Numerical Analysis Notes

Zoe Farmer

February 26, 2024

1 Solution of Nonlinear Equations

1.1 Bisection Method

We start by introducing the Bisection method for solving equations. The idea is straightforward, pick two points, a and b, one on each side of the root such that $f(a)f(b) < 0$. We then determine $c = \frac{1}{2}(a+b)$ and test if $f(a)f(c) < 0$ if it is, then we rename c as b. Otherwise the opposite must be true meaning we rename c as a.

```
def bisectionmethod(f, a, b, steps):
    m = np.zeros((steps, 2))
    m[0, 0] = a
    m[0, 1] = b
    for i in range(1, steps):
        c = (m[i - 1, 0] + m[i - 1, 1]) / 2
        if f(c) == 0:
            return c
        elif f(a) * f(c) < 0:
            m[i, 0] = m[i - 1, 0]
            m[i, 1] = c
        else:
            m[i, 0] = c
            m[i, 1] = m[i - 1, 1]
    return (m[-1, 0] + m[-1, 1]) / 2
```

Theorem 1. *If $[a_0, b_0], [a_1, b_1], \ldots, [a_n, b_n], \ldots$ denote the intervals in the bisection method, then the limits $\lim_{n\to\infty} a_n$ and $\lim_{nto\infty} b_n$ exist, are equal, and represent a zero of f. If $r = \lim n \to \infty c_n$ and $c_n = \frac{1}{2}(a_n + b_n)$, then*

$$|r - c_n| \leq 2^{-(n+1)}(b_0 - a_0)$$

1.2 Newton's Method

Newton's Method is defined as the following. We begin with some estimate x_0 of r and define inductively

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$$

This has quadratic convergence.

Theorem 2. *Let f'' be continuous and let r be a simple zero of f. Then there is a neighborhood of r and a constant C such that if Newton's method is started in that neighborhood, the successive points become steadily closer to r and satisfy*

$$|x_{n+1} - r| \leq C(x_n - r)^2 \qquad (n \geq 0)$$

Theorem 3. *If f belongs to $C^2(\mathbb{R})$, is increasing, is convex, and has a zero, then the zero is unique, and the Newton iteration will converge to it from any starting point.*

1.3 Secant Method

To fix an issue with Newton's iteration, namely the need for a derivative, we can use the Secant method which instead uses the secant line between two points.

$$x_{n+1} = x_n - f(x_n)\left[\frac{x_n - x_{n-1}}{f(x_n) - f(x_{n-1})}\right]$$

The convergence is superlinear.

1.4 Fixed Points and Functional Iteration

The above methods can be referred to as functional iteration. Functional iteration sequences have fixed points defined by $f(x^*) = x^*$. This point is referred to as the sequence's fixed point.

A mapping or function F is said to be contractive if there exists a number λ less than 1 such that

$$|F(x) - F(y)| \leq \lambda\,|x - y|$$

Theorem 4 (Contractive Mapping Theorem)**.** *Let C be a closed subset of the real line. If F is a contractive mapping of C into C, then F has a unique fixed point. Moreover, this fixed point is the limit of every sequence obtained from the above equation with a starting point $x_0 \in C$.*

1.4.1 Error Analysis

The order of convergence is defined to be the largest real number q such that the limit

$$\lim_{n\to\infty} \frac{|x_{n+1} - s|}{|x_n - s|^q}$$

exists and is non-zero.

2 Solving Systems of Linear Equations

2.1 LU and Cholesky Factorizations

A diagonal matrix is any matrix such that all the nonzero elements are on the main diagonal.

A lower or upper triangular matrix has only nonzero elements on the main diagonal, and then above or below depending on upper or lower.

Such a matrix can be solved using forward or backward substitution.

Suppose some matrix A can be factored into some product LU. We call this LU-decomposition.

Theorem 5. *If all n leading principal minors of the $n \times n$ matrix A are nonsingular, then A has an LU-decomposition.*

2.1.1 Cholesky Factorization

Theorem 6. *If A is a real, symmetric, and positive definite matrix, then it has a unique factorization, $A = LL^T$, in which L is lower triangular with a positive diagonal.*

The algorithm is as follows.

```
def cholesky(n, A):
    l = matrix(n, n)
    for k in range(1, n):
        l[k, k] = (A[k, k] -
            sum([l[k, s]^2
            for s in range(1, k - 1)]))^(1/2)
        for i in range(k + 1, n):
            l[i, k] = (A[i, k] -
                sum([l[i, s] * l[k, s]
                for s in range(1, k - 1)])) / l[k, k]
    return l
```

2.2 Pivoting and Constructing an Algorithm

In simple Gaussian elimination, a pivot element is the first non-zero element in the row.

Theorem 7. *If all the pivot elements $a_{kk}^{(k)}$ are nonzero in the gaussian elimination process, then $A = LU$.*

We find that in the simple Gaussian algorithm we sometimes cannot solve an otherwise easy system. This means that we need to include pivoting, which is merely switching around rows (and sometimes columns) in order to transform the matrix.

Described below is Gaussian Elimination with scaled row pivoting. For some $n \times n$ system we have two parts. First is the factorization phase which generates the factorization LU. Second is the solution phase which actually solves the permuted and factored system.

```
def factorization(n, A):
    for i in range(n):
        p[i] = i
        s[i] = max(A[i]) # max of row
    for k in range(n - 1):
        find j >= k such that:
            abs(A[p[j], k]) / s[p[j]] >= abs(A[p[i], k]) /
                                         s[p[i]] for i = k, k+1...
        p[k], p[j] = p[j], p[k]
```

```
            for i in range(k + 1, n):
                z = A[p[i], k] / A[p[k], k]
                A[p[i], k] = z
                for j in range(k + 1, n):
                    A[p[i], j] = A[p[i], j] - z * A[p[k], j]
        return A, p

    def solution(n, a, p, b):
        for k in range(n - 1):
            for i in range(k + 1, n):
                b[p[i]] = b[p[i]] - a[p[i], k] * b[p[k]]
        for i in range(n - 1, -1, -1):
            x[i] = (b[p[i]] - Sum(A[p[i], j] * x[j],
                                  j=i+1, n)) / A[p[i], i]
```

We can also use complete pivoting.

Theorem 8 (*LU* Factorization of *PA*)**.** *Define a permutation matrix P whose elements are $P_{ij} = \delta_{p_i j}$. Define an upper triangular matrix U whose elements are $u_{ij} = a_{p_i j}^{(n)}$ if $j \geq i$. Define a unit lower triangular matrix L whose elements are $l_{ij} = a_{p_i j}^{(n)}$ if $j < i$. Then $PA = LU$.*

Theorem 9 (Solving $PA = LU$)**.** *If the factorization $PA = LU$ is produced from the Gaussian algorithm with scaled row pivoting, then the solution of $Ax = b$ is obtained by first solving $Lz = Pb$ and then solving $Ux = z$. Similarly, the solution of $y^T A = c^T$ is obtained by solving $U^T z = c$ and then $L^T P y = z$.*

Theorem 10 (Long Operations)**.** *If Gaussian elimination is used with scaled row pivoting, then the solution of the system $Ax = b$, with fixed A, and m different vectors b, involved approximately*

$$\frac{1}{3}n^3 + \left(\frac{1}{2} + m\right)n^2$$

long operations (multiplications and divisions).

2.3 Diagonally Dominant Matrices

These are defined as

$$|a_{ii}| > \sum_{j=1, j\neq i}^{n} |a_{ij}| \qquad (1 \leq i \leq n)$$

In essence these are matrices in which the diagonal elements are greater than any other element in the matrix.

Theorem 11 (Preserving Diagonal Dominance)**.** *Gaussian elimination without pivoting preserves the diagonal dominance of a matrix.*

Theorem 12 (First Corollary on Diagonally Dominant Matrix)**.** *Every diagonally dominant matrix is nonsingular and has an LU-factorization.*

Theorem 13 (Second Corollary on Diagonally Dominant Matrix)**.** *If the scaled row pivoting version of Gaussian elimination recomputes the scale array after each major step and is applied to a diagonally dominant matrix, then the pivots will be the natural ones:* $1, 2, \ldots, n$*. Hence, the work of choosing the pivots can be omitted in this case.*

2.4 Tridiagonal System

A square matrix is said to be tridiagonal if $a_{ij} = 0$ for all pairs (i, j) that satisfy $|i - j| > 1$. This is like the following:

$$\begin{bmatrix} d_1 & c_1 & & & & & & \\ a_1 & d_2 & c_2 & & & & & \\ & a_2 & d_3 & c_3 & & & & \\ & & a_3 & d_4 & c_4 & & & \\ & & & a_4 & d_5 & c_5 & & \\ & & & & a_5 & d_6 & c_6 & \\ & & & & & a_6 & d_7 & c_7 \end{bmatrix}$$

3 Norms and the Analysis of Errors

On a vector space V, a norm is a function $\|\cdot\|$ from V to the set of nonnegative reals that obeys these three postulates:

- $||\mathbf{x}|| > 0$ if $x \neq 0, x \in V$
- $||\lambda\mathbf{x}|| = |\lambda|\,||\mathbf{x}||$ if $\lambda \in \mathbb{R}, x \in V$
- $||\mathbf{x} + \mathbf{y}|| \leq ||\mathbf{x}|| + ||\mathbf{y}||$ if $x, y \in V$ (triangle inequality)

A norm can be thought of as a length of magnitude of a vector. The most familiar norm on $\mathbb{R}^n$ is the Euclidean l_2-norm defined by

$$||\mathbf{x}||_2 = \left(\sum_{i=1}^{n} x_i^2\right)^{1/2} \qquad \text{where } x = (x_1, \ldots, x_n)^T$$

Other norms are also used, including the l_∞ norm.

$$||\mathbf{x}||_\infty = \max_{1 \leq i \leq n} |x_i|$$

As well as the l_1 norm.

$$||\mathbf{x}||_1 = \sum_{i=1}^{n} |x_i|$$

3.1 Matrix Norm

If some norm $||\cdot||$ has been specified, then the matrix norm subordinate is defined by

$$||\mathbf{A}|| = \sup\{||\mathbf{A}\mathbf{u}|| : u \in \mathbb{R}^n, ||\mathbf{u}|| = 1\}$$

We can use the triangle inequality and the vector norm properties to write

$$||\mathbf{A}+\mathbf{B}|| = ||\mathbf{A}|| + ||\mathbf{B}||$$

As well as

$$||\mathbf{Ax}|| \leq ||\mathbf{A}||||\mathbf{x}|| \qquad (x \in \mathbb{R}^n)$$

Theorem 14 (Infinity Matrix Norm)**.** *If the vector norm $||\cdot||_\infty$ is defined by*

$$||\mathbf{x}||_\infty = \max_{1\leq i\leq n} |x_i|$$

then its subordinate matrix norm is given by

$$||\mathbf{A}||_\infty = \max_{1\leq i\leq n} \sum_{j=1}^{n} |a_{ij}|$$

(In essence, the max absolute row sum.)

A matrix norm subordinate to a vector norm has additional properties:

$$||\mathbf{I}|| = 1$$
$$||\mathbf{AB}|| \leq ||\mathbf{A}||||\mathbf{B}||$$

3.2 Condition Number

Let's consider an equation

$$Ax = b$$

We define the condition number of matrix A to be

$$\kappa(A) \equiv ||\mathbf{A}|| \cdot ||\mathbf{A^{-1}}||$$

If the condition number is especially high we consider it to be ill-conditioned, otherwise it is well conditioned.

Theorem 15 (Bounds Involving Condition Number)**.** *In solving systems of equations $Ax = b$, the condition number $\kappa(A)$, the residual vector r, and the error vector e satisfy the following inequality:*

$$\frac{1}{\kappa(A)}\frac{||\mathbf{r}||}{||\mathbf{b}||} \leq \frac{||\mathbf{e}||}{||\mathbf{x}||} \leq \kappa(A)\frac{||\mathbf{r}||}{||\mathbf{b}||}$$

4 Neumann Series and Iterative Refinement

Theorem 16 (Neumann Series)**.** *If A is an $n \times n$ matrix such that $||\mathbf{A}|| < 1$, then $I - A$ is invertible, and*

$$(I-A)^{-1} = \sum_{k=0}^{\infty} A^k$$

From this follows

$$||(\mathbf{I}-\mathbf{A})^{-\mathbf{1}}|| \leq \sum_{k=0}^{\infty}||\mathbf{A^k}|| \leq \sum_{k=0}^{\infty}||\mathbf{A}||^k = \frac{1}{1-||\mathbf{A}||}$$

Theorem 17 (Invertible Matrices)**.** *If A and B are $n \times n$ matrices such that $||\mathbf{I}-\mathbf{AB}|| < 1$, then A and B are invertible. Furthermore, we have*

$$A^{-1} = B\sum_{k=0}^{\infty}(I-AB)^k \text{ and } B^{-1} = \sum_{k=0}^{\infty}(I-AB)^k A$$

Theorem 18 (Iterative Improvement)**.** *If $||\mathbf{I}-\mathbf{BA}|| < 1$, then the method of iterative improvement given by*

$$x^{(k+1)} = x^{(k)} + B(b-Ax^{(k)}) \quad (k \geq 0)$$

produces the sequence of vectors

$$x^{(m)} = B\sum_{k=0}^{m}(I-AB)^k b \quad (m \geq 0)$$

These are the partial sums in the following equation:

$$x = B\sum_{k=0}^{\infty}(I-AB)^k b$$

And therefore converge to x.

5 Orthogonal Factorizations and Least Squares Problems

An indexed set of vectors in $\mathbb{C}^n$ are said to be orthogonal if the inner product for any corresponding vectors is equal to zero. It is said to be orthonormal if $\langle v_i, v_j\rangle = \delta_{ij}$.

An inner product space is a system in which $\mathbb{C}^n$ has one embodiment. It is a linear space over the complex field in which the inner product follows these axioms.

- $\langle x, x\rangle > 0$ if $x \neq 0$
- $\langle \alpha x + \beta y, z\rangle = \alpha\langle x, z\rangle + \beta\langle y, z\rangle \qquad \alpha, \beta \in \mathbb{C}$
- $\langle x, y\rangle = \overline{\langle y, x\rangle}$

The Pythagorean Rule applies to any inner-product space.

5.1 The Gram-Schmidt Process

This classic process can be used to obtain orthonormal systems in any inner-product space. If we are given a linearly independent sequence of vectors, $[x_1, x_2, \ldots]$. We can generate an orthonormal sequence by the formula

$$u_k = \left\| x_k - \sum_{i<k}\langle x_k, u_i\rangle u_i \right\|_2^{-1} \left(x_k - \sum_{i<k}\langle x_k, u_i\rangle u_i \right) \qquad (k \geq 1)$$

Theorem 19 (Gram-Schmidt Sequence)**.** *The* ***Gram-Schmidt Sequence*** $[u_1, u_2, \ldots]$ *has the property that* $[u_1, u_2, \ldots, u_k]$ *is an orthonormal base for the linear span of* $\{x_1, x_2, \ldots x_k\}$ *for* $k \geq 1$*.*

Theorem 20 (Gram-Schmidt Factorization)**.** *The Gram-Schmidt process, when applied to the columns of an* $m \times n$ *matrix* A *of rank* n*, produces a factorization*

$$A = BT$$

in which B *is an* $m \times n$ *matrix with orthonormal columns and* T *is an* $n \times n$ *upper triangular matrix with positive diagonal.*

If we modify our process a little, we can get better results.