AMSC460 Computational Methods

Michael Li

Contents

1 Introduction

1.1 Review of Calculus

- A function f has a **limit** L at x_0, written $\lim_{x\to x_0} f(x) = L$, if for an arbitrary $\epsilon > 0$, there exists a $\delta > 0$ such that

$$|x - x_0| < \delta \implies |f(x) - L| < \epsilon$$

- f is **continuous** at x_0 if

$$\lim_{x\to x_0} f(x) = f(x_0)$$

- A sequence $\{x_n\}$ **converges** to x if for an arbitrary $\epsilon > 0$, there exists a positive N such that for $n \geq N$,

$$|x_n - x| < \epsilon$$

- **Note**: the following statements are equivalent for a function f
  - f is continuous at x_0
  - if $\{x_n\}$ converges to x_0 then $\lim_{n\to\infty} f(x_n) = f(x_0)$
- f is **differentiable** at x_0 if

$$f'(x_0) = \lim_{x\to x_0} \frac{f(x) - f(x_0)}{x - x_0}$$

  - **Note**: if f is differentiable at x_0, then f is continuous at x_0
- **Rolle's Theorem**: suppose $f \in C[a, b]$ and f is differentiable on (a, b) and $f(a) = f(b)$. Then there is a number $c \in (a, b)$ such that $f'(c) = 0$
- **Mean Value Theorem**: suppose $f \in C[a, b]$ and f is differentiable on (a, b), then there is a number $c \in (a, b)$ s.t.

$$f'(c) = \frac{f(b) - f(a)}{b - a}$$

- **Extreme Value Theorem**: suppose $f \in C[a, b]$, $c_1, c_2 \in [a, b]$, and $f(c_1) \leq f(x) \leq c_2$ for all $x \in [a, b]$. If f is differentiable on (a, b) then c_1, c_2 either occur at the endpoints $[a, b]$ or where $f' = 0$
- **Riemann Integral**: $\int_a^b f(x)\,dx = \lim_{\max \Delta x_i} \to 0 \sum_{i=1}^n f(z_i)\Delta x_i$
- **Weighted Mean Value Theorem for Intergrals**: suppose f, g exist on $[a, b]$ and $g(x)$ doesn't change sign on $[a, b]$. Then there exists a $c \in (a, b)$ with

$$\int_a^b f(x)g(x)\,dx = f(c)\int_a^b g(x)\,dx$$

  - When $g(x) = 1$, we have MVT for integrals:

$$f(c) = \frac{1}{b - a}\int_a^b f(x)\,dx$$

- **Taylor's Theorem**: suppose $f \in C^n[a, b]$, $f^{(n+1)}$ exists on $[a, b]$, $x_0 \in [a, b]$. Then for every $x \in [a, b]$, there exists a z between x_0 and x such that

$$f(x) = P_n(x) + R_n(x)$$

  - **nth Taylor Polynomial**: $P_n(x) = f(x_0) + f'(x_0)(x - x_0) + \ldots + \frac{f^{(n)}(x_0)}{n!}(x - x_0)^n$
  - **Remainder Term**: $R_n(x) = \frac{f^{(n+1)}(z)}{(n+1)!}(x - x_0)^{n+1}$

1.2 Computer Arithmetic

Round-off Error: error produced when a computer uses arithmetic with only a finite number of digits

1.2.1 Binary Machine Numbers

Real numbers are represent as 64-bits with

- first bit is a sign indicator, denoted s
  - 0 positive, 1 negative
- 11-bit exponent called the **characteristic** and denoted c
  - Gives range of exponent $[-1023, 1024]$
- 52-bit binary fraction called the **mantissa** denoted f
- Floating-point numbers are of the form: $(-1)^s 2^{c-1023}(1+f)$
- Smallest positive number has $s = 0, c = 1, f = 0 \implies 2^{-1022}(1+0) \approx 0.22251 \times 10^{-307}$
  - Numbers that have magnitude less than this result in **underflow** are generally set to 0
- Largest positive number has $s = 0, c = 2046, f = 1 - 2^{-52} \implies 2^{1023}(2 - 2^{-52}) \approx 0.17977 \times 10^{39}$
  - Numbers that have magnitude greater than this result in **overflow** and usually cause computations to stop

1.2.2 Decimal Machine Numbers

Numbers are represented as k-digit **decimal machine numbers**: $\pm 0.d_1 d_2 \dots d_k \times 10^n$

- floating point form is obtained by terminating the mantissa of the number at k decimal digits. 2 ways of doing this
  - **Chopping**: chop off digits $d_{k+1} d_{k+2} \dots$
  - **Rounding**: round d_k based on the value of d_{k+1}

Suppose p^* is an approximation of p. Then

- **Absolute Error**: $|p - p^*|$
- **Relative Error**: $\frac{|p-p^*|}{|p|}$ if $p \neq 0$

p^* is said to approximate p to t **significant digits** if t is the largest nonnegative integer for which

$$\frac{|p - p^*|}{|p|} \leq 5 \times 10^{-t}$$

2 Solution of Equations in One Variable

2.1 Bisection Method

Suppose f is a cont function defined on $[a, b]$ with $f(a)$ and $f(b)$ of opposite signs. By IVT, there is a $p \in (a, b)$ with $f(p) = 0$

```
INPUT: endpoints a,b; tolerance TOL, max iterations N_0
OUTPUT: approximate solution f(p) = 0 or FAILURE

i = 1;
FA = f(a);
while i ≤ N_0:
  p = a + (b-a)/2;
  FP = f(p);
  if FP = 0 or (b-a)/2 < TOL:
```

```
    output(p);
    STOP;
  i = i + 1;
  if FA · FP > 0:
    a = p;
    FA = FP;
  else:
    b = p;
    (FA unchanged)
Output FAILURE;
```

Issues:

- there are some sequences where $p_n - p_{n-1} \to 0$ but the sequence diverges
- situations where $f(p_n) \approx 0$ while p_n is far from p
- relatively slow to converge
- good intermediate approximation might be accidently discarded

Benefits:

- always converges to a solution

Has a rate of convergence of $O(\frac{1}{2^n})$: $p_n = p + O(\frac{1}{2^n})$

2.2 Fixed-Point Iteration

Fixed point: a point p such that $f(p) = p$. Fixed-point and root-finding are similar in that:

- given a root problem $f(p) = 0$, we can define a g with a fixed point at p: $g(x) = x - f(x)$ or $g(x) = x + f(x)$
  - If g has a fixed point at p, then $f(x) = x - g(x)$ has a zero at p

Theorem: conditions for existence and uniqueness of a fixed point:

- if $g \in C[a, b]$ and $g(x) \in [a, b]$ for all $x \in [a, b]$ then g has a least 1 fixed point in $[a, b]$
- if $g'(x)$ exists on (a, b) and a positive $k < 1$ exists such that $|g'(x)| \leq k$ for all $x \in (a, b)$ then there is exactly 1 fixed point in $[a, b]$

Algorithm: idea is to find p using $p_n = g(p_{n-1})$

```
INPUT: p_0 initial, TOL, N_0
OUTPUT: p or FAILURE

i = 1;
while i ≤ N_0:
  p = g(p_0);
  if |p - P_0| < TOL;
    OUTPUT p;
    STOP;
  i += 1;
  p_0 = p
Output FAILURE
```

Theorem: let $g \in C[a, b]$, $g(x) \in [a, b]$ for all x, $g'(x)$ defined on (a, b) where $|g'(x)| \leq k < 1$. Then $p_n = g(p_{n-1})$ converges to p

Proof: since $p_n \in [a,b]$ and $|g'(x)| \le k < 1$, by MVT we have

$$|p_n p| = |g(p_{n-1} - g(p)| = |g'(x_0)||p_{n-1} - p| \le k|p_{n-1} - p|$$

This leads to

$$|p_n - p| \le k|p_{n-1} - p| \le k^2|p_{n-2} - p| \le \ldots \le k^n|p_0 - p|$$

Thus $\lim_{n\to\infty} |p_n - p| = 0$

2.3 Newton's Method and its Extentions

Let $f \in C^2[a,b]$, $p_0 \in [a,b]$, $f'(p_0) \neq 0$, and $|p - p_0|$ be small. Then we can use Taylor Polynomials:

$$f(p) = f(p_0) + (p - p_0)f'(p_0) + \frac{(p - p_0)^2}{2} f''(p^*)$$

where p^* is between p and p_0

Since p is a root, we have that $0 \approx f(p_0 + (p - p_0)f'(p_0) \implies p \approx p_0 - \frac{f(p_0)}{f'(p_0)} \equiv 1$

This then gives the recurrence: $p_n = p_{n-1} - \frac{f(p_{n-1})}{f'(p_{n-1})}$

2.3.1 Algorithm

```
INPUT: p_0, TOL, N_0
OUTPUT: p or FAILURE

i = 1;
while i ≤ N_0:
  p = p_0 − f(p_0)/f'(p_0);
  if |p − P_0| < TOL:
    Output p;
  i+ = 1;
  p_0 = p;
Output FAILURE;
```

Issues:

- Main issue of Newton's method is that we have to know the value of $f'(x)$. Solved using **Secant Method**

2.3.2 Second Method

Uses $f(p_{n-1}) \approx \frac{f(p_{n-1}) - f(p_{n-2})}{p_{n-1} - p_{n-2}}$ to create the recurrence formula:

$$p_n = p_{n-1} - \frac{f(p_{n-1})(p_{n-1} - p_{n-2})}{f(p_{n-1} - f(p_{n-1})}$$

```
INPUT: initial p_0, p_1; TOL, N_0
OUTPUT: solution p or FAILURE

i = 2;
q_0 = f(p_0);
q_1 = f(p_1);
while i ≤ N_0:
  p = p_1 − q_1(p_1 − p_0)/(q_1 − q_0);
  if |p − p_1| < TOL:
```

```
        OUTPUT p
    i+ = 1;
    p_0 = p_1;
    q_0 = q_1;
    p_1 = p;
    q_1 = f(p);
Output FAILURE;
```

2.3.3 Method of False Position

Problem of Secant Method is that the intermediary values we use are sometimes outside the desired bracket. False Position ensures that intermediary values are inside the desired bracket by testing if $f(p_0)$ and $f(p_1)$ have opposite signs