July 20, 2013

Chapter 9: Nonlinear Systems and Optimization

Uri M. Ascher and Chen Greif
Department of Computer Science
The University of British Columbia
{ascher,greif}@cs.ubc.ca

Slides for the book
A First Course in Numerical Methods (published by SIAM, 2011)
`http://www.ec-securehost.com/SIAM/CS07.html`

Goals of this chapter

- To devise and assess algorithms for (continuous) unconstrained optimization, a problem setting which arises frequently in applications;
- to solve systems of nonlinear equations, thus extending Chapter 3;
- to consider more advanced conditions and techniques for constrained optimization.

Outline

- Systems of nonlinear equations
- Unconstrained optimization
- *Constrained optimization

*advanced

Systems of equations

- Equipped with knowledge on how to solve scalar nonlinear equations as well as linear systems, it is time to combine them and consider systems of nonlinear equations

$$\begin{aligned} f_1(x_1, x_2, \ldots, x_n) &= 0, \\ f_2(x_1, x_2, \ldots, x_n) &= 0, \\ \vdots \quad &= \quad \vdots \\ f_n(x_1, x_2, \ldots, x_n) &= 0. \end{aligned}$$

- In vector notation, write this as $\mathbf{f}(\mathbf{x}) = \mathbf{0}$.
- Chapter 3 promises that Newton's method extends directly to the present problem. But Newton's method requires the derivative of f, so here we have to extend the concept of derivative first.

Systems of equations

- Equipped with knowledge on how to solve scalar nonlinear equations as well as linear systems, it is time to combine them and consider systems of nonlinear equations

$$\begin{aligned} f_1(x_1, x_2, \ldots, x_n) &= 0, \\ f_2(x_1, x_2, \ldots, x_n) &= 0, \\ \vdots \quad &= \quad \vdots \\ f_n(x_1, x_2, \ldots, x_n) &= 0. \end{aligned}$$

- In vector notation, write this as $\mathbf{f}(\mathbf{x}) = \mathbf{0}$.
- Chapter 3 promises that Newton's method extends directly to the present problem. But Newton's method requires the derivative of f, so here we have to extend the concept of derivative first.

Multivariate Taylor expansion

- Let $\mathbf{x} = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix}$, function $\mathbf{f} = \begin{pmatrix} f_1 \\ f_2 \\ \vdots \\ f_m \end{pmatrix}$, direction $\mathbf{p} = \begin{pmatrix} p_1 \\ p_2 \\ \vdots \\ p_n \end{pmatrix}$.
  Assume that $\mathbf{f}$ is sufficiently smooth (at least two bounded derivatives).
- Then Taylor expansion gives

$$\mathbf{f}(\mathbf{x}+\mathbf{p}) = \mathbf{f}(\mathbf{x}) + J(\mathbf{x})\mathbf{p} + \mathcal{O}(\|\mathbf{p}\|^2),$$

where $J(\mathbf{x})$ is the Jacobian matrix of first derivatives of $\mathbf{f}$ at $\mathbf{x}$,

$$J(\mathbf{x}) = \begin{pmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & \cdots & \frac{\partial f_2}{\partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial f_m}{\partial x_1} & \frac{\partial f_m}{\partial x_2} & \cdots & \frac{\partial f_m}{\partial x_n} \end{pmatrix}.$$

Newton's method

- Derivation: By Taylor series,
$$\mathbf{f}(\mathbf{x}) = \mathbf{f}(\mathbf{x}_k) + J(\mathbf{x}_k)(\mathbf{x} - \mathbf{x}_k) + \mathcal{O}(\|\mathbf{x} - \mathbf{x}_k\|^2).$$
- For $\mathbf{x} = \mathbf{x}^*$, also $\mathbf{f}(\mathbf{x}) = \mathbf{0}$.
- Neglect nonlinear term and define method by
$$\mathbf{0} = \mathbf{f}(\mathbf{x}_k) + J(\mathbf{x}_k)(\mathbf{x}_{k+1} - \mathbf{x}_k).$$
- This is conceptually identical to the procedure (Chapter 3) for one function in one variable.
- Algorithm: Given an initial guess $\mathbf{x}_0$;

```
for $k = 0, 1, \ldots,$ until convergence
        solve $J(\mathbf{x}_k)\mathbf{p} = -\mathbf{f}(\mathbf{x}_k)$,
        set   $\mathbf{x}_{k+1} = \mathbf{x}_k + \mathbf{p}$.
end
```

Newton's method

- Derivation: By Taylor series,

$$\mathbf{f}(\mathbf{x}) = \mathbf{f}(\mathbf{x}_k) + J(\mathbf{x}_k)(\mathbf{x} - \mathbf{x}_k) + \mathcal{O}(\|\mathbf{x} - \mathbf{x}_k\|^2).$$

- For $\mathbf{x} = \mathbf{x}^*$, also $\mathbf{f}(\mathbf{x}) = \mathbf{0}$.
- Neglect nonlinear term and define method by

$$\mathbf{0} = \mathbf{f}(\mathbf{x}_k) + J(\mathbf{x}_k)(\mathbf{x}_{k+1} - \mathbf{x}_k).$$

- This is conceptually identical to the procedure (Chapter 3) for one function in one variable.
- Algorithm: Given an initial guess $\mathbf{x}_0$;

```
for $k = 0, 1, \ldots,$ until convergence
        solve $J(\mathbf{x}_k)\mathbf{p} = -\mathbf{f}(\mathbf{x}_k)$,
        set   $\mathbf{x}_{k+1} = \mathbf{x}_k + \mathbf{p}$.
end
```

Newton's method

- Derivation: By Taylor series,

$$\mathbf{f}(\mathbf{x}) = \mathbf{f}(\mathbf{x}_k) + J(\mathbf{x}_k)(\mathbf{x} - \mathbf{x}_k) + \mathcal{O}(\|\mathbf{x} - \mathbf{x}_k\|^2).$$

- For $\mathbf{x} = \mathbf{x}^*$, also $\mathbf{f}(\mathbf{x}) = \mathbf{0}$.
- Neglect nonlinear term and define method by

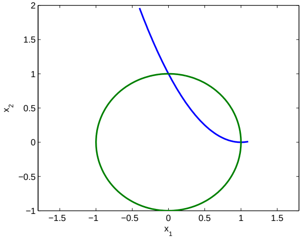
$$\mathbf{0} = \mathbf{f}(\mathbf{x}_k) + J(\mathbf{x}_k)(\mathbf{x}_{k+1} - \mathbf{x}_k).$$

- This is conceptually identical to the procedure (Chapter 3) for one function in one variable.
- Algorithm: Given an initial guess $\mathbf{x}_0$;

```
for $k = 0, 1, \ldots,$ until convergence
        solve $J(\mathbf{x}_k)\mathbf{p} = -\mathbf{f}(\mathbf{x}_k)$,
        set   $\mathbf{x}_{k+1} = \mathbf{x}_k + \mathbf{p}$.
end
```

Example: a parabola meets a circle

$$\mathbf{x} = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}, \quad \mathbf{f}(\mathbf{x}) = \begin{pmatrix} x_1^2 - 2x_1 - x_2 + 1 \\ x_1^2 + x_2^2 - 1 \end{pmatrix}.$$

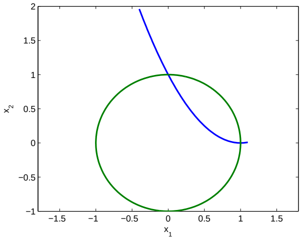


Two solutions: $(0,1)^T$ and $(1,0)^T$.

$$J(\mathbf{x}) = \begin{pmatrix} 2x_1 - 2 & -1 \\ 2x_1 & 2x_2 \end{pmatrix}$$

Example: a parabola meets a circle

$$\mathbf{x} = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}, \quad \mathbf{f}(\mathbf{x}) = \begin{pmatrix} x_1^2 - 2x_1 - x_2 + 1 \\ x_1^2 + x_2^2 - 1 \end{pmatrix}.$$



Two solutions: $(0,1)^T$ and $(1,0)^T$.

$$J(\mathbf{x}) = \begin{pmatrix} 2x_1 - 2 & -1 \\ 2x_1 & 2x_2 \end{pmatrix}$$

Example (cont.)

Using Newton's method for finding the two roots, varying the starting point $\mathbf{x}_0$.

Stopping tolerance `tol` $=$ 1.e-7.

1. Starting at $\mathbf{x}_0 = (0,0)^T$ is bad because $J(\mathbf{x}_0)$ is singular!
2. Starting at $\mathbf{x}_0 = (1,1)^T$ obtain root $(0,1)^T$ in 5 iterations. Observe quadratic convergence.
3. Starting at $\mathbf{x}_0 = (-1,1)^T$ obtain root $(1,0)^T$ in 5 iterations. Observe quadratic convergence.

Example: two-point boundary value ordinary differential equation

- Consider the differential problem

$$u''(t) + e^{u(t)} = 0, \quad 0 < t < 1,$$
$$u(0) = u(1) = 0.$$

- Discretize on a uniform mesh (grid) $t_i = ih, \ i = 0, 1, \ldots, n+1$, where $(n+1)h = 1$:

$$\frac{v_{i+1} - 2v_i + v_{i-1}}{h^2} + e^{v_i} = 0, \qquad i = 1, 2, \ldots, n.$$
$$v_0 = v_{n+1} = 0.$$

- This is a system of nonlinear equations, with $\mathbf{x} \leftarrow \mathbf{v}$ and $f_i(\mathbf{v}) = \frac{v_{i+1} - 2v_i + v_{i-1}}{h^2} + e^{v_i}$.

Example: two-point boundary value ordinary differential equation

- Consider the differential problem
$$u''(t) + e^{u(t)} = 0, \quad 0 < t < 1,$$
$$u(0) = u(1) = 0.$$

- Discretize on a uniform mesh (grid) $t_i = ih,\ i = 0, 1, \ldots, n+1$, where $(n+1)h = 1$:
$$\frac{v_{i+1} - 2v_i + v_{i-1}}{h^2} + e^{v_i} = 0, \qquad i = 1, 2, \ldots, n.$$
$$v_0 = v_{n+1} = 0.$$

- This is a system of nonlinear equations, with $\mathbf{x} \leftarrow \mathbf{v}$ and $f_i(\mathbf{v}) = \frac{v_{i+1} - 2v_i + v_{i-1}}{h^2} + e^{v_i}$.

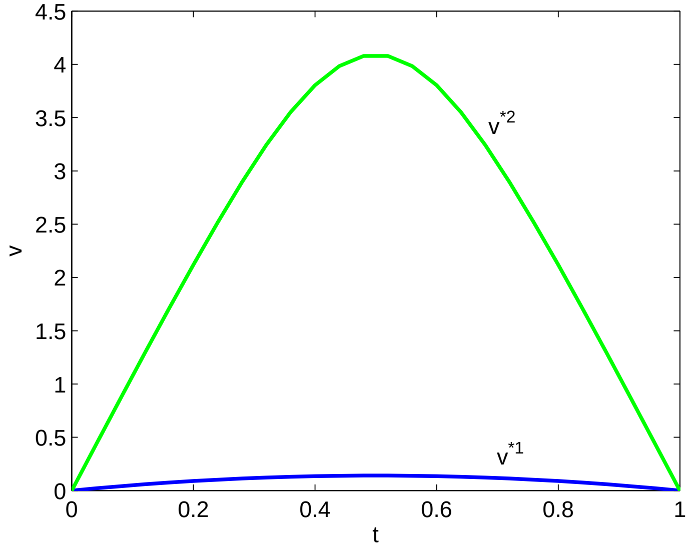
Newton for TPBVP

- Jacobian matrix is possibly large but tridiagonal

$$J = \frac{1}{h^2}\begin{pmatrix} -2+h^2e^{v_1} & 1 & & & \\ 1 & -2+h^2e^{v_2} & 1 & & \\ & \ddots & \ddots & \ddots & \\ & & 1 & -2+h^2e^{v_{n-1}} & 1 \\ & & & 1 & -2+h^2e^{v_n} \end{pmatrix}.$$

- Initial guess $\mathbf{v}_0 = \alpha\left(t_1(1-t_1),\ldots,t_n(1-t_n)\right)^T$.
- Take various values of α and see what happens, setting $\texttt{tol} = 1.\text{e-}8$, $n = 24$:
  1. $\alpha = 0 \;\Rightarrow$ converges in 4 iterations
  2. $\alpha = 10 \;\Rightarrow$ converges in 6 iterations
  3. $\alpha = 20 \;\Rightarrow$ converges in 6 iterations to another solution
  4. $\alpha = 50 \;\Rightarrow$ diverges

Two solutions



[illegible]

Outline

- Systems of nonlinear equations
- Unconstrained optimization
- *Constrained optimization

Multivariate Taylor expansion

- Consider the problem

$$\min_{\mathbf{x}\in\mathbb{R}^n} \phi(\mathbf{x})$$

  Assume that $\phi(\mathbf{x})$ is smooth enough.

- Define gradient vector $\boldsymbol{\nabla}\phi(\mathbf{x})$ and Hessian matrix $\nabla^2\phi(\mathbf{x})$ by

$$\boldsymbol{\nabla}\phi(\mathbf{x}) = \begin{pmatrix} \frac{\partial\phi}{\partial x_1} \\ \frac{\partial\phi}{\partial x_2} \\ \vdots \\ \frac{\partial\phi}{\partial x_n} \end{pmatrix}, \quad \nabla^2\phi(\mathbf{x}) = \begin{pmatrix} \frac{\partial^2\phi}{\partial x_1^2} & \frac{\partial^2\phi}{\partial x_1\partial x_2} & \cdots & \frac{\partial^2\phi}{\partial x_1\partial x_n} \\ \frac{\partial^2\phi}{\partial x_2\partial x_1} & \frac{\partial^2\phi}{\partial x_2^2} & \cdots & \frac{\partial^2\phi}{\partial x_2\partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2\phi}{\partial x_n\partial x_1} & \frac{\partial^2\phi}{\partial x_n\partial x_2} & \cdots & \frac{\partial^2\phi}{\partial x_n^2} \end{pmatrix}.$$

- Taylor expansion near a point $\mathbf{x}\in\mathbb{R}^n$:

$$\phi(\mathbf{x}+\mathbf{p}) = \phi(\mathbf{x}) + \boldsymbol{\nabla}\phi(\mathbf{x})^T\mathbf{p} + \frac{1}{2}\mathbf{p}^T\nabla^2\phi(\mathbf{x})\mathbf{p} + \mathcal{O}(\|\mathbf{p}\|^3).$$

Example

- Given the function

$$\phi(x_1, x_2) = x_1^4 - 2x_1^3 x_2^2 + 4x_1 x_2^3,$$

- the gradient at a point $\mathbf{x}$ is

$$\boldsymbol{\nabla}\phi(\mathbf{x}) = \begin{pmatrix} 4x_1^3 - 6x_1^2 x_2^2 + 4x_2^3 \\ -4x_1^3 x_2 + 12x_1 x_2^2 \end{pmatrix};$$

- the Hessian matrix is

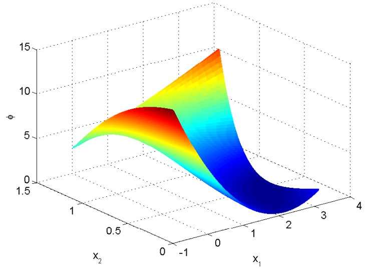
$$H(\mathbf{x}) = \nabla^2\phi(\mathbf{x}) = \begin{pmatrix} 12x_1^2 - 12x_1 x_2^2 & -12x_1^2 x_2 + 12x_2^2 \\ -12x_1^2 x_2 + 12x_2^2 & -4x_1^3 + 24x_1 x_2 \end{pmatrix}.$$

Critical points

- Taylor expansion near a suspected minimum point $\mathbf{x}^*$: in any direction $\mathbf{p}$ with magnitude $\|\mathbf{p}\|$ small enough, must have $\phi(\mathbf{x}^* + \mathbf{p}) \geq \phi(\mathbf{x}^*)$.
- Hence

$$\nabla\phi(\mathbf{x}^*) = \mathbf{0}.$$

  This defines a critical point.
- Similar condition also for a maximum or a saddle point.



Conditions for unconstrained minimum

$$\min_{\mathbf{x}\in\mathbb{R}^n} \phi(\mathbf{x}).$$

Assume that $\phi(\mathbf{x})$ is smooth enough.

- A necessary condition for having a local minimum at $\mathbf{x}^*$ is that $\mathbf{x}^*$ be a critical point

  $$\boldsymbol{\nabla}\phi(\mathbf{x}^*) = \mathbf{0},$$

  and that the symmetric Hessian matrix $\nabla^2\phi(\mathbf{x}^*)$ be positive semi-definite.
- A sufficient condition is that also $\nabla^2\phi(\mathbf{x}^*)$ be positive definite.

Descent direction

- At point $\mathbf{x}$ the vector $\mathbf{p}$ is a descent direction if

$$\nabla \phi(\mathbf{x})^T \mathbf{p} = \sum_{i=1}^{n} p_i \frac{\partial \phi}{\partial x_i} < 0.$$

- A small step in a descent direction gives reduction in the objective function:

$$\phi(\mathbf{x} + \alpha \mathbf{p}) < \phi(\mathbf{x})$$

  for scalar $0 < \alpha \ll 1$.

- Therefore, we can construct an iterative method that keeps reducing ϕ until convergence by using descent directions and controlled step sizes.

Descent direction

- At point $\mathbf{x}$ the vector $\mathbf{p}$ is a descent direction if

$$\boldsymbol{\nabla}\phi(\mathbf{x})^T\mathbf{p} = \sum_{i=1}^{n} p_i \frac{\partial \phi}{\partial x_i} < 0.$$

- A small step in a descent direction gives reduction in the objective function:

$$\phi(\mathbf{x} + \alpha\mathbf{p}) < \phi(\mathbf{x})$$

  for scalar $0 < \alpha \ll 1$.

- Therefore, we can construct an iterative method that keeps reducing ϕ until convergence by using descent directions and controlled step sizes.

Gradient-based methods

Consider

$$\begin{aligned} \mathbf{x}_{k+1} &= \mathbf{x}_k + \alpha_k \mathbf{p}_k, \quad \text{where} \\ \mathbf{p}_k &= -B_k^{-1} \boldsymbol{\nabla} \phi(\mathbf{x}_k). \end{aligned}$$

If B_k is symmetric positive definite then $\mathbf{p}_k$ is a descent direction. Note $\alpha_k > 0$.

1. Gradient descent: $B_k = I$. How to choose α_k?
2. Nonlinear conjugate gradients.
3. Newton: $B_k = \nabla^2 \phi(\mathbf{x}_k)$. Set $\alpha_k = 1$ or damped Newton: search for $\alpha_k \le 1$ guaranteeing descent.
4. Secant, or quasi-Newton.
5. Inexact Newton, Newton-CG.
6. Gauss-Newton for nonlinear least squares data fitting.

Gradient-based methods

Consider

$$\begin{aligned}\mathbf{x}_{k+1} &= \mathbf{x}_k + \alpha_k \mathbf{p}_k, \quad \text{where} \\ \mathbf{p}_k &= -B_k^{-1} \nabla \phi(\mathbf{x}_k).\end{aligned}$$

If B_k is symmetric positive definite then $\mathbf{p}_k$ is a descent direction. Note $\alpha_k > 0$.

1. Gradient descent: $B_k = I$. How to choose α_k?
2. Nonlinear conjugate gradients.
3. Newton: $B_k = \nabla^2 \phi(\mathbf{x}_k)$. Set $\alpha_k = 1$ or damped Newton: search for $\alpha_k \le 1$ guaranteeing descent.
4. Secant, or quasi-Newton.
5. Inexact Newton, Newton-CG.
6. Gauss-Newton for nonlinear least squares data fitting.

Gradient-based methods

Consider

$$\begin{aligned} \mathbf{x}_{k+1} &= \mathbf{x}_k + \alpha_k \mathbf{p}_k, \quad \text{where} \\ \mathbf{p}_k &= -B_k^{-1} \boldsymbol{\nabla}\phi(\mathbf{x}_k). \end{aligned}$$

If B_k is symmetric positive definite then $\mathbf{p}_k$ is a descent direction. Note $\alpha_k > 0$.

1. Gradient descent: $B_k = I$. How to choose α_k?
2. Nonlinear conjugate gradients.
3. Newton: $B_k = \nabla^2\phi(\mathbf{x}_k)$. Set $\alpha_k = 1$ or damped Newton: search for $\alpha_k \leq 1$ guaranteeing descent.
4. Secant, or quasi-Newton.
5. Inexact Newton, Newton-CG.
6. Gauss-Newton for nonlinear least squares data fitting.

Gradient-based methods

Consider

$$\begin{aligned} \mathbf{x}_{k+1} &= \mathbf{x}_k + \alpha_k \mathbf{p}_k, \quad \text{where} \\ \mathbf{p}_k &= -B_k^{-1} \nabla \phi(\mathbf{x}_k). \end{aligned}$$

If B_k is symmetric positive definite then $\mathbf{p}_k$ is a descent direction. Note $\alpha_k > 0$.

1. Gradient descent: $B_k = I$. How to choose α_k?
2. Nonlinear conjugate gradients.
3. Newton: $B_k = \nabla^2 \phi(\mathbf{x}_k)$. Set $\alpha_k = 1$ or damped Newton: search for $\alpha_k \le 1$ guaranteeing descent.
4. Secant, or quasi-Newton.
5. Inexact Newton, Newton-CG.
6. Gauss-Newton for nonlinear least squares data fitting.

Gradient-based methods

Consider

$$\begin{aligned}\mathbf{x}_{k+1} &= \mathbf{x}_k + \alpha_k \mathbf{p}_k, \quad \text{where} \\ \mathbf{p}_k &= -B_k^{-1} \nabla \phi(\mathbf{x}_k).\end{aligned}$$

If B_k is symmetric positive definite then $\mathbf{p}_k$ is a descent direction. Note $\alpha_k > 0$.

1. Gradient descent: $B_k = I$. How to choose α_k?
2. Nonlinear conjugate gradients.
3. Newton: $B_k = \nabla^2 \phi(\mathbf{x}_k)$. Set $\alpha_k = 1$ or damped Newton: search for $\alpha_k \le 1$ guaranteeing descent.
4. Secant, or quasi-Newton.
5. Inexact Newton, Newton-CG.
6. Gauss-Newton for nonlinear least squares data fitting.

Newton's method

- Simply solve the nonlinear equations

$$\mathbf{f}(\mathbf{x}) \equiv \boldsymbol{\nabla}\phi(\mathbf{x}) = \mathbf{0}$$

  using Newton's method.
- At iteration k do

$$\text{solve } \nabla^2\phi(\mathbf{x}_k)\ \mathbf{p}_k = -\boldsymbol{\nabla}\phi(\mathbf{x}_k) \text{ for } \mathbf{p}_k$$
$$\text{set } \mathbf{x}_{k+1} = \mathbf{x}_k + \mathbf{p}_k.$$

- Note that if the Hessian matrix $\nabla^2\phi(\mathbf{x}_k)$ is positive definite then $\mathbf{p}_k$ minimizes a quadratic approximation to ϕ at $\mathbf{x}_k$:

$$\phi(\mathbf{x}_k + \mathbf{p}) \approx \phi(\mathbf{x}_k) + \boldsymbol{\nabla}\phi(\mathbf{x}_k)^T\mathbf{p} + \tfrac{1}{2}\mathbf{p}^T\nabla^2\phi(\mathbf{x}_k)\mathbf{p}.$$

- Furthermore, Newton's method has the following advantages:
  1. locally, it converges quadratically;
  2. (tautologically, it) retains Hessian sparsity.

Newton's method

- Simply solve the nonlinear equations

$$\mathbf{f}(\mathbf{x}) \equiv \boldsymbol{\nabla}\phi(\mathbf{x}) = \mathbf{0}$$

using Newton's method.

- At iteration k do

$$\text{solve } \nabla^2\phi(\mathbf{x}_k)\ \mathbf{p}_k = -\boldsymbol{\nabla}\phi(\mathbf{x}_k) \text{ for } \mathbf{p}_k$$
$$\text{set } \mathbf{x}_{k+1} = \mathbf{x}_k + \mathbf{p}_k.$$

- Note that if the Hessian matrix $\nabla^2\phi(\mathbf{x}_k)$ is positive definite then $\mathbf{p}_k$ minimizes a quadratic approximation to ϕ at $\mathbf{x}_k$:

$$\phi(\mathbf{x}_k + \mathbf{p}) \approx \phi(\mathbf{x}_k) + \boldsymbol{\nabla}\phi(\mathbf{x}_k)^T\mathbf{p} + \tfrac{1}{2}\mathbf{p}^T\nabla^2\phi(\mathbf{x}_k)\mathbf{p}.$$

- Furthermore, Newton's method has the following advantages:
  1. locally, it converges quadratically;
  2. (tautologically, it) retains Hessian sparsity.

Newton's method deficiencies

1. Requires existence of the Hessian.
2. Requires evaluation of the Hessian.
3. Requires solving a linear system at each iteration.
4. $B_k = \nabla^2\phi(\mathbf{x}_k)$ is symmetric but may not be positive definite.
5. No control over convergence (does it converge? to a minimum point?)

Advanced methods are based on capitalizing on the strengths of Newton's method, while trying to weaken or eliminate its deficiencies.

(Weak) line search

- Suppose that $\mathbf{p}_k$ is a descent direction at $\mathbf{x}_k$. Then for $\alpha_k > 0$ small enough, $\phi(\mathbf{x}_k + \alpha_k \mathbf{p}_k) < \phi(\mathbf{x}_k)$.
- Here is a simple algorithm (Armijo) for determining step size α_k, given descent direction $\mathbf{p}_k$:
  Starting from $\alpha = \alpha_{max}$, repeat until sufficient decrease in ϕ is obtained,

$$\alpha \leftarrow \alpha/2.$$

- The result is $\alpha_k = \alpha$, and set $\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{p}_k$.
- For Newton's method, set $\alpha_{max} = 1$ (and ensure that $\mathbf{p}_k$ is a descent direction).
- For gradient descent, $B_k = I$ so $\mathbf{p}_k$ is always a descent direction, but there is no obvious default value α_{max}.

Combining Newton and gradient descent

- Want the Newton efficiency. But what if $\nabla^2\phi(\mathbf{x}_k)$ is not positive definite?! – Need to ensure a descent search direction.
- So, consider mixing Newton and gradient descent:

$$B_k = \nabla^2\phi(\mathbf{x}_k) + \mu_k I.$$

For $\mu_k > 0$ large enough, this shifts the eigenvalues of B_k into positivity.

- Big question: choose scalar $\mu_k \geq 0$ adaptively, not too small and not too large; this leads to trust region methods.

Secant (quasi-Newton)

- Do not want to form or evaluate Hessian matrix explicitly; and *also*,
- want a positive definite B_k that is *easy to invert*.
- Note by Taylor's expansion

$$\nabla\phi(\mathbf{x}_k) \approx \nabla\phi(\mathbf{x}_{k+1}) - \nabla^2\phi(\mathbf{x}_{k+1})(\mathbf{x}_{k+1} - \mathbf{x}_k).$$

The essential action of the Hessian is therefore in the direction of $\mathbf{w}_k = \mathbf{x}_{k+1} - \mathbf{x}_k$.

- Thus, equire

$$B_{k+1}\mathbf{w}_k = \mathbf{y}_k, \quad \mathbf{y}_k = \nabla\phi(\mathbf{x}_{k+1}) - \nabla\phi(\mathbf{x}_k).$$

- Obtain B_{k+1} as a positive definite rank-2 update of B_k, thus satisfying the above requirements.

Secant (quasi-Newton)

- Do not want to form or evaluate Hessian matrix explicitly; and *also*,
- want a positive definite B_k that is *easy to invert*.
- Note by Taylor's expansion

$$\nabla \phi(\mathbf{x}_k) \approx \nabla \phi(\mathbf{x}_{k+1}) - \nabla^2 \phi(\mathbf{x}_{k+1})(\mathbf{x}_{k+1} - \mathbf{x}_k).$$

  The essential action of the Hessian is therefore in the direction of $\mathbf{w}_k = \mathbf{x}_{k+1} - \mathbf{x}_k$.
- Thus, equire

$$B_{k+1}\mathbf{w}_k = \mathbf{y}_k, \quad \mathbf{y}_k = \nabla \phi(\mathbf{x}_{k+1}) - \nabla \phi(\mathbf{x}_k).$$

- Obtain B_{k+1} as a positive definite rank-2 update of B_k, thus satisfying the above requirements.

Secant (quasi-Newton)

- Do not want to form or evaluate Hessian matrix explicitly; and *also*,
- want a positive definite B_k that is *easy to invert*.
- Note by Taylor's expansion

$$\boldsymbol{\nabla}\phi(\mathbf{x}_k) \approx \boldsymbol{\nabla}\phi(\mathbf{x}_{k+1}) - \nabla^2\phi(\mathbf{x}_{k+1})(\mathbf{x}_{k+1} - \mathbf{x}_k).$$

  The essential action of the Hessian is therefore in the direction of $\mathbf{w}_k = \mathbf{x}_{k+1} - \mathbf{x}_k$.
- Thus, equire

$$B_{k+1}\mathbf{w}_k = \mathbf{y}_k, \quad \mathbf{y}_k = \boldsymbol{\nabla}\phi(\mathbf{x}_{k+1}) - \boldsymbol{\nabla}\phi(\mathbf{x}_k).$$

- Obtain B_{k+1} as a positive definite rank-2 update of B_k, thus satisfying the above requirements.

BFGS method

Update $G_k = B_k^{-1}$ directly.

Choose $\mathbf{x}_0$ and G_0 (e.g., $G_0 = I$)

for $k = 0, 1, \ldots,$ until convergence

```
    $\mathbf{p}_k = -G_k \nabla_\phi(\mathbf{x}_k)$
    find a suitable step size $\alpha_k$
    $\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{p}_k$
    $\mathbf{w}_k = \alpha_k \mathbf{p}_k$
    $\mathbf{y}_k = \nabla_\phi(\mathbf{x}_{k+1}) - \nabla_\phi(\mathbf{x}_k)$
    $G_{k+1} = (I - \frac{\mathbf{w}_k \mathbf{y}_k^T}{\mathbf{y}_k^T \mathbf{w}_k}) G_k (I - \frac{\mathbf{y}_k \mathbf{w}_k^T}{\mathbf{y}_k^T \mathbf{w}_k}) + \frac{\mathbf{w}_k \mathbf{w}_k^T}{\mathbf{y}_k^T \mathbf{w}_k}.$
```

BFGS method

Update $G_k = B_k^{-1}$ directly.
Choose $\mathbf{x}_0$ and G_0 (e.g., $G_0 = I$)
for $k = 0, 1, \ldots,$ until convergence

$$
\begin{aligned}
&\mathbf{p}_k = -G_k \nabla_\phi(\mathbf{x}_k) \\
&\text{find a suitable step size } \alpha_k \\
&\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{p}_k \\
&\mathbf{w}_k = \alpha_k \mathbf{p}_k \\
&\mathbf{y}_k = \nabla_\phi(\mathbf{x}_{k+1}) - \nabla_\phi(\mathbf{x}_k) \\
&G_{k+1} = (I - \frac{\mathbf{w}_k \mathbf{y}_k^T}{\mathbf{y}_k^T \mathbf{w}_k}) G_k (I - \frac{\mathbf{y}_k \mathbf{w}_k^T}{\mathbf{y}_k^T \mathbf{w}_k}) + \frac{\mathbf{w}_k \mathbf{w}_k^T}{\mathbf{y}_k^T \mathbf{w}_k}.
\end{aligned}
$$

Secant (quasi-Newton) pros and cons

- Cheap update of $G_k = B_k^{-1}$ using rank-2 updates.
- Local superlinear convergence.
- Only descent directions because $G_k = B_k^{-1}$ symmetric positive definite.
- The method of choice for most problems.
- May lose Hessian sparsity, though.
- Limited memory versions L-BFGS exist for large, sparse matrices. They have their pros and cons...

Secant (quasi-Newton) pros and cons

- Cheap update of $G_k = B_k^{-1}$ using rank-2 updates.
- Local superlinear convergence.
- Only descent directions because $G_k = B_k^{-1}$ symmetric positive definite.
- The method of choice for most problems.
- May lose Hessian sparsity, though.
- Limited memory versions L-BFGS exist for large, sparse matrices. They have their pros and cons...

Inexact Newton, Newton-CG

- If the problem is large and has a sparse Hessian, may want to use Newton. But how to solve the linear system at each iteration?
- Use iterative method (Chapter 7) for the linear system at each iteration:

  apply preconditioned conjugate gradient iterations towards solving
  $\nabla^2\phi(\mathbf{x}_k)\ \mathbf{p}_k = -\nabla_\phi(\mathbf{x}_k)$ for $\mathbf{p}_k$
  set $\mathbf{x}_{k+1} = \mathbf{x}_k + \mathbf{p}_k$.

- Newton - outer iteration
  PCG - inner iteration
- How many inner iterations?! (intuitively, fewer when far, more when close) – leads to inexact Newton.

Nonlinear least squares

$$\min_{\mathbf{x}} \phi(\mathbf{x}) = \frac{1}{2}\|\mathbf{g}(\mathbf{x}) - \mathbf{b}\|^2,$$

where $\mathbf{b}$ is data (m values) and $\mathbf{g}$ a nonlinear function of n parameters $\mathbf{x}$.

$$A(\mathbf{x}) = \begin{pmatrix} \frac{\partial g_1}{\partial x_1} & \frac{\partial g_1}{\partial x_2} & \cdots & \frac{\partial g_1}{\partial x_n} \\ \frac{\partial g_2}{\partial x_1} & \frac{\partial g_2}{\partial x_2} & \cdots & \frac{\partial g_2}{\partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial g_{m-1}}{\partial x_1} & \frac{\partial g_{m-1}}{\partial x_2} & \cdots & \frac{\partial g_{m-1}}{\partial x_n} \\ \frac{\partial g_m}{\partial x_1} & \frac{\partial g_m}{\partial x_2} & \cdots & \frac{\partial g_m}{\partial x_n} \end{pmatrix},$$

Then

$$\boldsymbol{\nabla}\phi(\mathbf{x}) = A(\mathbf{x})^T(\mathbf{g}(\mathbf{x}) - \mathbf{b}).$$

Nonlinear least squares

$$\min_{\mathbf{x}} \phi(\mathbf{x}) = \frac{1}{2}\|\mathbf{g}(\mathbf{x}) - \mathbf{b}\|^2,$$

where $\mathbf{b}$ is data (m values) and $\mathbf{g}$ a nonlinear function of n parameters $\mathbf{x}$.

$$A(\mathbf{x}) = \begin{pmatrix} \frac{\partial g_1}{\partial x_1} & \frac{\partial g_1}{\partial x_2} & \cdots & \frac{\partial g_1}{\partial x_n} \\ \frac{\partial g_2}{\partial x_1} & \frac{\partial g_2}{\partial x_2} & \cdots & \frac{\partial g_2}{\partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial g_{m-1}}{\partial x_1} & \frac{\partial g_{m-1}}{\partial x_2} & \cdots & \frac{\partial g_{m-1}}{\partial x_n} \\ \frac{\partial g_m}{\partial x_1} & \frac{\partial g_m}{\partial x_2} & \cdots & \frac{\partial g_m}{\partial x_n} \end{pmatrix},$$

Then

$$\boldsymbol{\nabla}\phi(\mathbf{x}) = A(\mathbf{x})^T(\mathbf{g}(\mathbf{x}) - \mathbf{b}).$$

Gauss-Newton

Hessian matrix

$$\nabla^2\phi(\mathbf{x}) = A(\mathbf{x})^T A(\mathbf{x}) + L(\mathbf{x}),$$

where L is $n \times n$

$$L_{i,j} = \sum_{l=1}^{m} \frac{\partial^2 g_l}{\partial x_i \partial x_j}(g_l - b_l).$$

L can be ugly. Appears in Newton's method!

Gauss-Newton: drop the ugly term. Define iteration by

$$\begin{aligned}\left[A(\mathbf{x}_k)^T A(\mathbf{x}_k)\right]\mathbf{p}_k &= -\boldsymbol{\nabla}\phi(\mathbf{x}_k)\\ \mathbf{x}_{k+1} &= \mathbf{x}_k + \mathbf{p}_k.\end{aligned}$$

These are normal equations for

$$\min_{\mathbf{p}} \|A(\mathbf{x}_k)\mathbf{p} - (\mathbf{b} - \mathbf{g}(\mathbf{x}_k))\|,$$

so at each iteration solve a linear least squares problem (Chapter 6).

Gauss-Newton

Hessian matrix

$$\nabla^2 \phi(\mathbf{x}) = A(\mathbf{x})^T A(\mathbf{x}) + L(\mathbf{x}),$$

where L is $n \times n$

$$L_{i,j} = \sum_{l=1}^{m} \frac{\partial^2 g_l}{\partial x_i \partial x_j}(g_l - b_l).$$

L can be ugly. Appears in Newton's method!
Gauss-Newton: drop the ugly term. Define iteration by

$$\begin{aligned} \left[A(\mathbf{x}_k)^T A(\mathbf{x}_k)\right] \mathbf{p}_k &= -\nabla \phi(\mathbf{x}_k) \\ \mathbf{x}_{k+1} &= \mathbf{x}_k + \mathbf{p}_k. \end{aligned}$$

These are normal equations for

$$\min_{\mathbf{p}} \| A(\mathbf{x}_k)\mathbf{p} - (\mathbf{b} - \mathbf{g}(\mathbf{x}_k)) \|,$$

so at each iteration solve a linear least squares problem (Chapter 6).

Gauss-Newton vs. Newton

- The Gauss-Newton direction, unlike Newton's, is guaranteed to be a descent direction. This is because $A^T A$ is positive definite even when $A^T A + L$ is not.
- The Gauss-Newton iteration is cheaper and can be better conditioned (more stable) than Newton's iteration.
- The convergence order of Gauss-Newton is only linear, as the difference between it and Newton's iteration does not vanish in the limit.
- Gauss-Newton converges faster for problems where the model fits the data well! This is because then $\|\mathbf{g}(\mathbf{x}) - \mathbf{b}\|$ is "small" near the solution, hence L is small and Gauss-Newton is closer to Newton.

Outline

- Systems of nonlinear equations
- Unconstrained optimization
- *Constrained optimization

Constrained problem

- General form

$$\min_{\mathbf{x}\in\Omega} \quad \phi(\mathbf{x}), \quad \text{where}$$
$$\Omega = \{\mathbf{x} \in \mathbb{R}^n | c_i(\mathbf{x}) = 0,\ i \in \mathcal{E}, \quad c_i(\mathbf{x}) \geq 0,\ i \in \mathcal{I}\}.$$

- Equality constraints: reducing space; algebraic; domain Ω has empty interior.
- Inequality constraints: combinatorial; if $\mathcal{E}$ empty then domain Ω can have nonempty interior.
- Active set

$$\mathcal{A}(\mathbf{x}) = \mathcal{E} \cup \{i \in \mathcal{I} | c_i(\mathbf{x}) = 0\}.$$

Consider problems where $\mathcal{A}(\mathbf{x}^*)$ is nonempty.

KKT conditions for a minimum

- Assume constraint qualification.
- Lagrangian
$$\mathcal{L}(\mathbf{x}, \boldsymbol{\lambda}) = \phi(\mathbf{x}) - \sum_{i \in \mathcal{E} \cup \mathcal{I}} \lambda_i c_i(\mathbf{x}).$$
- KKT conditions necessary for a minimum:
$$\begin{aligned} \boldsymbol{\nabla}_{\mathbf{x}} \mathcal{L}(\mathbf{x}^*, \boldsymbol{\lambda}^*) &= \mathbf{0}, \\ c_i(\mathbf{x}^*) &= 0, \quad \forall i \in \mathcal{E}, \\ c_i(\mathbf{x}^*) &\geq 0, \quad \forall i \in \mathcal{I}, \\ \lambda_i^* &\geq 0, \quad \forall i \in \mathcal{I}, \\ \lambda_i^* c_i(\mathbf{x}^*) &= 0, \quad \forall i \in \mathcal{E} \cup \mathcal{I}. \end{aligned}$$

Active set methods

- Assuming solution is on $\partial\Omega$, search for the optimum along the boundary.
- For inequality constraints, keep track of $\mathcal{A}(\mathbf{x}_k)$, shuffling constraints in and out of the active set.
- e.g. quadratic programming (QP): quadratic objective function subject to linear inequality constraints.
- Sequential quadratic programming (SQP): At each iteration solve QP for search direction $\mathbf{p}_k$ at $(\mathbf{x}_k, \boldsymbol{\lambda}_k)$

$$\min_{\mathbf{p}} \quad \frac{1}{2}\mathbf{p}^T W_k \mathbf{p} + \nabla_\phi(\mathbf{x}_k)^T \mathbf{p},$$

$$c_i(\mathbf{x}_k) + \nabla \mathbf{c}_i(\mathbf{x}_k)^T \mathbf{p} = 0, \ i \in \mathcal{E}, \quad c_i(\mathbf{x}_k) + \nabla \mathbf{c}_i(\mathbf{x}_k)^T \mathbf{p} \geq 0, \ i \in \mathcal{I}.$$

Active set methods

- Assuming solution is on $\partial\Omega$, search for the optimum along the boundary.
- For inequality constraints, keep track of $\mathcal{A}(\mathbf{x}_k)$, shuffling constraints in and out of the active set.
- e.g. quadratic programming (QP): quadratic objective function subject to linear inequality constraints.
- Sequential quadratic programming (SQP): At each iteration solve QP for search direction $\mathbf{p}_k$ at $(\mathbf{x}_k, \boldsymbol{\lambda}_k)$

$$\min_{\mathbf{p}} \quad \frac{1}{2}\mathbf{p}^T W_k \mathbf{p} + \nabla_\phi(\mathbf{x}_k)^T \mathbf{p},$$

$$c_i(\mathbf{x}_k) + \boldsymbol{\nabla}\mathbf{c}_i(\mathbf{x}_k)^T\mathbf{p} = 0, \ i \in \mathcal{E}, \quad c_i(\mathbf{x}_k) + \boldsymbol{\nabla}\mathbf{c}_i(\mathbf{x}_k)^T\mathbf{p} \geq 0, \ i \in \mathcal{I}.$$

Active set methods

- Assuming solution is on $\partial\Omega$, search for the optimum along the boundary.
- For inequality constraints, keep track of $\mathcal{A}(\mathbf{x}_k)$, shuffling constraints in and out of the active set.
- e.g. quadratic programming (QP): quadratic objective function subject to linear inequality constraints.
- Sequential quadratic programming (SQP): At each iteration solve QP for search direction $\mathbf{p}_k$ at $(\mathbf{x}_k, \boldsymbol{\lambda}_k)$

$$\min_{\mathbf{p}} \quad \frac{1}{2}\mathbf{p}^T W_k \mathbf{p} + \nabla_\phi(\mathbf{x}_k)^T \mathbf{p},$$

$$c_i(\mathbf{x}_k) + \boldsymbol{\nabla}\mathbf{c}_i(\mathbf{x}_k)^T \mathbf{p} = 0,\ i \in \mathcal{E}, \quad c_i(\mathbf{x}_k) + \boldsymbol{\nabla}\mathbf{c}_i(\mathbf{x}_k)^T \mathbf{p} \geq 0,\ i \in \mathcal{I}.$$

Interior point and other methods

- Penalty methods

$$\min_{\mathbf{x}} \psi(\mathbf{x}, \mu) = \phi(\mathbf{x}) + \frac{1}{2\mu} \sum_{i \in \mathcal{E}} c_i^2(\mathbf{x}),$$

where $\mu \downarrow 0$.

- Barrier methods

$$\min_{\mathbf{x}} \psi(\mathbf{x}, \mu) = \phi(\mathbf{x}) - \mu \sum_{i \in \mathcal{I}} \log c_i(\mathbf{x}),$$

where $\mu \downarrow 0$.

- Augmented Lagrangian

$$\min_{\mathbf{x}} \psi(\mathbf{x}, \boldsymbol{\lambda}, \mu) = \phi(\mathbf{x}) - \sum_{i \in \mathcal{E}} \lambda_i c_i(\mathbf{x}) + \frac{1}{2\mu} \sum_{i \in \mathcal{E}} c_i^2(\mathbf{x}).$$

Given estimates $\boldsymbol{\lambda}_k, \ \mu_k$, solve the unconstrained minimization problem for $\mathbf{x} = \mathbf{x}_{k+1}$, then update the multipliers to $\boldsymbol{\lambda}_{k+1}, \ \mu_{k+1}$.

Interior point and other methods

- Penalty methods

$$\min_{\mathbf{x}} \psi(\mathbf{x}, \mu) = \phi(\mathbf{x}) + \frac{1}{2\mu} \sum_{i \in \mathcal{E}} c_i^2(\mathbf{x}),$$

where $\mu \downarrow 0$.

- Barrier methods

$$\min_{\mathbf{x}} \psi(\mathbf{x}, \mu) = \phi(\mathbf{x}) - \mu \sum_{i \in \mathcal{I}} \log c_i(\mathbf{x}),$$

where $\mu \downarrow 0$.

- Augmented Lagrangian

$$\min_{\mathbf{x}} \psi(\mathbf{x}, \boldsymbol{\lambda}, \mu) = \phi(\mathbf{x}) - \sum_{i \in \mathcal{E}} \lambda_i c_i(\mathbf{x}) + \frac{1}{2\mu} \sum_{i \in \mathcal{E}} c_i^2(\mathbf{x}).$$

Given estimates $\boldsymbol{\lambda}_k,\ \mu_k$, solve the unconstrained minimization problem for $\mathbf{x} = \mathbf{x}_{k+1}$, then update the multipliers to $\boldsymbol{\lambda}_{k+1},\ \mu_{k+1}$.

Interior point and other methods

- Penalty methods

$$\min_{\mathbf{x}} \psi(\mathbf{x}, \mu) = \phi(\mathbf{x}) + \frac{1}{2\mu} \sum_{i \in \mathcal{E}} c_i^2(\mathbf{x}),$$

  where $\mu \downarrow 0$.

- Barrier methods

$$\min_{\mathbf{x}} \psi(\mathbf{x}, \mu) = \phi(\mathbf{x}) - \mu \sum_{i \in \mathcal{I}} \log c_i(\mathbf{x}),$$

  where $\mu \downarrow 0$.

- Augmented Lagrangian

$$\min_{\mathbf{x}} \psi(\mathbf{x}, \boldsymbol{\lambda}, \mu) = \phi(\mathbf{x}) - \sum_{i \in \mathcal{E}} \lambda_i c_i(\mathbf{x}) + \frac{1}{2\mu} \sum_{i \in \mathcal{E}} c_i^2(\mathbf{x}).$$

  Given estimates $\boldsymbol{\lambda}_k,\ \mu_k$, solve the unconstrained minimization problem for $\mathbf{x} = \mathbf{x}_{k+1}$, then update the multipliers to $\boldsymbol{\lambda}_{k+1},\ \mu_{k+1}$.