Lecture 5: Systems of nonlinear equations and optimization

CSC 338: Numerical Methods

Ray Wu

University of Toronto

February 8, 2023

Overview

- This lecture extends Lecture 2, where we considered methods of solving nonlinear equations in one variable.
- First, study solving systems of nonlinear equations, then, the related topic of optimization.

Nonlinear Systems of Equations

- Nonlinear Systems of Equations are defined by

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

  or, in vector form, simply $f(x) = 0$.
- Recall the following methods for one-dimensional problems:
  - Bisection Method
  - Fixed-point iteration
  - Newton's Method
  - Secant Method
- The last three methods all have a multidimensional analogue, but we will focus on Newton's method.

Multidimensional Taylor Series

- First, we need to review the concept of a derivative of a vector-valued function
- The **Taylor Expansion** of a function $f : R^n \to R^m$ gives

$$f(x+p) = f(x) + J(x)p + \mathcal{O}(\|p\|^2) \tag{1}$$

where J is the **Jacobian matrix** of first derivatives:

$$J(x) = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & \cdots & \frac{\partial f_2}{\partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial f_m}{\partial x_1} & \frac{\partial f_m}{\partial x_2} & \cdots & \frac{\partial f_m}{\partial x_n} \end{bmatrix} \tag{2}$$

Derivation of Newton's Method

- By Taylor series,

$$f(x+p) = f(x) + J(x)p + \mathcal{O}(\|p\|^2) \tag{3}$$

  Replace $x+p$ with x^* and x with x_k to get

$$f(x^*) = f(x_k) + J(x_k)(x^* - x_k) + \mathcal{O}(\|x^* - x_k\|^2) \tag{4}$$

- When $x = x^*$, $f(x) = 0$. Sub this in, and drop the $\mathcal{O}(h^2)$ term (linearization)

$$0 = f(x_k) + J(x_k)(x^* - x_k) \tag{5}$$

- Define x^* to be the next iterate x_{k+1}.
- Algorithm: on each iterate,
  - Calculate $p = x_{k+1} - x_k$ by solving $J(x_k)(x_{k+1} - x_k) = -f(x_k)$
  - Calculate $x_{k+1} = x_k + p$

When Does Newton's Method fail?

- One situation: When initial guess is too far away.
- Another situation: When Jacobian matrix $J(x)$ is singular.

One dimension vs Multiple dimensions

- Recall in one dimension, Newton's method gives

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)} \tag{6}$$

- In multiple dimensions,

$$x_{k+1} = x_k - J^{-1}(x_k)f(x_k) \tag{7}$$

  - Newton's method is exactly the same
  - To compute $J^{-1}(x_k)f(x_k)$ we solve a linear system, and do not invert the matrix.
  - It is almost never a good idea to invert a matrix.

Example 9.3 - Ascher & Greif - 1

Consider the nonlinear differential equation

$$u'' + \exp(u) = 0, \quad 0 < t < 1 \tag{8}$$

with boundary conditions $u(0) = u(1) = 0$.

- **Discretization**: partition the interval $[0, 1]$ into n equal subintervals at $t_1, t_2, \dots, t_{n-1}$.
  - Unknowns are now real numbers $u_1, u_2, \dots, u_{n-1}$.
  - Let $h = 1/n = t_i - t_{i-1}$
- Apply **finite difference** for all i:

$$f_i(u) \equiv \frac{u_{i+1} - 2u_i + u_{i-1}}{h^2} + \exp(u_i) = 0 \tag{9}$$

- What is the Jacobian matrix of f?

Example 9.3 - Ascher & Greif - 2

- System of equations:

$$f_i(u) \equiv \frac{u_{i+1} - 2u_i + u_{i-1}}{h^2} + \exp(u_i) = 0 \tag{10}$$

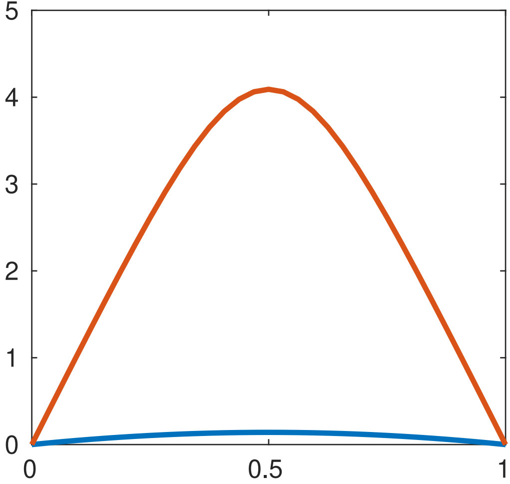
- Jacobian:

$$J_{i,j} = \frac{\partial f_i}{\partial u_j} = \begin{cases} 1/h^2 & \text{if } j = i - 1 \\ -2/h^2 + \exp(u_i) & \text{if } j = i \\ 1/h^2 & \text{if } j = i + 1 \end{cases} \tag{11}$$

- Initial guess: Let's choose $\alpha t(1 - t)$.
  - We know the boundaries are zero
  - Can scale up/down as we want.

Matlab implementation

- Show matlab implementation (L05.m)
- The two solutions to the nonlinear differential equation:



Unconstrained optimization

- The unconstrained optimization problem is given by

$$\min \phi(x), x \in \mathbb{R}^n \tag{12}$$

- Define the **gradient** of ϕ to be

$$\nabla \phi(x) = \begin{bmatrix} \frac{\partial \phi}{\partial x_1} \\ \frac{\partial \phi}{\partial x_2} \\ \ldots \\ \frac{\partial \phi}{\partial x_n} \end{bmatrix} \tag{13}$$

- Define the **Hessian** to be the Jacobian of the gradient.
- Gradient and Hessian are the n-dimensional analogues of first and second derivatives of a one-variable function.

Necessary and Sufficient conditions for minimum

- Recall from Calc 1, the critical points of a function are when its derivative are zero. In n dimensions this means

$$\nabla \phi(x) = 0 \tag{14}$$

- Recall again from Calc 1, the second derivative test: if the function has a positive second derivative at a critical point, that critical point is a local minimum. In n dimensions this means that if the Hessian $\nabla^2 \phi(x)$ is positive-definite, we have a local minimum.
- Positive definite: if a matrix A is positive definite, then for **any** nonzero vector x, $x^T A x > 0$.

Nonlinear Systems vs Optimization

- Nonlinear systems: Solve

$$f(x) = 0 \iff \min \|f(x)\| \tag{15}$$

- Optimization

$$\min \phi(x) \iff \nabla\phi(x) = 0 \tag{16}$$

- The relationship between solving nonlinear systems and optimization:
  - We solve the left-hand side, but the problem can be cast as the right-hand side.

- Newton's Method (for optimization) is the same as Newton's method for nonlinear equations: Solve the system

$$\nabla \phi(x) = 0. \tag{17}$$

- The iteration becomes
  - Calculate $p = x_{k+1} - x_k$ by solving $\nabla^2 \phi(x_k)(x_{k+1} - x_k) = -\nabla \phi(x_k)$
  - Calculate $x_{k+1} = x_k + p$
- Advantages: second order convergence
- Disadvantages: requires Hessian, requires solving linear systems, linear systems may not be positive definite, etc.

Descent Directions

- At a point x, the vector p is a descent direction if

$$\nabla\phi(x)^T p < 0 \tag{18}$$

  - Note that $\nabla\phi(x)^T p$ is the *directional derivative*.
- If a step is small enough, then the objective function will decrease in a descent direction:

$$\phi(x + \alpha p) < \phi(x) \text{ for a small enough } \alpha \tag{19}$$

- Hence, as long as we can compute a descent direction, we can construct an iterative method that is guaranteed to decrease the function value.

Gradient Descent

- Gradient Descent chooses $p_k = -\nabla\phi(x_k)$.
- Guaranteed to be a descent direction (by norms of vectors):

$$-\nabla\phi(x_k)^T\nabla\phi(x_k) < 0 \tag{20}$$

- In fact, also the steepest descent, hence its alternative name.
  - An analogy is that if you place a ball at x_k, it will roll in the steepest direction – the direction of the negative gradient.
- Drawbacks: no 2nd order information used, convergence can be slow.

Line Search

- Gradient descent can take too large of a stepsize; the guarantees for descent may not extend all the way from x_k to $x_k + p_k$.
- Instead, consider the update

$$x_{k+1} = x_k + \alpha_k p_k \tag{21}$$

- For pure methods, $\alpha_k = 1$. For constant stepsize, $\alpha_k = c$ for some constant c.
- For a small enough α, we have

$$\phi(x_k + \alpha_k p_k) < \phi(x_k) \tag{22}$$

- Line search addresses the question of what value of α is appropriate.

Line Search techniques

- We look along the line $x_k + \alpha_k p_k$ such that

$$\phi(x_k + \alpha_k p_k) \leq \phi(x_k) + \sigma \alpha_k \nabla \phi(x_k)^T p_k \tag{23}$$

  This is known as the first Wolfe condition, also known as a "sufficient decrease condition".
  - Typically, $\sigma = 10^{-4}$.
- Backtracking line search: Start with $\alpha = 1$, if the Wolfe condition is not satisfied, halve it, etc.
- Exact line search: Find α such that

$$\phi(x_k + \alpha p_k) \tag{24}$$

  is minimized.

Revisiting Newton's Method

- Newton's method chooses the descent direction p_k

$$p_k = -[\nabla^2 \phi(x_k)]^{-1} \nabla \phi(x_k) \tag{25}$$

- What happens when we test $\nabla \phi(x_k)^T p_k < 0$?
  - Let B denote the matrix $\nabla^2 \phi(x_k)$, and we see that if B is positive-definite, its inverse is also positive-definite, then we have a descent direction.
  - Recall that positive-definite means for any nonzero vector x, $x^T B x > 0$.
  - Gradient Descent chooses $B = I$.
- What do we do if B is **not** positive-definite? – Quasi-Newton methods.

Inexact Newton Methods

- Recall that we want the matrix B_k from the previous slide to be positive-definite.
- Suppose B_k has some positive and some negative eigenvalues, list them in order of $\lambda_1 > \lambda_2 > \cdots > 0 > \cdots > \lambda_n$.
- The matrix $B_k + \mu I$ has eigenvalues $\lambda_i + \mu$.
- Idea: Choose μ large enough to move the eigenvalues postive, and we have a descent direction.

Quasi-Newton Methods – BFGS

- Start with an estimation of $G = B^{-1}$, apply updates based on gradient information every iteration.
- Superlinear convergence
- Positive-definite property of G_k is retained every iteration
- Considered to be the method of choice for most problems.
- Limited-memory versions L-BFGS exist for very large, sparse matrices.

Constrained optimization

- Constrained optimization problems have the following form:

$$\min \phi(x), \text{ subject to } c_i(x) \geq 0. \tag{26}$$

- c_i is the constraint function.
- No equality constraints, because if we have $c_j(x) = 0$, $c_j(x) \geq 0$ and $-c_j(x) \geq 0$ impose the same condition.
- In general, we like our problems to be defined as simple as possible, and avoid unnecessary families of constraints.

Constrained optimization – penalty and barrier methods

- Penalty and Barrier methods are among the simplest methods for solving constrained optimization problems
- Convert constrained optimization problem to unconstrained optimization problem.
- Penalty methods, as the name suggests, *penalize* (in the value of the objective function) solutions that violate the constraint $c_i(x) \geq 0$.

$$\min \psi(x) = \phi(x) + \mu \sum c_i^2(x) \tag{27}$$

- Barrier methods, on the other hand, introduce terms that prevent the constraints from being violated.

$$\min \psi(x) = \phi(x) - \mu \sum \log c_i(x) \tag{28}$$

Beyond this course

- Linear Optimization (Linear Programming): linear objective function, linear constrants. Use simplex method/IPM.
- Karush-Kuhn-Tucker (KKT) conditions: necessary first-order conditions for a minimum in constrained optimization
- Example paper: Fast Energy Projection
- Underdetermined systems: Solving

$$Ax = b \tag{29}$$

when A is not full rank. The obvious answer is to

$$\min \|x\|_2 \tag{30}$$

subject to

$$Ax = b \tag{31}$$

- But also

$$\min \|x\|_1 \tag{32}$$

is of interest, since minimization in 1-norms lead to sparse solutions.