- **Automatic differentiation.** The chain rule is applied systematically to the discrete computational graph of the forward solver. In reverse mode, automatic differentiation produces gradients at a cost comparable to that of the adjoint method, although with larger memory overhead; it may be viewed as an algorithmic implementation of the adjoint principle applied at the discrete level.

In the deblurring problem, the forward operator was a linear convolution and the misfit functional was quadratic in the unknown image; the gradient was therefore available in closed form by direct differentiation. For the PDE-governed problems treated in the remainder of this chapter, the parameter dimension is large and the forward map is defined only implicitly through the solution of a differential equation. We accordingly develop the adjoint method in detail.

4.2 Elliptic problem

4.2.1 Problem definition

We begin the discussion with a simple model elliptic problem. Given measured data u_d and a source term f, find the parameter field p by solving

$$\min_p J[u,p], \quad J[u,p] \coloneqq \underbrace{\frac{1}{2}(u-u_d, u-u_d)}_{=M[u]} + \underbrace{\frac{\alpha}{2}(\operatorname{grad} p, \operatorname{grad} p)}_{=R[p]}, \tag{4.1}$$

subject to the governing equation and the boundary condition

$$\begin{cases} \operatorname{div}(p \operatorname{grad} u) + f = 0 & \text{in } \Omega, \\ u = 0 & \text{on } \Gamma_D, \\ p \operatorname{grad} u \cdot n = 0 & \text{on } \Gamma_N \end{cases} \tag{4.2}$$

where $\Gamma_D \cup \Gamma_N = \partial\Omega$ and the parameter field p belongs to the admissible set

$$W \coloneqq \left\{ p \in H^1(\Omega) \ : \ p \geq p_o > 0 \text{ in } \Omega \right\}. \tag{4.3}$$

Recall

$$(a,b) = (a,b)_\Omega = \int_\Omega ab \, d\Omega,$$

$$(a,b)_\Gamma = \int_\Gamma ab \, d\Gamma,$$

where $\Gamma = \partial\Omega$.

Here, the functional $M[u]$ is called the misfit functional and $R[p]$ is the regularization on the parameter field p. Specifically, we use the *first-order Tikhonov regularization*:

$$R[p] = \frac{\alpha}{2}(\operatorname{grad} p, \operatorname{grad} p) = \frac{\alpha}{2}\int_\Omega |\operatorname{grad} p|^2 \, d\Omega, \tag{4.4}$$

which penalizes oscillations in p while leaving the mean value unpenalized: the null space of grad consists of constant functions, so spatially uniform shifts incur no cost.

4.2.2 Lagrangian and necessary condition

We enforce the governing equation as a constraint using a Lagrange multiplier (also called *adjoint variable*) v, which yields the Lagrangian functional

$$L[u, v, p] = M[u] + R[p] + E[u, v, p]. \tag{4.5}$$

The constraint $E[u, v, p]$ is the weak form of (4.2) tested against v,

$$E[u, v, p] = (p \operatorname{grad} u, \operatorname{grad} v) - (f, v). \tag{4.6}$$

Here, we have a symmetric functional setting for $u \in U$ and $v \in V$ such that

$$U = V = \left\{ u \in H^1(\Omega) \ : \ u = 0 \text{ on } \Gamma_D \right\}. \tag{4.7}$$

The first-order necessary conditions (KKT conditions) require that the variations of the Lagrangian with respect to all three fields vanish:

$$d_u L[u, v, p](\tilde{u}) = 0, \quad \forall\, \tilde{u} \in U, \tag{4.8a}$$
$$d_p L[u, v, p](\tilde{p}) = 0, \quad \forall\, \tilde{p} \in W. \tag{4.8b}$$
$$d_v L[u, v, p](\tilde{v}) = 0, \quad \forall\, \tilde{v} \in V, \tag{4.8c}$$

The third necessary condition (4.8c) leads to the constraint, or the *state problem*:

$$0 = (p \operatorname{grad} u, \operatorname{grad} \tilde{v}) - (f, \tilde{v}). \tag{4.9}$$

The first necessary condition (4.8a) gives the *adjoint problem*:

$$0 = (\tilde{u}, u - u_d) + (p \operatorname{grad} \tilde{u}, \operatorname{grad} v). \tag{4.10}$$

The bilinear form $(p \operatorname{grad} u, \operatorname{grad} v)$ is symmetric in u and v, so the state and adjoint operators coincide. Notice that the adjoint problem is driven by the misfit $u - u_d$: if the state field were to match the data exactly, the right-hand side would vanish and the adjoint variable would be trivially zero. The corresponding strong form, i.e., the Euler equation, reads

$$\begin{cases} \operatorname{div}(p \operatorname{grad} v) + u - u_d = 0 & \text{in } \Omega, \\ v = 0 & \text{on } \Gamma_D, \\ p \operatorname{grad} v \cdot n = 0 & \text{on } \Gamma_N. \end{cases} \tag{4.11}$$

Lastly, the second necessary condition (4.8b) gives the *control problem*:

$$0 = \alpha (\operatorname{grad} \tilde{p}, \operatorname{grad} p) + (\tilde{p} \operatorname{grad} u, \operatorname{grad} v). \tag{4.12}$$

4.2.3 Gradient-based optimization and the adjoint method

The three necessary conditions are coupled and, once discretized, form a system that is too large to store and factorize directly. We therefore adopt an iterative algorithm to find the optimal p starting from an initial guess $p^{(0)}$. At each iteration, the current estimate $p^{(k)}$ is updated by

$$p^{(k+1)} = p^{(k)} + sd^{(k)}, \tag{4.13}$$

where $d^{(k)}$ is the *search direction* and s is the step length. Several choices for the search direction are available: setting $d^{(k)}$ to the negative gradient yields the *steepest descent method*; incorporating information from previous gradient gives the *nonlinear conjugate gradient method*; and retaining a limited history of gradients leads to the *L-BFGS method*. All of these methods require the gradient of the objective functional J with respect to the parameter field p.

The gradient can be approximated numerically by finite differencing, i.e.,

$$d_p J[\ldots, p](\tilde{p}) \approx \frac{J[\ldots, p + \epsilon\tilde{p}] - J[\ldots, p]}{\epsilon}. \tag{4.14}$$

In addition to the choice of a suitably small ϵ, this computation is problematic because it requires $N + 1$ functional evaluations for N degrees of freedom in p. Alternatively, the adjoint method obtains the gradient by a sequential evaluation of the KKT conditions. Given a current estimate $p^{(k)}$, the gradient $g^{(k)}$ is obtained as follows.

1. Solve the state problem for $u^{(k)}$:

$$0 = \left(p^{(k)} \operatorname{grad} u^{(k)}, \operatorname{grad} \tilde{v}\right) - (f, \tilde{v}), \quad \forall \tilde{v} \in V. \tag{4.15}$$

2. Solve the adjoint problem for $v^{(k)}$:

$$0 = \left(p^{(k)} \operatorname{grad} \tilde{u}, \operatorname{grad} v^{(k)}\right) + \left(\tilde{u}, u^{(k)} - u_d\right), \quad \forall \tilde{u} \in U. \tag{4.16}$$

3. Evaluate the gradient equation for $g^{(k)}$:

$$\left(g^{(k)}, \tilde{p}\right) = \alpha \left(\operatorname{grad} \tilde{p}, \operatorname{grad} p^{(k)}\right) + \left(\tilde{p} \operatorname{grad} u^{(k)}, \operatorname{grad} v^{(k)}\right), \quad \forall \tilde{p} \in W. \tag{4.17}$$

Notice that the right-hand side of (4.17) is in general nonzero away from the optimum; it vanishes precisely when the KKT conditions (4.8) are satisfied. Strictly speaking, the right-hand side defines a linear functional on W, i.e., an element of the dual space W^*. The function $g^{(k)} \in W$ is its Riesz representation with respect to the L^2 inner product, i.e.,

$$\begin{cases} g^{(k)} = -\alpha \operatorname{div} \operatorname{grad} p^{(k)} + \operatorname{grad} u^{(k)} \cdot \operatorname{grad} v^{(k)}, & x \in \Omega \\ \tilde{p} = 0 \text{ or } \operatorname{grad} p^{(k)} = 0, & x \in \Gamma. \end{cases} \tag{4.18}$$

Upon finite element discretization, this amounts to solving a mass-matrix system $\mathbf{M}\mathbf{g}^{(k)} = \mathbf{r}^{(k)}$, where $\mathbf{r}^{(k)}$ is the assembled right-hand-side vector.

Overall, the adjoint method computes the gradient at a cost independent of the parameter dimension, and the resulting expressions are exact at the continuous level; approximation enters only through the finite element discretization of the function spaces.

Algorithm 2 Steepest descent method

Choose initial guess $p^{(0)} \in W$ and tolerance $\varepsilon > 0$
Compute gradient direction $g^{(0)}$ via (4.17)
for $k = 0, 1, 2, \ldots$ **do**
    if $\left\| g^{(k)} \right\| \leq \varepsilon$ **then**
        break ▷ convergence
    end if
    $d^{(k)} := -g^{(k)}$ ▷ search direction
    Determine step length $s^{(k)} > 0$
    $p^{(k+1)} := p^{(k)} + s^{(k)}\, d^{(k)}$ ▷ update parameter field
    Compute gradient direction $g^{(k+1)}$ via (4.17)
end for
return $p^{(k)}$

Algorithm 3 Nonlinear conjugate gradient method

Choose initial guess $p^{(0)} \in W$ and tolerance $\varepsilon > 0$
Compute gradient direction $g^{(0)}$ via (4.17)
$d^{(0)} := -g^{(0)}$ ▷ initial search direction
for $k = 0, 1, 2, \ldots$ **do**
    if $\left\| g^{(k)} \right\| \leq \varepsilon$ **then**
        break ▷ convergence
    end if
    Determine step length $s^{(k)} > 0$
    $p^{(k+1)} := p^{(k)} + s^{(k)}\, d^{(k)}$ ▷ update parameter field
    Compute gradient direction $g^{(k+1)}$ via (4.17)
    $\beta^{(k+1)} := \dfrac{\left(g^{(k+1)},\, g^{(k+1)}\right)}{\left(g^{(k)},\, g^{(k)}\right)}$ ▷ Fletcher–Reeves
    $d^{(k+1)} := -g^{(k+1)} + \beta^{(k+1)}\, d^{(k)}$ ▷ update search direction
end for
return $p^{(k)}$

4.2.4 Step length criteria

The gradient determines the search direction, but the step length remains to be chosen. Given a descent direction $d^{(k)}$, a line search seeks a step length $s > 0$ such that the update $p^{(k+1)} = p^{(k)} + sd^{(k)}$ produces a sufficient decrease in the objective. A naive choice such as exact minimization,

$$s = \arg\min_{s>0} J[p^{(k)} + sd^{(k)}], \tag{4.19}$$

is generally impractical for PDE-constrained problems because each evaluation of J requires the solution of the state equation. Instead, one enforces *inexact* line search conditions that guarantee global convergence while requiring only a modest number of functional evaluations per iteration.

Figure 4.1 illustrates the effect of the step length on convergence. A near-optimal step length yields monotone decrease toward the minimizer, whereas an excessively large step length overshoots, causing the iterates to oscillate about the solution.

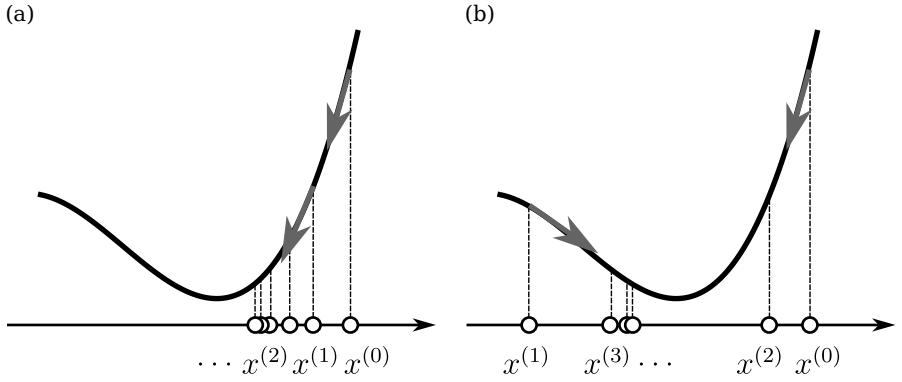

Figure 4.1: Effect of step length on steepest descent. (a) Near-optimal step length: the iterates converge monotonically. (b) Excessively large step length: the iterates overshoot and oscillate.

Armijo condition (sufficient decrease of objective)

The step length s satisfies the *Armijo condition* if

$$J[p^{(k)} + sd^{(k)}] \leq J[p^{(k)}] + c_1 s \left(g^{(k)}, d^{(k)}\right), \tag{4.20}$$

where $c_1 \in (0, 1)$ is a tolerance parameter (typically $c_1 = 10^{-4}$). The right-hand side is the first-order Taylor expansion of J along $d^{(k)}$, scaled by c_1; the condition requires the actual reduction to be at least a fraction c_1 of the predicted linear decrease.

The Armijo condition alone does not prevent s from being arbitrarily small. In practice, one combines it with a *backtracking* strategy [Quarteroni et al., 2006]: starting from an initial trial step $\bar{s}$, the step is repeatedly contracted by a factor $\rho \in (0, 1)$,

$$s \leftarrow \rho s, \tag{4.21}$$

until (4.20) is satisfied (See Algorithm 4). Each backtracking step requires one evaluation of J, and hence one solve of the state equation. The backtracking Armijo method is simple, inexpensive per trial step, and widely used in PDE-constrained optimization.

Algorithm 4 Backtracking line search (Armijo)

Require: Descent direction $d^{(k)}$, current objective value $J^{(k)} := J[p^{(k)}]$, directional derivative $\delta := \left(g^{(k)}, d^{(k)}\right) < 0$

Require: Parameters $\bar{s} > 0$ (initial step), $\rho \in (0,1)$ (contraction factor), $c_1 \in (0,1)$ (sufficient decrease parameter)

```
s := s̄
while J[p^(k) + s d^(k)] > J^(k) + c1 s δ do
    s := ρ s                                   ▷ contract step length
end while
return s
```

Wolfe conditions (sufficient decrease of objective and its slope)

The *Wolfe conditions* augment (4.20) with a *curvature condition* that prevents the accepted step from being too short:

$$J[p^{(k)} + sd^{(k)}] \leq J[p^{(k)}] + c_1 s \left(g^{(k)}, d^{(k)}\right), \tag{4.22}$$

$$\left(g^{(k+1)}, d^{(k)}\right) \geq c_2 \left(g^{(k)}, d^{(k)}\right), \tag{4.23}$$

where $0 < c_1 < c_2 < 1$ (typical choices: $c_1 = 10^{-4}$, $c_2 = 0.9$ for Newton and quasi-Newton methods, $c_2 = 0.1$ for nonlinear conjugate gradient). The curvature condition (4.23) requires the directional derivative at the new iterate to be less negative than at the current iterate, ensuring that the step is not accepted in a region where the objective is still steeply decreasing. A more restrictive variant, the *strong Wolfe condition*, replaces (4.23) with

$$\left|\left(g^{(k+1)}, d^{(k)}\right)\right| \leq c_2 \left|\left(g^{(k)}, d^{(k)}\right)\right|, \tag{4.24}$$

which additionally excludes points where the directional derivative is large and positive.

4.2.5 Newton method

Gradient-based algorithms, such as descent methods, approximate a functional by a linear model at each iteration around the current iterate. The resulting linear model determines a descent direction but not a step length, since a linear functional is unbounded below along any descent direction. Consequently, a separate line-search or step-length selection procedure is required.

In contrast, Newton's method approximates the functional by a quadratic model at each iteration (Figure 4.2). Under appropriate conditions, this quadratic model admits a unique minimizer that is analytically computable, yielding both the search direction and the step length simultaneously.

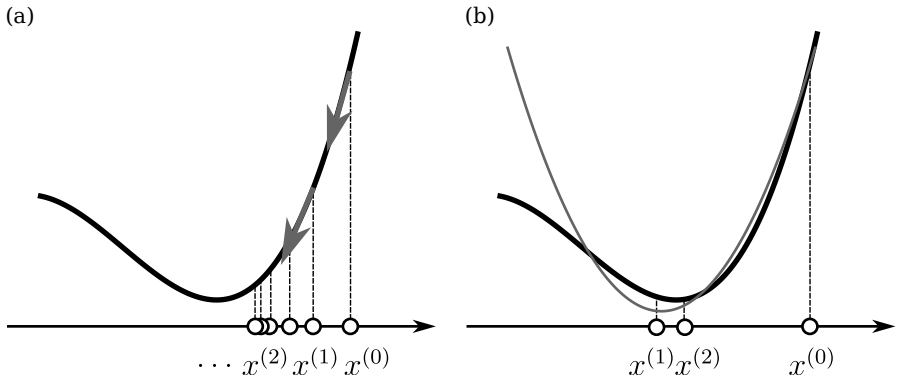

Figure 4.2: Comparison of steepest descent and Newton's method. (a) Steepest descent provides only a search direction; the step length must be determined by a separate line search. (b) Newton's method determines both the search direction and a natural step length from the local quadratic model of the objective.

Constructing such a quadratic model requires both the first and second derivatives of the functional. Suppose the stationary condition for a minimization problem is

$$d_p J[p](\tilde{p}) = 0, \quad \forall\, \tilde{p}. \tag{4.25}$$

Given a current iterate $p^{(k)}$, we seek an update $\hat{p}$ such that $p^{(k+1)} = p^{(k)} + \hat{p}$. Linearizing the stationary condition about $p^{(k)}$ via a first-order Taylor expansion yields

$$\begin{aligned} 0 &= d_p J[p^{(k)} + \hat{p}](\tilde{p}) \\ &\approx d_p J[p^{(k)}](\tilde{p}) + d_p^2 J[p^{(k)}](\hat{p}, \tilde{p}), \quad \forall \tilde{p}. \end{aligned} \tag{4.26}$$

Thus, the update $\hat{p}$, which encodes both the search direction and the step length, is obtained by solving the above linear problem.

We now apply the approach in the KKT condition (4.8). Letting $[\ldots]$ denote $[u, v, p]$, we have

$$0 = d_u L\,[\ldots]\,(\tilde{u}) + d_u d_u L\,[\ldots]\,(\tilde{u}, \hat{u}) + d_p d_u L\,[\ldots]\,(\tilde{u}, \hat{p}) + d_v d_u L\,[\ldots]\,(\tilde{u}, \hat{v})\,, \tag{4.27a}$$

$$0 = d_p L\,[\ldots]\,(\tilde{p}) + d_u d_p L\,[\ldots]\,(\tilde{p}, \hat{u}) + d_p d_p L\,[\ldots]\,(\tilde{p}, \hat{p}) + d_v d_p L\,[\ldots]\,(\tilde{p}, \hat{v})\,, \tag{4.27b}$$

$$0 = d_v L\,[\ldots]\,(\tilde{v}) + d_u d_v L\,[\ldots]\,(\tilde{v}, \hat{u}) + d_p d_v L\,[\ldots]\,(\tilde{v}, \hat{p}) + d_v d_v L\,[\ldots]\,(\tilde{v}, \hat{v})\,. \tag{4.27c}$$

For the model problem, the above equations read ($\nabla = \text{grad}\,$)

$$\begin{array}{lll} \left(\tilde{u}, \hat{u}^{(k)}\right) + \left(\hat{p}^{(k)} \nabla \tilde{u}, \nabla v^{(k)}\right) + \left(p^{(k)} \nabla \tilde{u}, \nabla \hat{v}^{(k)}\right) &= -\left(p^{(k)} \nabla \tilde{u}, \nabla v^{(k)}\right) - \left(\tilde{u}, u^{(k)} - u_d\right), \\ \left(\tilde{p}\, \nabla \hat{u}^{(k)}, \nabla v^{(k)}\right) + \alpha \left(\nabla \tilde{p}, \nabla \hat{p}^{(k)}\right) + \left(\tilde{p}\, \nabla u^{(k)}, \nabla \hat{v}^{(k)}\right) &= -\alpha \left(\nabla \tilde{p}, \nabla p^{(k)}\right) - \left(\tilde{p}\, \nabla u^{(k)}, \nabla v^{(k)}\right), \\ \left(p^{(k)} \nabla \hat{u}^{(k)}, \nabla \tilde{v}\right) + \left(\hat{p}^{(k)} \nabla u^{(k)}, \nabla \tilde{v}\right) &= -\left(p^{(k)} \nabla u^{(k)}, \nabla \tilde{v}\right) + (f, \tilde{v})\,. \end{array} \tag{4.28}$$

Removing the dependence on the test functions $\tilde{u}$, $\tilde{p}$, $\tilde{v}$, the above system can be written as

$$\begin{bmatrix} F & A^\dagger & B^\dagger \\ A & D & C^\dagger \\ B & C & 0 \end{bmatrix} \begin{pmatrix} \hat{u}^{(k)} \\ \hat{p}^{(k)} \\ \hat{v}^{(k)} \end{pmatrix} = - \begin{pmatrix} g_u \\ g_p \\ g_v \end{pmatrix}. \tag{4.29}$$

We would never attempt to store and factorize the matrix in (4.29). Instead, we solve the state and adjoint problems sequentially so that, for a given $p^{(k)}$, we find $u^{(k)}$ and $v^{(k)}$ satisfying $g_u = g_v = 0$. The first and last block rows can then be condensed out, since:

$$B\hat{u}^{(k)} = -C\hat{p}^{(k)}, \tag{4.30}$$

$$B^\dagger \hat{v}^{(k)} = -F\hat{u}^{(k)} - A^\dagger \hat{p}^{(k)}. \tag{4.31}$$

Substituting into the second block row yields the reduced Newton system

$$H\hat{p}^{(k)} = -g_p, \tag{4.32}$$

where

$$H = D + C^\dagger B^{-\dagger}\left(FB^{-1}C - A^\dagger\right) - AB^{-1}C. \tag{4.33}$$

The action of the reduced Hessian on a trial direction $\hat{p}^{(k)}$ is therefore computed sequentially as follows.

1. Solve the incremental state problem for $\hat{u}^{(k)}$:

$$B\hat{u}^{(k)} = -C\hat{p}^{(k)}. \tag{4.34}$$

2. Solve the incremental adjoint problem for $\hat{v}^{(k)}$:

$$B^\dagger \hat{v}^{(k)} = -F\hat{u}^{(k)} - A^\dagger \hat{p}^{(k)}. \tag{4.35}$$

3. Evaluate the reduced Hessian action:

$$H\hat{p}^{(k)} = A\hat{u}^{(k)} + D\hat{p}^{(k)} + C^\dagger \hat{v}^{(k)}. \tag{4.36}$$

The Newton method requires the reduced Hessian H to be positive definite, which is not guaranteed away from the solution. Two strategies are commonly used to address this difficulty.

The first is the *Gauss-Newton method*, in which the second-order adjoint terms involving A and $A^\dagger$ are dropped from H. The resulting approximation

$$H^{\mathrm{GN}} = D + C^\dagger B^{-\dagger} F\, B^{-1}C \tag{4.37}$$

is symmetric positive definite by construction, so the conjugate gradient (CG) method can always be applied. The trade-off is that quadratic convergence is lost; Gauss–Newton converges superlinearly or linearly, depending on the size of the residual at the solution.

The second is the *truncated Newton method*, in which one solves the full Newton system with CG but terminates early if a direction of non-positive curvature is encountered, i.e., if $d^\top H d \leq 0$ for some CG iterate d. The truncated step is then used within a trust-region framework to ensure global convergence. This approach retains the full Hessian information when it is beneficial and effortlessly degrades to a gradient or Gauss-Newton-like step when H is indefinite.

Algorithm 5 summarizes the truncated Newton-CG procedure for solving the discretized reduced Hessian system $\mathbf{H}\hat{\mathbf{p}} = -\mathbf{g}_p$.

In practice, both the Gauss–Newton and truncated Newton methods are combined with a backtracking line search using an initial step length of 1, since neither method solves the Newton system exactly.

Algorithm 5 Truncated Newton-CG method for $\mathbf{H}\hat{\mathbf{p}} = -\mathbf{g}_p$

Set initial value $\hat{\mathbf{p}}^{(0)}$
$\mathbf{r}^{(0)} \coloneqq -\mathbf{g}_p - \mathbf{H}\hat{\mathbf{p}}^{(0)}$ ▷ initial residual
$\mathbf{d}^{(0)} \coloneqq \mathbf{r}^{(0)}$ ▷ initial search direction
for $k = 0, 1, 2, \ldots$ **do**
  $\kappa^{(k)} \coloneqq \left(\mathbf{d}^{(k)}\right)^\dagger \mathbf{H}\,\mathbf{d}^{(k)}$ ▷ curvature along search direction
  if $\kappa^{(k)} \leq 0$ **then**
    return $\hat{\mathbf{p}}^{(k)}$ ▷ negative curvature detected; truncate
  end if
  $\alpha^{(k)} \coloneqq \dfrac{\|\mathbf{r}^{(k)}\|^2}{\kappa^{(k)}}$ ▷ step length
  $\hat{\mathbf{p}}^{(k+1)} \coloneqq \hat{\mathbf{p}}^{(k)} + \alpha^{(k)}\mathbf{d}^{(k)}$ ▷ update iterate
  $\mathbf{r}^{(k+1)} \coloneqq \mathbf{r}^{(k)} - \alpha^{(k)}\mathbf{H}\,\mathbf{d}^{(k)}$ ▷ update residual
  if $\|\mathbf{r}^{(k+1)}\| \leq \varepsilon\,\|\mathbf{r}^{(0)}\|$ **then**
    return $\hat{\mathbf{p}}^{(k+1)}$ ▷ convergence criterion
  end if
  $\beta^{(k)} \coloneqq -\dfrac{\|\mathbf{r}^{(k+1)}\|^2}{\|\mathbf{r}^{(k)}\|^2}$ ▷ direction update coefficient
  $\mathbf{d}^{(k+1)} \coloneqq \mathbf{r}^{(k+1)} - \beta^{(k)}\mathbf{d}^{(k)}$ ▷ update search direction
end for
return $\hat{\mathbf{p}}^{(k+1)}$

Exercise 4.2.1 *Let B be invertible, $F = F^\dagger \succeq 0$, $D = D^\dagger \succ 0$, and let A, C be arbitrary matrices of compatible dimensions. Define*

$$H \coloneqq D + C^\dagger B^{-\dagger}(FB^{-1}C - A^\dagger) - AB^{-1}C, \tag{4.38}$$

$$H^{\mathrm{GN}} \coloneqq D + C^\dagger B^{-\dagger} F\, B^{-1}C. \tag{4.39}$$

Show that H is symmetric and H^{GN} is symmetric positive definite.