Lecture Notes

Kyle Chui

2022-01-04

Contents

1 Lecture 1

The goal of this class is to solve *mathematical* problems with the help of *computers*.

1.1 Chapter Summaries

1. Computations in computers
   - How to store (real) numbers in the computer

     > **Note.** If the number has finitely many digits, then it is simple. What about numbers with infinitely many digits, i.e. $\frac{1}{3}$? We have to truncate or round, and store an approximation with finitely many digits.

   - How to perform computations

     > **Note.** From regular math, we know that $\frac{1}{3} + \frac{1}{3} = \frac{2}{3}$. However, in the computer, due to errors, we have $\frac{1}{3} \oplus \frac{1}{3} = ? \oplus ? = ??$.

   - Errors
2. - Find roots of $f(x) = 0$ using bisection, Newton's method, ...
   - Convergence
   - Convergence order (how fast it converges)
3. Polynomial interpolation
   - Approximate a function $f(x)$ by a polynomial $P(x)$, where $f(x_i) = P(x_i)$ for finitely many x_i
   - *Accuracy* of the polynomial approximations
4. Numerical differentiations and numerical integrations
   - Using the approximations from chapter 3, we can approximate using

     $$f'(x^*) \approx P'(x^*) = \sum_{i=0}^{k} f(x_k)c_k,$$

     and

     $$\int_a^b f(x)\,\mathrm{d}x \approx \int_a^b P(x)\,\mathrm{d}x = \sum_{i=1}^{k} f(\overline{x_k})\overline{c_k}.$$

   - Error analysis

6.7. Solving linear systems of equations
   - Direct methods: Gaussian elimination (computationally expensive)
   - Iterative methods: (faster and cheaper)
   - Solution stability

1.2 Round-off errors and computer arithmetics

There are three kinds of errors:

- Modeling Error: Occurs when we convert a problem from the real world into the mathematical world.
- Method Error: Occurs when we try to solve the mathematical problem numerically.
- Round-off error: Occurs when the computer gives an incorrect result with the correct algorithm (comes from storage and computation).

1.2.1 Storage

Infinite digit real numbers are *stored* as finite digit numbers, using the normalized decimal form of real numbers.

Definition. *Normalized decimal form of a real number*
For any $y \in \mathbb{R}$, we may write

$$y = \pm 0.d_1 d_2 d_3 \dots d_k d_{k+1} \dots \cdot 10^n,$$

where $0 < d_1 \leq 9$, $0 \leq d_i \leq 9$, n are integers. For the particular case where $y = 0$, we write $y = 0.0 \cdot 10^0$.

Definition. *Normalized machine numbers (Floating-point form)*
Any machine number y can be written as

$$y = \pm 0.d_1 d_2 \cdots d_k \cdot 10^n,$$

where $0 < d_1 \leq 9$, $0 \leq d_i \leq 9$, n are integers.

We can think of the storage process as mapping normalized real numbers to normalized machine numbers. We do this via rounding or truncating.

Consider some $y \in \mathbb{R} \setminus \{0\}$.

- Truncating (k-digit truncation of $y = \pm 0.d_1 d_2 d_3 \dots d_k d_{k+1} d_{k+2} \dots \cdot 10^n$) Simply omit the digits from d_{k+1} and onwards, in other words

  $$\text{fl}(\pm 0.d_1 d_2 d_3 \dots d_k d_{k+1} d_{k+2} \dots \cdot 10^n) = \pm 0.d_1 d_2 \dots d_k \cdot 10^n.$$

  Thus we have $y \approx \text{fl}(y)$.

- Rounding (k-digit rounding of $y = \pm 0.d_1 d_2 d_3 \dots d_k d_{k+1} d_{k+2} \dots \cdot 10^n$)

  If $d_{k+1} < 5$, then we drop $d_{k+1} d_{k+2} \dots$ (same with truncating)

  If $d_{k+1} \geq 5$, then add 1 to d_k and drop $d_{k+1} d_{k+2} \dots$

  $$\text{fl}(\pm 0.d_1 d_2 d_3 \dots d_k d_{k+1} d_{k+2} \dots \cdot 10^n) = \pm \delta_1 \delta_2 \dots \delta_k \cdot 10^m.$$

2 Lecture 2

Note. Since we use the notation fl to denote both k-digit truncation as well as k-digit rounding, be sure not to mix the two up.

Definition. *Errors*
Suppose p^* is an approximation of p. Then the *actual error* is $p - p^*$, the *absolute error* is $|p - p^*|$, and the relative error is $\frac{|p-p^*|}{|p|}$, where $p \neq 0$.

Definition. *Significant Digits*
The number p^* is said to approximate p to "t" significant digits if "t" is the largest non-negative integer for which

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

2.1 Computer Arithmetic

Assume that x, y are real numbers, then

$$\begin{aligned} x \oplus y &= \text{fl}(\text{fl}(x) + \text{fl}(y)) \\ x \ominus y &= \text{fl}(\text{fl}(x) - \text{fl}(y)) \\ x \otimes y &= \text{fl}(\text{fl}(x) \cdot \text{fl}(y)) \\ x \oslash y &= \text{fl}(\text{fl}(x) + \text{fl}(y)) \end{aligned}$$

3 Lecture 3

Note. When computing relative error, keep 2 non-zero digits.

If after performing our operation with k-digit chopping, we still have k significant digits, then our operation is pretty good (because it did not lose precision). In general, the $\oplus$ operator will lose at most one significant digit. Unlike addition, the $\ominus$ operator can lose multiple significant digits (when you subtract almost identical numbers).

We see errors introduced in computations:

- Subtraction of almost identical numbers results in loss of significant digits
- More operations $\to$ more errors $\to$ loss of significant digits

To avoid loss of *accuracy*, we can perform some manipulations to avoid scenarios that involve inaccuracies:

Example. *Quadratic formula*
For a given quadratic $ax^2 + bx + c = 0$, we know that a solution is

$$x_1 = \frac{-b + \sqrt{b^2 - 4ac}}{2a}.$$

Suppose $b > 0$ and $b^2 \gg 4ac$. Then $b^2 - 4ac \approx b^2$, so $b \approx \sqrt{b^2 - 4ac}$. This can lead to inaccuracies because we are subtracting nearly identical numbers. To avoid this, we can perform some algebra as follows:

$$\begin{aligned} x_1 &= \frac{-b + \sqrt{b^2 - 4ac}}{2a} \\ &= \frac{-b + \sqrt{b^2 - 4ac}}{2a} \cdot \frac{-b - \sqrt{b^2 - 4ac}}{-b - \sqrt{b^2 - 4ac}} \\ &= \frac{b^2 - (b^2 - 4ac)}{2a(-b - \sqrt{b^2 - 4ac})} \\ &= \frac{4ac}{2a(-b - \sqrt{b^2 - 4ac})} \\ &= \frac{2c}{-b - \sqrt{b^2 - 4ac}}. \end{aligned}$$

Since this solution avoids the subtraction of two nearly identical numbers, it is more accurate. For the other root, we are subtracting two negative numbers that are nearly identical, so there is no issue.

4 Lecture 4

4.1 Nested Algorithm

Goal. We want to reduce our error by reducing the number of operations done.

Example. Consider the polynomial

$$p(x) = 4x^4 + 5x^2 + 2x^2 + 5x - 10.$$

We need to perform 4 multiplications for the first term, 3 for the second, etc., for a total of 10 multiplication operations and 4 addition operations. Notice that we can factor out a common x from the first few terms, to get

$$p(x) = x \cdot (4x^3 + 5x^1 + 2x + 5) - 10.$$

Continuing this pattern we get

$$p(x) = x(x(x(4x + 5) + 2) + 5) - 10.$$

We now only have 4 multiplications and 4 additions, which can improve the accuracy of our algorithm.

4.2 Convergence Order

Definition. *Convergence Order*
Assume $\{P_n\}_{n=0}^{\infty}$ converges to P with $P_n \neq P$. If there exists $0 < \lambda < \infty$ and $\alpha > 0$ such that

$$\lim_{n\to\infty} \frac{|P_{n+1} - P|}{|P_n - P|^{\alpha}} = \lambda.$$

Then we say $\{P_n\}_{n=0}^{\infty}$ converges to P with order α.

Note. Larger α implies a faster convergence.

Definition. *Linear and Quadratic Convergence*
If $\alpha = 1$, then $\{P_n\}_{n=0}^{\infty}$ converges to P *linearly*. If $\alpha = 2$, then $\{P_n\}_{n=0}^{\infty}$ converges to P *quadratically*.

5 Lecture 5

5.1 Root Finding (Single Variable)

Given a function $f(x) = 0$ with a root $p \in [a, b]$, we want to find a sequence $p_1, p_2, \ldots$ such that $p_n \to p$. We want to find the root using the Intermediate Value Theorem. Suppose f is a continuous function on $[a, b]$ with $f(a) \cdot f(b) < 0$. Then f has at least one root in (a, b).

Note. This statement holds true because it means that $f(a)$ and $f(b)$ have opposite sign, or that 0 is between $f(a)$ and $f(b)$. Hence there must exist some $p \in (a, b)$ such that $f(p) = 0$.

5.1.1 Bisection Method

Idea. Use binary search to find the root.

You first find the midpoint of your interval (let's call this m), and check whether the sign of $f(m)$ is the same as $f(a)$ or $f(b)$. If the former, then we may "shrink" the interval to $[m, b]$, otherwise we may "shrink" the interval to $[a, m]$. If $f(m) = 0$, then we have found our root. If we perform this algorithm recursively, then we may find a sequence of points that converges to our root.

Note.

- The bisection method *always* converges to a root.
- We stop the iteration if:
  - We reach the maximum iteration number.
  - For some small $\varepsilon > 0$,

$$|p_n - p_{n-1}| < \varepsilon \text{ or } \frac{|p_n - p_{n-1}|}{|p_n|} < \varepsilon \text{ or } |f(p_n)| < \varepsilon.$$

Theorem. Suppose $f \in C[a, b]$ and $f(a) - f(b) < 0$. The Bisection method generates a sequence $\{p_n\}_{n=1}^{\infty}$ approximating a zero p of $f(x)$ with

$$|p_n - p| \leq \frac{b - a}{2^n}, \text{ where } n \geq 1.$$

Proof. For $n \geq 1$, we have $p_n = \frac{a_n + b_n}{2^n}$, $p \in (a_n, b_n)$. Then we know that

$$\begin{aligned} |p_n - p| &\leq \frac{1}{2}(b_n - a_n) \\ &= \frac{1}{2}\left(\frac{1}{2}(b_{n-1} - a_{n-1})\right) \\ &= \cdots \\ &= \underbrace{\frac{1}{2} \cdot \cdots \cdot \frac{1}{2}}_{n \text{ times}}(b_1 - a_1) \\ &= \frac{1}{2^n}(b - a). \end{aligned}$$

□

6 Lecture 6

6.1 Fixed Point Method (Finding Roots)

Definition. *Fixed Point*
We say p is a *fixed point* of $g(x)$ if $g(p) = p$.

- We first need to translate our root-finding problem into a fixed point problem.
- We want to solve $f(x) = 0$, so we should find p such that $f(p) = 0$.
- We rewrite $f(x) = 0$ as $g(x) \coloneqq f(x) + x = x$.
- Consider the relation between f and g:
  - Assume p is a root of $f(x)$, so $g(p) = f(p) + p = 0 + p = p$.
  - Thus if p is a root of f, then it must be a fixed point of g.

Note. The fixed point function $g(x)$ is *not* unique! In general, we have that $g(x) \coloneqq cf(x) + x$ is a fixed point function of f, given that $c \neq 0$.

How can we find the fixed point p of $g(x)$?

- Choose an initial approximation p_0 (of p).
- We have $p_1 = g(p_0), p_2 = g(p_1), \ldots$, so compute $p_n = g(p_{n-1})$ for $n \geq 1$.
- If $|p_n - p_{n-1}| < \varepsilon$ or $\frac{|p_n - p_{n-1}|}{|p_n|} < \varepsilon$, then stop.

Note. Some fixed point functions can diverge! You should choose a fixed point that converges.

Theorem — *Fixed Point Theorem*

(1) If $g \in C[a, b]$ and $g(x) \in [a, b]$ for all $x \in [a, b]$, then $g(x)$ has at least one fixed point in $[a, b]$.

(2) If, in addition, $g'(x)$ exists on (a, b) and there exists a positive constant $k < 1$ such that

$$|g'(x)| \leq k, \quad \text{for all } x \in (a, b),$$

then there is exactly one fixed point in $[a, b]$.

(1) *Proof.* If $g(a) = a$ or $g(b) = b$ then we are done. Hence we may assume that $g(a) \neq a, g(b) \neq b$. Let us define a function $h(x) \coloneqq g(x) - x$, which is continuous. Then we know that since $g(a) > a$, then $h(a) > 0$. Similarly, as $g(b) < b$, we have $h(b) < 0$. Hence by Intermediate Value Theorem there exists a point $c \in (a, b)$ such that $h(c) = 0$. In other words, there exists a point such that $g(c) - c = 0$, or $g(c) = c$. □

(2) *Proof.* We have that $|g'(x)| \leq k < 1$. Suppose towards a contradiction that there are two fixed points p, q of $g(x)$ on $[a, b]$ where $p \neq q$. Thus we have

$$\begin{aligned} |p - q| &= |g(p) - g(q)| \\ &= |p - q| \cdot \frac{g(p) - g(q)}{|p - q|} \\ &= |p - q| \cdot g'(c) && \text{(MVT, } c \in (a, b)\text{)} \end{aligned}$$

Dividing both sides by $|p - q|$, we have $g'(c) = 1$, a contradiction. Hence $g(x)$ has a *unique* fixed point on $[a, b]$. □

6.1.1 Convergence of Fixed Point Iteration

A fixed point of $g(x)$ is the point where $g(x)$ intersects the line $y = x$.

7 Lecture 7

If we consider the error generated by the fixed point iteration method, $e_n = |p - p_n|$, we have that this is equal to $|g(p) - g(p_{n-1})|$, where $g \in C[p, p_{n-1}]$. By Mean Value Theorem we know that there exists some $\xi \in (p, p_{n-1})$ such that

$$\begin{aligned} |g(p) - g(p_{n-1})| &= |g'(\xi)|\,|p - p_{n-1}| \\ &= |g'(\xi)|\,e_{n-1}. \end{aligned}$$

Hence if $e_n < e_{n-1}$ then $|g'(\xi)| < 1$ and the fixed point iteration converges to p. On the other hand, if $e_n \geq e_{n-1}$, then $|g'(\xi)| \geq 1$ and the fixed point iteration diverges.

Theorem — *Convergence of the Fixed Point Iteration*
Let $g(x) \in C[a, b]$ such that $g(x) \in [a, b]$ for all $x \in [a, b]$. Furthermore, suppose that $g'(x)$ exists on (a, b) and there exists $0 < k < 1$ such that

$$|g'(x)| \leq k \quad \text{for all } x \in (a, b).$$

Then for any $p_0 \in [a, b]$, the sequence defined by $p_n = g(p_n - 1)$ for all $n \geq 1$ converges to the unique fixed-point p in $[a, b]$.

Proof. Let $p_0 \in [a, b]$. Then

$$\begin{aligned} |p - p_n| &= |g(p) - g(p_{n-1})| \\ &= |g'(\xi)(p - p_{n-1})| && (\xi \in (p, p_{n-1}) \subset [a, b]) \\ &= |g'(\xi)|\,|p - p_{n-1}| \\ &\leq k\,|p - p_{n-1}| \\ &\leq k^2\,|p - p_{n-2}| \\ &\leq \cdots \\ &\leq k^n\,|p - p_0|\,. \end{aligned}$$

Hence

$$\begin{aligned} \lim_{n\to\infty} |p - p_n| &\leq \lim_{n\to\infty} k^n\,|p - p_0| \\ &= |p - p_0| \lim_{n\to\infty} k^n \\ &= 0. \end{aligned}$$

Thus $\lim_{n\to\infty} |p - p_n| = 0$, so $p_n \to p$ for any $p_0 \in [a, b]$. □

Error Analysis. We want to bound $|p - p_n|$ by some known values. Since $p_0 \in [a, b]$ and $|p - p_n| \leq k^n\,|p - p_0|$, we have

$$|p - p_n| \leq k^n \max\{|a - p_0|\,, |b - p_0|\}\,.$$

We can then use this equation to find the number of iterations required to get to a certain error bound. If we want to have some error bound of ε, then

$$\begin{aligned} k^n \max\{|a - p_0|\,, |b - p_0|\} &\leq \varepsilon \\ k^n &\leq \frac{\varepsilon}{\max\{|a - p_0|\,, |b - p_0|\}} \\ n \log k &\leq \log \frac{\varepsilon}{\max\{|a - p_0|\,, |b - p_0|\}} \\ n &\geq \frac{1}{\log k} \cdot \log \frac{\varepsilon}{\max\{|a - p_0|\,, |b - p_0|\}}. && (\log k < 0) \end{aligned}$$

8 Lecture 8

8.1 Newton's Method

- Choose an initial approximation p_0 of p.
- Set p_n to be the x-intercept of the tangent line passing through $(p_{n-1}, f(p_{n-1}))$. If we wish to solve this equation we have:

$$y - f(p_{n-1}) = f'(p_{n-1}) \cdot (x - p_{n-1})$$
$$x = p_{n-1} - \frac{f(p_{n-1})}{f'(p_{n-1})}.$$

  We can only use this for $f'(p_{n-1}) \neq 0$, since if it were then the tangent line would have no x-intercept. We can treat this method as a form of a fixed point method, with $g(x) = x - \frac{f(x)}{f'(x)}$.

Note. Newton's method converges faster than the regular fixed point method when $p_0 \approx p$.

Theorem — *Convergence of Newton's Method*
Let $f(x) \in C^2[a, b]$. If $p \in (a, b)$ such that $f(p) = 0$ and $f'(p) \neq 0$, then there exists a $\delta > 0$ such that the Newton's method generates a sequence $\{p_n\}_{n=1}^{\infty}$ which converges to p for any initial approximation $p_0 \in [p - \delta, p + \delta]$.

Proof. Consider Newton's method to be a fixed-point method with

$$g(x) = x - \frac{f(x)}{f'(x)}.$$

We wish to show:

(i) $g(x)$ exists on $(p - \delta, p + \delta)$.

(ii) $g(x)$ exists on $(p - \delta, p + \delta)$ and there exists $0 < k < 1$ such that $|g'(x) \leq k|$.

(iii) $g(x) \in [p - \delta, p + \delta]$ for $x \in [p - \delta, p + \delta]$.

Since $f(x) \in C^2[a, b]$, we know that f, f', and f'' are continuous on $[a, b]$. Since $f'(p) \neq 0$ and f' continuous, there exists $\delta_1 > 0$ such that $f'(x) \neq 0$ for all $x \in [p - \delta_1, p + \delta_1]$. Hence $g(x)$ is continuous on $[p - \delta_1, p + \delta_1]$. Observe that

$$g'(x) = 1 - \frac{(f'(x))^2 - f(x)f''(x)}{(f'(x))^2} = \frac{f(x) \cdot f''(x)}{(f'(x))^2}.$$

Since f, f', and f'' are continuous on $[p - \delta_1, p + \delta_1]$ and $f'(x) \neq 0$ on $[p - \delta_1, p + \delta_1]$, we have that $g'(x)$ is continuous on $[p - \delta_1, p + \delta_1]$. Hence

$$g'(p) = \frac{f(p)f''(p)}{(f'(p))^2} = 0,$$

so there must be some small interval $0 < \delta < \delta_1$ such that $|g'(x)| \leq k$ where $0 < k < 1$.

We must finally show that $g(x) \in [p - \delta, p + \delta]$ when $x \in [p - \delta, p + \delta]$. Let $x \in [p - \delta, p + \delta]$, so

$$\begin{aligned} |g(x) - p| &= |g(x) - g(p)| \\ &= |g'(\xi) \cdot (x - p)| && \text{(MVT)} \\ &= |g'(\xi)|\,|x - p| \\ &< |x - p| \\ &\leq \delta. \end{aligned}$$

Thus $g(x) \in (p - \delta, p + \delta)$.
Therefore there exists some $\delta > 0$ such that the above three statements hold, and so by the convergence of a fixed point iteration we have that any $p_0 \in [p - \delta, p + \delta]$ generates a sequence that converges to p. □

Note. This theorem only tells us the *existence* of such a δ, and doesn't tell us anything about δ. In practice, we usually run a few bisection iterations to get closer to the actual root (since it will converge towards the root), and then switch over to Newton's method.

9 Lecture 9

9.1 Secant Method (extension of Newton's Method)

From last lecture, we have

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

where p_0 is given and $n \geq 1$. However:

- $f'(x)$ might not be available
- $f'(x)$ might be expensive to compute
- $f'(p_{n-1}) \neq 0$

Instead, we can try approximating the tangent line using secant lines, giving us

$$f'(p_{n-1}) = \lim_{x \to p_{n-1}} \frac{f(x) - f(p_{n-1})}{x - p_{n-1}}$$
$$f'(p_{n-1}) \approx \frac{f(p_{n-2}) - f(p_{n-1})}{p_{n-2} - p_{n-1}}.$$

Hence for the Secant Method, we replace $f'(p_{n-1})$ with $\dfrac{f(p_{n-2}) - f(p_{n-1})}{p_{n-2} - p_{n-1}}$, yielding

$$\begin{aligned} p_n &= p_{n-1} - \frac{f(p_{n-1})}{f'(p_{n-1})} \\ &= p_{n-1} - f(p_{n-1}) \cdot \frac{p_{n-2} - p_{n-1}}{f(p_{n-2}) - f(p_{n-1})} \\ &= p_{n-1} - f(p_{n-1}) \cdot \frac{p_{n-1} - p_{n-2}}{f(p_{n-1}) - f(p_{n-2})}. \end{aligned}$$

Note. In order to run this algorithm, we have to have *two* previous iterations, i.e. we need both p_{n-1} and p_{n-2} to run the iteration, instead of just p_{n-1} like we need for Newton's method.

9.2 Convergence of Some Iterative Methods

Theorem. Let $g \in C^1[a, b]$ such that $g(x) \in [a, b]$ for all $x \in [a, b]$. Then there exists $0 < k < 1$ such that $|g'(x)| < k$ for all $x \in (a, b)$. If $g'(p) \neq 0$, then for any number $p_0 \in [a, b]$ with $p_0 \neq p$, the sequence

$$p_n = g(p_{n-1}) \qquad (n \geq 1)$$

converges linearly to the unique fixed point in $[a, b]$.

Proof. We know that the unique fixed point p exists and the sequence $\{p_n\}$ converges, as we have shown before. Observe that if we Taylor expand at p, then we get

$$p_n = g(p_{n-1}) = g(p) + g'(\xi_{n-1})(p - p_{n-1}),$$

so

$$p_n - p = g'(\xi_{n-1})(p - p_{n-1}).$$

Hence we have

$$\begin{aligned} \lim_{n \to \infty} \frac{|p_n - p|}{|p_{n-1} - p|^1} &= \lim_{n \to \infty} \frac{|g'(\xi_{n-1})(p - p_{n-1})|}{|p_{n-1} - p|} \\ &= \lim_{n \to \infty} |g'(\xi_{n-1})|. \end{aligned}$$

Since $g \in C^1$, we know that g' is continuous on $[a, b]$. Thus

$$\begin{aligned} &= \left| g'\left(\lim_{n\to\infty} \xi_{n-1} \right) \right| \\ &= |g'(p)| \\ &> 0. \end{aligned}$$

Furthermore, since g' is continuous on $[a, b]$, by Extreme Value Theorem we know that it is bounded, so

$$\lim_{n\to\infty} \frac{|p_n - p|}{p_{n-1} - p} = \lambda,$$

where $0 < \lambda < \infty$. □

Theorem. Let $\alpha \in \mathbb{Z}$, and $g \in C^\alpha[a, b]$ such that $g(x) \in [a, b]$ for all $x \in [a, b]$. Then there exists $0 < k < 1$ such that $|g'(x)| < k$ for all $x \in (a, b)$. If $g'(p) \neq 0$, then for any number $p_0 \in [a, b]$ with $p_0 \neq p$, the sequence

$$p_n = g(p_{n-1}) \qquad (n \geq 1)$$

converges with order α to the unique fixed point in $[a, b]$.

Proof. Just like the last proof, we Taylor expand:

$$\begin{aligned} g(p_{n-1}) &= g(p) + g'(p)(p - p_{n-1}) + \cdots + \frac{g^{(\alpha-1)}(p)}{(\alpha-1)!}(p - p_{n-1})^{\alpha-1} + \frac{g^{(\alpha)}(\xi_{n-1})}{\alpha!}(p - p_{n-1})^\alpha \\ &= g(p) + \frac{g^{(\alpha)}(\xi_{n-1})}{\alpha!}(p - p_{n-1})^\alpha \\ &= p + \frac{g^{(\alpha)}(\xi_{n-1})}{\alpha!}(p - p_{n-1})^\alpha. \end{aligned}$$

Hence we have

$$\begin{aligned} \lim_{n\to\infty} \frac{|p_n - p|}{(p - p_{n-1})^\alpha} &= \lim_{n\to\infty} \frac{g^{(\alpha)}(\xi_{n-1})}{\alpha!} \\ &= \frac{g^{(\alpha)}(p)}{\alpha!}. \end{aligned}$$

□

Thus from the above two theorems we have the following:

- If $g'(p) \neq 0$, then it converges linearly.
- If $g'(p) = 0$ but $g''(p) \neq 0$, then it converges quadratically.
- If $g'(p) = 0$ and $g''(p) = 0$ but $g'''(p) \neq 0$, then it converges with third order.

If we apply this to Newton's method, we see that $f(p) = 0$ and $f'(p) \neq 0$. Furthermore, for $g(x) \coloneqq x - \frac{f(x)}{f'(x)}$, we have $g(p) = p$ and $g'(p) = 0$. Hence we conclude that Newton's method is *at least* quadratic.

10 Lecture 10

For Newton's Method to work, we assume that $f(p) = 0$ and $f'(p) \neq 0$. Let us now consider the case where $f'(p) = 0$. We have

$$g(p) = p - \frac{f(p)}{f'(p)},$$

which is undefined because $f'(p) = 0$. If we try to compute $g'(p)$, we get

$$g'(p) = \frac{f(p)f''(p)}{(f'(p))^2},$$

which is also undefined. Hence in this case Newton's Method converges *slower* than quadratically (specifically linearly).

10.1 Multiple Roots (Multiplicity)

Definition. *Multiplicity*
We say that p is a root of $f(x)$ with *multiplicity* m if $f(x) = (x-p)^m q(x)$ where $\lim_{x\to p} q(x) \neq 0$.

Example. Consider the function $f(x) = e^x - x - 1$. We see that it has a root at $p = 0$. We write

$$\begin{aligned} f(x) &= 1 + x + \frac{x^2}{2!} + \cdots - x - 1 \\ &= \frac{1}{2}x^2 + \frac{1}{6}x^3 + \cdots \\ &= (x-0)^2 \underbrace{\left(\frac{1}{2} + \frac{1}{6}x + \cdots\right)}_{q(x)}. \end{aligned}$$

Hence we have

$$\lim_{x\to 0} q(x) = \frac{1}{2} \neq 0.$$

Therefore $f(x)$ has a root at 0 with multiplicity 2.

In general, if the multiplicity of a root is larger than 1, then it converges slower. What this means for Newton's method is that:

- If $f(p) = 0$ and $f'(p) \neq 0$ then the root is a single root, and so the convergence is at least quadratic.
- If $f(p) = 0$ and $f'(p) = 0$ then the root has multiplicity at least two, and so the convergence is at best linear.

Theorem. The function $f(x) \in C^m[a,b]$ has a zero of multiplicity m at p if and only if $f(p) = f'(p) = \cdots = f^{(m-1)}(p) = 0$ and $f^{(m)}(p) \neq 0$.

Theorem. If p is a root of $f(x)$ with multiplicity m then p is a single root of

$$\mu(x) = \frac{f(x)}{f'(x)}.$$

Proof. If p is a root of $f(x)$ with multiplicity m then we may write

$$f(x) = (x-p)^m \cdot q(x),$$

where $\lim_{x\to p} q(x) \neq 0$. Hence

$$\begin{aligned}\mu(x) &= \frac{(x-p)^m \cdot q(x)}{(x-p)^m \cdot q'(x) + m(x-p)^{m-1}q(x)} \\ &= (x-p) \cdot \frac{q(x)}{(x-p)q'(x) + mq(x)}.\end{aligned}$$

Since $\lim_{x\to p} q(x) \neq 0$, we have

$$\lim_{x\to p} \frac{q(x)}{(x-p)q'(x) + mq(x)} = \frac{1}{m} \neq 0.$$

Thus by definition p is a single root of $\mu(x)$. □

10.2 Modified Newton's Method

If p is a root of $f(x)$ with multiplicity $m \geq 2$, then we apply Newton's Method to

$$\mu(x) \coloneqq \frac{f(x)}{f'(x)}.$$

The new method will converge at least quadratically.

Note. The drawback of the Modified Newton's Method is that we now require the *second* derivative of f, rather than just the first.

11 Lecture 11

11.1 Lagrange Interpolation Polynomial

The problem is that we are given $n+1$ points of the form (x_i, y_i) for $i = 0, 1, \ldots, n$ (x_i are distinct). We wish to find a polynomial $P_n(x)$ of degree at most "n" such that

$$P_n(x_i) = y_i$$

for all $i = 0, 1, \ldots, n$.

- If we have a y^* corresponding to some $x^* \neq x_i$, we can approximate it with $y^* \approx P_n(x^*)$.
- Given $f(x)$ and distinct x_i ($i = 0, 1, \ldots, n$). We can find a polynomial function $P_n(x)$ of degree at most "n" such that $P_n(x_i) = f(x_i)$.
- We are approximating our function f with a polynomial, and this extends to derivatives and integrals as well.
- General form of a polynomial $P_n(x)$ of degree at most "n":

  $$P_n(x) = a_n x^n + a_{n-1}x^{n-1} + \cdots + a_1 x + a_0.$$

  Hence it suffices to just find the coefficients such that the polynomial value equals the original function value.

If we try to solve for the coefficients by plugging in the $n+1$ points, we essentially solve the matrix:

$$\begin{bmatrix} 1 & x_0 & \cdots & x_0^2 \\ 1 & x_1 & \cdots & x_1^n \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_n & \cdots & x_n^n \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_n \end{bmatrix} = \begin{bmatrix} y_0 \\ y_1 \\ \vdots \\ y_n \end{bmatrix}.$$

Note. If the x_i are distinct, we know that the matrix is non-singular (i.e. there exists a unique solution for the a_i).

Drawbacks

- Solving the linear system might be computationally expensive.
- The matrix is "ill-conditioned", which means that it's "sensitive to inaccuracies" (i.e. storage or computation errors).

11.2 Lagrange Polynomials

We are given (x_i, y_i), where the x_i are distinct.

- Construct base functions (cardinal functions)

  Given $x_0, x_1, \ldots, x_n$ (distinct), construct $\ell_0(x), \ell_1(x), \ldots, \ell_n(x)$ of degree "n" polynomials such that

  $$\ell_i(x_j) = \delta_{ij} = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{otherwise} \end{cases}.$$

  Then we have

  $$\ell_i(x) = \frac{(x-x_0)(x-x_1)\cdots(x-x_{i-1})(x-x_{i+1})\cdots(x-x_n)}{(x_i-x_0)(x_i-x_1)\cdots(x_i-x_{i-1})(x_i-x_{i+1})\cdots(x_i-x_n)}.$$

  In other words,

  $$\ell_i(x) = \prod_{\substack{j=0 \\ j\neq i}}^{n} \frac{x-x_j}{x_i-x_j}.$$

We can then construct $\mathbb{P}_n(x)$ via

$$P_n(x) = \sum_{i=0}^{n} \ell_i(x) \cdot y_i.$$

12 Lecture 12

Note. Using Lagrange interpolation yields the same result as solving the system of linear equations, but without the need to solve a matrix equation.

Theorem. Suppose $x_0, x_1, \ldots, x_n$ are distinct numbers in the interval $[a, b]$ and $f \in C^{n+1}[a, b]$. Then for each $x \in [a, b]$, there exists $\xi_x \in (a, b)$ such that

$$f(x) = P_n(x) + \frac{f^{(n+1)}(\xi_x)}{(n+1)!}(x - x_0) \cdots (x - x_n)$$

where $P_n(x)$ is the interpolation polynomial.

Proof. Let $x \in [a, b]$. Then we have two cases: either $x = x_i$ for some $i = 0, 1, \ldots, n$, or $x \neq x_i$ for any $i = 0, 1, \ldots, n$. In the former case we have $(x - x_0) \cdots (x - x_n) = 0$ so for any $\xi_x \in (a, b)$, we have

$$\frac{f^{(n+1)}(\xi_x)}{(n+1)!}(x - x_0) \cdots (x - x_n) = 0.$$

Furthermore, since $f(x_i) = P(x_i)$ for all $i = 0, 1, \ldots, n$, the statement holds true for any $\xi_x \in (a, b)$. In the latter case, we define a new function $g(t) \colon [a, b] \to \mathbb{R}$ by

$$g(t) \coloneqq f(t) - P_n(t) - (f(x) - P_n(x)) \cdot \prod_{i=0}^{n} \frac{t - x_i}{x - x_i}.$$

Since $f(t), P(x) \in C^{n+1}[a, b]$, we have that $g(t) \in C^{n+1}[a, b]$. Observe that $g(x_i) = 0$ for all $i = 0, 1, \ldots, n$, so g has at least $n + 1$ distinct roots. Furthermore, we also have $g(x) = 0$, so g has a total of $n + 2$ distinct roots. Hence we may apply generalized Rolle's Theorem to get that there exists some $\xi_x \in (a, b)$ such that $g^{(n+1)}(\xi_x) = 0$. Therefore

$$g^{(n+1)}(\xi_x) = f^{(n+1)}(\xi_x) - 0 - (f(x) - P_n(x)) \cdot (n + 1)! = 0$$
$$f(x) = P_n(x) + \frac{f^{(n+1)}(\xi_x)}{(n+1)!}(x - x_0) \cdots (x - x_n).$$

□

13 Lecture 13

Since we have that $f(x) = P_n(x) + R(x)$, we have that our error term is given by:

$$\begin{aligned}\text{error} &= |f(x) - P_n(x)| \\ &= |R(x)| \\ &= \left| \frac{f^{(n+1)}(\xi_x)}{(n+1)!}(x - x_0) \cdots (x - x_n) \right|,\end{aligned}$$

for $x \in [a, b]$, $\xi_x \in (a, b)$. However, we can't explicitly find ξ_x, so we can't find the error. However, we can find an upper bound for the error by just finding the largest value of $f^{(n+1)}(x)$ over the interval (a, b). Hence our equation becomes

$$\text{error} \leq \frac{M}{(n+1)!} \prod_{i=0}^{n} |x - x_i|,$$

for $x \in [a, b]$ and $M = \max_{a<\xi_x<b} \left| f^{(n+1)}(\xi_x) \right|$.

Question. It is a bit wasteful to always have to recompute the interpolation polynomial whenever we add a new point. Can we somehow use $P_n(x)$ to generate $P_{n+1}(x)$?

13.1 Neville's Method

Let $P_{0,1,\ldots,n}(x)$ be the Lagrange polynomial interpolating $(x_0, f(x_0)), \ldots, (x_n, f(x_n))$. We need to recursively generate the interpolation polynomial. Suppose we have the $k + 1$ points $x_0, \ldots, x_k$. Let $0 \leq i < j \leq k$. Then a polynomial interpolating these $k + 1$ points can be given by

$$P_{0,\ldots,k}(x) = \frac{(x - x_i)P_{0,\ldots,i-1,i+1,\ldots,k}(x) - (x - x_j)P_{0,\ldots,j-1,j+1,\ldots,k}(x)}{x_j - x_i}.$$

14 Lecture 14

14.1 Newton's Divided Difference

As long as you are interpolating the same n points, you will always get the same polynomial if you use any of the methods studied before.

This method is similar to Neville's method in the sense that it is iterative, and the procedure is as follows:

- First find $P_0(x) = f(x_0)$.
- We interpolate $(x_1, f(x_1))$ by writing

$$P_1(x) = P_0(x) + a_1(x - x_0),$$

for some a_1 such that $P_1(x_1) = f(x_1)$.

- We interpolate $(x_2, f(x_2))$ by writing

$$P_2(x) = P_1(x) + a_2(x - x_0)(x - x_1),$$

for some a_2 such that $P_2(x_2) = f(x_2)$.

- Hence we have

$$\begin{aligned} P_n(x) &= P_{n-1}(x) + a_n(x - x_0)\cdots(x - x_{n-1}) \\ &= f(x_0) + a_1(x - x_0) + \cdots + a_{n-1}(x - x_0)\cdots(x - x_{n-2}) + a_n(x - x_0)\cdots(x - x_{n-1}). \end{aligned}$$

It remains to find these coefficients a_i for $i \in 0, \ldots, n$.

15 Lecture 15

Definition. *Divided Difference*
The zeroth divided difference is written as

$$f[x_i] = f(x_i).$$

The first divided difference is written as

$$f[x_i, x_{i+1}] = \frac{f[x_{i+1}] - f[x_i]}{x_{i+1} - x_i}.$$

The second divided difference is written as

$$f[x_i, x_{i+1}, x_{i+2}] = \frac{f[x_{i+1,x_{i+2}}] - f[x_i, x_{i+1}]}{x_{i+2} - x_i}.$$

Hence the k^{th} divided difference is

$$f[x_i, \ldots, x_{i+k}] = \frac{f[x_{i+1}, \ldots, x_{i+k}] - f[x_i, \ldots, x_{i+k-1}]}{x_{i+k} - x_i}.$$

Note. The value of the divided differences is *independent* of the order of $x_0, \ldots, x_k$.

15.1 Newton's Divided Difference Form of the Interpolation Polynomial

When we take $n = 1$, we have $P_1(x) = f(x_0) + a_1(x - x_0)$. To find a_1 such that $P_1(x_1) = f(x_1)$, we have

$$\begin{aligned} f(x_0) + a_1(x_1 - x_0) &= f(x_1) \\ a_1 &= \frac{f(x_1) - f(x_0)}{x_1 - x_0} \\ a_1 &= f[x_0, x_1]. \end{aligned}$$

When we have $n = 2$, we have

$$P_2(x) = f(x_0) + a_1(x - x_0) + a_2(x - x_0)(x - x_1).$$

Hence we solve for a_2 such that $P_2(x_2) = f(x_2)$. If we continue these computations we find that

$$P_n(x) = f[x_0] + f[x_0, x_1](x - x_0) + \cdots + f[x_0, \ldots, x_n](x - x_0) \cdots (x - x_{n-1}).$$

Note. A benefit for using the Divided difference algorithm is that we can continually factor out $x - x_i$ terms, without expanding the polynomial.

Theorem — *Uniqueness of Interpolation Polynomials*
If $x_0, x_1, \ldots, x_n$ are $n+1$ distinct numbers and f is a function whose values are gives at these inputs, then a unique polynomial P of degree at most "n" exists with $P(x_i) = f(x_i)$ for all $i \in 0, \ldots, n$.

Proof. We first show existence, by defining

$$P(x) = \sum_{i=0}^{n} \ell_i(x) f(x_i),$$

where $\ell_i(x)$ is our base function from before. We now show uniqueness. Let $P(x)$ and $Q(x)$ be two interpolation polynomials such that $P(x_i) = Q(x_i) = f(x_i)$ for all $i = 0, \ldots, n$. We define a difference

function $D(x) = P(x) - Q(x)$, so $D(x_i) = 0$ for all $i = 0, \ldots, n$ and $D(x)$ has degree at most "n". Since it has more distinct roots than it has degree, it must be identically zero. Therefore $P(x) = Q(x)$ and our interpolation polynomial is unique. $\square$

16 Lecture 16

16.1 Numerical Differentiation and Integration

Given $f(x)$:

- Approximate $f'(x^*)$.
- Approximate $\int_a^b f(x)\,\mathrm{d}x$.

Given $f(x^*), f(x^* \pm h), f(x^* \pm 2h), \ldots$:

- Approximate $f'(x^*)$.
- Approximate $\int_a^b f(x)\,\mathrm{d}x$.

16.1.1 Numerical Differentiation

We can just use secant lines to approximate the slopes of a tangent line:

$$f'(x^*) \approx \frac{f(x^* \pm h) - f(x^*)}{h}.$$

We have the forward difference given by

$$\Delta f(x^*) = f(x^* + h) - f(x^*),$$

and the backwards difference given by

$$\nabla f(x^*) = f(x^*) - f(x^* - h).$$

Idea. We should take the average of these two approximations and see if the "errors cancel out". We have

$$f'(x^*) \approx \frac{f(x^* + h) - f(x^* - h)}{2h}.$$

We call our central difference $f(x^* + h) - f(x^* - h)$.

From polynomial interpolations we can get an approximation for f, and we use this to approximate f'. We find a polynomial that interpolates the points $(x^*, f(x^*)), (x^* + h, f(x^* + h))$ to be

$$f'(x^*) = \frac{f(x^* + h) - f(x^*)}{h} - \underbrace{\frac{h}{2} f''(\xi_*)}_{\text{Error}} \qquad (\xi_* \in (x^*, x^* + h))$$

> **Note.** If our function is in C^2, we have that $|f''(\xi_*)|$ has a maximum value, and hence the error is order $\mathcal{O}(h)$.

17 Lecture 17

17.1 Multiple Point Formulas for Numerical Differentiation

We have the $n+1$ points $(x_0, f(x_0)), \ldots, (x_n, f(x_n))$. We have that $f(x) = \text{Lagrange Interpolation} + \text{Error}$, so

$$f(x) = \sum_{k=0}^{n} f(x_k)\ell_k(x) + \frac{(x-x_0)\cdots(x-x_n)}{(n+1)!} f^{(n+1)}(\xi_x). \qquad \xi_x \in (x_0, x_n)$$

We then approximate $f'(x)$ by

$$f'(x) = \sum_{k=0}^{n} f(x_k)\ell_k'(x) + D_x\left(\frac{(x-x_0)\cdots(x-x_n)}{(n+1)!}\right) f^{(n+1)}(\xi_x) + D_x(f^{(n+1)}(\xi_x)) \cdot \frac{(x-x_0)\cdots(x-x_n)}{(n+1)!}.$$

If we only consider the derivative at our interpolation points, i.e. x_j, then we have

$$f'(x_j) = \sum_{k=0}^{n} f(x_k)\ell_k'(x_j) + \frac{(x_j-x_0)\cdots(x_j-x_{j-1})(x_j-x_{j+1})\cdots(x_j-x_n)}{(n+1)!} f^{(n+1)}(\xi_x).$$

We call the above the $n+1$ point formula.

> **Note.** More points yield more accurate approximations, but also increase the number of operations which leads to round-off error. Because of this, too many points can also be a bad thing, and two, three, and five point formulas are the most common.

17.2 Three Point Formula

17.2.1 Three Point Formula (Left)

Suppose we are given the three points $(x_0, f(x_0)), (x_1, f(x_1)), (x_2, f(x_2))$, where x_0, x_1, x_2 are equidistant points. Then we have

$$\ell_0(x) = \frac{(x-x_1)(x-x_2)}{(x_0-x_1)(x_0-x_2)},$$
$$\ell_1(x) = \frac{(x-x_0)(x-x_2)}{(x_1-x_0)(x_1-x_2)},$$
$$\ell_2(x) = \frac{(x-x_0)(x-x_1)}{(x_2-x_0)(x_2-x_1)}.$$

Hence their derivatives are

$$\ell_0'(x) = \frac{x-x_2+x-x_1}{(x_0-x_1)(x_0-x_2)},$$
$$\ell_1'(x) = \frac{x-x_2+x-x_0}{(x_1-x_0)(x_1-x_2)},$$
$$\ell_2'(x) = \frac{x-x_1+x-x_0}{(x_2-x_1)(x_2-x_0)}.$$

Therefore

$$f'(x_j) = \frac{2x_j-x_2-x_1}{2h^2} f(x_0) + \frac{2x_j-x_2-x_0}{-h^2} f(x_1) + \frac{2x_j-x_1-x_0}{2h^2} f(x_2) + \prod_{\substack{k=0 \\ k\neq j}}^{2} \frac{x_j-x_k}{(n+1)!} \cdot f^{(3)}(\xi_{x_j}).$$

If we take $x^* = x_0$, with $x_1 = x^* + h$ and $x_2 = x^* + 2h$, we can simplify to get

$$f'(x^*) = \frac{1}{2h}\left(-3f(x^*) + 4f(x^*+h) - f(x^*+2h)\right) - \frac{1}{3}h^2 f^{(3)}(\xi_x).$$

17.2.2 Three Point Formula (Central)

$$f'(x^*) = \frac{f(x^* + h) - f(x^* - h)}{2h} - \frac{h^2}{6} f^{(3)}(\xi_*).$$

17.2.3 Three Point Formula (Right)

$$f'(x^*) = \frac{1}{2h}\left(3f(x^*) - 4f(x^* - h) + f(x^* - 2h)\right) + \frac{1}{3}h^2 f^{(3)}(\xi_x).$$

17.3 Five Point Formula

17.3.1 Five Point Formula (Left)

$$f'(x) = \frac{1}{12h}\left(-25f(x^*) + 48f(x^* + h) - 36f(x^* + 2h) + 16f(x^* + 3h) - 3f(x^* + 4h)\right) + \frac{h^4}{5} f^{(5)}(\xi_*).$$

17.3.2 Five Point Formula (Central)

$$f'(x) = \frac{1}{12h}\left(f(x^* - 2h) - 8f(x^* - h) + 8f(x^* + h) - f(x^* + 2h)\right) + \frac{h^4}{30} f^{(5)}(\xi_*).$$

18 Lecture 18

If we Taylor expand $f(x^* + h)$ and $f(x^* - h)$ at x^*, we can solve for $f''(x^*)$, yielding

$$f''(x^*) = \frac{f(x^* + h) - 2f(x^*) + f(x^* - h)}{h^2} + \frac{h^2}{24}\left(f^{(4)}(\xi_h) + f^{(4)}(\xi_{-h})\right),$$

where $\xi_{-h} \in (x^* - h, x^*)$ and $\xi_h \in (x^*, x^* + h)$.

18.1 Richardson's Extrapolation

- From lower accuracy method to higher accuracy method.
- Need to have the formula for approximation, as well as explicit error terms.
  - For the first case, we suppose that $M = N_1(h) + k_1 h + k_2 h^2 + k_3 h^3 + \cdots$, so $N_1(h)$ is an $\mathcal{O}(h)$ approximation of M. We also have that $M = N_1(\frac{h}{2}) + k_1 \cdot \frac{h}{2} + \cdots$, so we can combine these two equations to find
  $$M = 2N_1\left(\frac{h}{2}\right) - N_1(h) - \frac{k_2 h^2}{2} - \frac{k_3 h^3}{4} - \cdots,$$
  which is an $\mathcal{O}(h^2)$ approximation of M.

  > **Note.** We can define $N_2(h) = 2N_1(\frac{h}{2}) - N_1(h)$.

In general, we have

$$N_k(h) = \frac{2^{k-1} N_{k-1}\left(\frac{h}{2}\right) - N_{k-1}(h)}{2^{k-1} - 1}.$$

19 Lecture 19

For the second case, if we have that

$$M = N_1(h) + k_1h^2 + k_2h^4 + k_3h^6 + \cdots,$$

then we can use the same method shown in the previous lecture to derive that

$$N_k(h) = \frac{4^{k-1}N_{k-1}(\frac{h}{2}) - N_{k-1}(h)}{4^{k-1} - 1}.$$

The error term is hence given by

$$M = N_k(h) + \mathcal{O}(h^{2k}).$$

Since the forward difference and backwards difference of $f'(x^*)$, as well as the central difference of $f''(x^*)$ are of the form in the first case, we may apply it to extrapolate. Similarly, the central difference for $f'(x^*)$ takes the form given in the second case.

19.1 Numerical Integration

Given a function $f(x)$ on $[a, b]$, we want to find an approximation for $\int_a^b f(x)\,\mathrm{d}x$.

Numerical Quadrature We know that

$$\int_a^b f(x)\,\mathrm{d}x = \sum_{i=0}^{n} a_i f(x_i) + \text{Error}.$$

- Case 1: $x_0, x_1, \ldots, x_n \in [a, b]$ are given, and we need to find a_i.

  We have the points $(x_0, f(x_0)), (x_1, f(x_1)), \ldots, (x_n, f(x_n))$. We then interpolate at the aforementioned points to get an approximation for $f(x)$, yielding $f(x) = P(x) + \text{Error}$. Hence

  $$\int_a^b f(x)\,\mathrm{d}x = \int_a^b P(x)\,\mathrm{d}x + \int_a^b \text{Error}\,\mathrm{d}x.$$

  Using our Lagrange interpolation we find that our coefficients take the form

  $$a_i = \int_a^b \ell_i(x)\,\mathrm{d}x.$$

- Case 2: Find $x_0, \ldots, x_n \in [a, b]$ and $a_0, \ldots, a_n \in \mathbb{R}$.

19.1.1 Trapezoidal Rule

Theorem — *Weighted Mean Value for Integrals*
If we have a continuous function $f \in C[a, b]$, and $\int_a^b g(x)\,\mathrm{d}x$ exists and $g(x)$ does not change sign on $[a, b]$, then there exists $c \in (a, b)$ such that

$$\int_a^b f(x)g(x)\,\mathrm{d}x = f(c)\int_a^b g(x)\,\mathrm{d}x.$$

We know that the error term for Trapezoidal rule is given by

$$\begin{aligned}
\text{Error}^* &= \int_{x_0}^{x_1} \frac{f''(\xi)}{2}(x-x_0)(x-x_1)\,\mathrm{d}x \\
&= \frac{f''(\xi_c)}{2}\int_{x_0}^{x_1}(x-x_0)(x-x_1)\,\mathrm{d}x && \xi_c \in (x_0, x_1) \\
&= \frac{f''(\xi_c)}{2}\int_0^h z(z-h)\,\mathrm{d}z \\
&= \frac{f''(\xi_c)}{2}\cdot -\frac{1}{6}h^3 \\
&= -\frac{1}{12}f''(\xi_c)h^3
\end{aligned}$$

Hence we may write

$$\int_a^b f(x)\,\mathrm{d}x = \frac{h}{2}(f(x_0)+f(x_1)) - \frac{h^3}{12}f''(\xi_c) \qquad \xi_c \in (x_0, x_1)$$

20 Lecture 20

Trapezoidal Rule Let $x_0 = a, x_1 = a + h = b$, with $h = b - a$. Then

$$\int_{x_0}^{x_1} f(x)\,\mathrm{d}x = \underbrace{\frac{h}{2}(f(x_0) + f(x_1))}_{T(f)} - \underbrace{\frac{f''(\xi)}{12}h^3}_{E(f)} \qquad \xi \in (x_0, x_1)$$

Simpson's Rule Let $x_0 = a, x_1 = a + h, x_2 = b$, with $h = \frac{b-a}{2}$. We approximate $f(x)$ by $P(x)$ such that $f(x_0) = P(x_0), f(x_1) = P(x_1), f(x_2) = P(x_2)$. We then have

$$\int_a^b f(x)\,\mathrm{d}x = a_0 f(x_0) + a_1 f(x_1) + a_2 f(x_2) + \text{Error},$$

where $a_0 = \frac{h}{3}, a_1 = \frac{4h}{3}, a_2 = \frac{h}{3}$. In other words, we may write

$$\int_{x_0}^{x_2} f(x)\,\mathrm{d}x = \underbrace{\frac{h}{3}(f(x_0) + 4f(x_1) + f(x_2))}_{S(f)} - \underbrace{\frac{h^5}{90}f^{(4)}(\xi)}_{E(f)}. \qquad \xi \in (x_0, x_2)$$

Note. Trapezoidal rule is exact for x^0, x^1, but not x^2. Simpson's rule is exact for x^0, x^1, x^2, x^3, but not x^4.

Definition. *Degree of Precision*
The degree of precision of a numerical quadrature is "k" if the formula is exact for $x^0, \ldots, x^k$ but not exact for x^{k+1}.

Note. If the quadrature rule has degree of precision "ℓ", then the error has the form $Kf^{(\ell+1)}(\xi)$, where $\xi \in (a, b)$.

21 Lecture 21

Note. We can increase our accuracy by increasing the number of quadrature points, but this leads to complicated quadrature rules.

21.1 Composed Quadrature Rules

Idea. We can cut up our interval into smaller sub-intervals, and use simpler quadrature rules to approximate the area.

21.1.1 Composed Trapezoidal Rule

We divide $[a,b]$ into "n" sub-intervals, so $a = x_0, b = x_n$. We can then sum the areas yielded by Trapezoidal rule over the intervals $[x_i, x_{i+1}]$, for $i \in [0,n)$. The length of each interval is $h = \frac{b-a}{n}$, and $x_j = a + jh$. Hence we have

$$\begin{aligned}
\int_a^b f(x)\,\mathrm{d}x &= \int_{x_0}^{x_1} f(x)\,\mathrm{d}x + \cdots + \int_{x_{n-1}}^{x_n} f(x)\,\mathrm{d}x \\
&= \sum_{j=1}^{n} \left(\frac{h}{2}(f(x_j) + f(x_{j-1})) - \frac{h^3}{12} f''(\xi_j) \right) \\
&= \frac{h}{2}(f(x_0) + 2f(x_1) + \cdots + 2f(x_{n-1}) + f(x_n)) - \frac{h^3}{12} \sum_{j=1}^{n} f''(\xi_j) \\
&= \frac{h}{2}(f(x_0) + 2f(x_1) + \cdots + 2f(x_{n-1}) + f(x_n)) - \frac{h^3}{12} \cdot n f''(\eta) \\
&= \underbrace{\frac{h}{2}(f(x_0) + 2f(x_1) + \cdots + 2f(x_{n-1}) + f(x_n))}_{C_T(f)} - \frac{h^2(b-a)}{12} f''(\eta).
\end{aligned}$$

21.1.2 Composed Simpson's Rule

Again, we divide $[a,b]$ into n sub-intervals, where n is an even integer. We have

$$\begin{aligned}
\int_a^b f(x)\,\mathrm{d}x &= \sum_{j=1}^{\frac{n}{2}} \int_{x_{2j-2}}^{x_{2j}} f(x)\,\mathrm{d}x \\
&= \sum_{j=1}^{\frac{n}{2}} \frac{h}{3}(f(x_{2j-2}) + f(4x_{2j-1}) + f(x_{2j})) - \frac{h^5}{90} f^{(4)}(\xi_j) \\
&= \cdots \\
&= \frac{h}{3}\left(f(x_0) + 2\sum_{j=1}^{\frac{n}{2}-1} f(x_{2j}) + 4\sum_{j=1}^{\frac{n}{2}} f(x_{2j-1}) + f(x_n) \right) - \frac{h^5}{90} \sum_{j=1}^{\frac{n}{2}} f^{(4)}(\xi_j) \\
&= \underbrace{\frac{h}{3}\left(f(x_0) + 2\sum_{j=1}^{\frac{n}{2}-1} f(x_{2j}) + 4\sum_{j=1}^{\frac{n}{2}} f(x_{2j-1}) + f(x_n) \right)}_{C_S(f)} - \frac{h^4(b-a)}{180} f^{(4)}(\eta).
\end{aligned}$$

21.1.3 Composed Midpoint Rule

Again, we need to divide $[a, b]$ into n sub-intervals, where n is an even integer. Hence

$$\begin{aligned}\int_a^b f(x)\,\mathrm{d}x &= \sum_{j=1}^{\frac{n}{2}} \int_{x_{2j-2}}^{x_{2j}} f(x)\,\mathrm{d}x \\ &= \sum_{j=1}^{\frac{n}{2}} 2hf(x_{2j-1}) + \sum_{j=1}^{\frac{n}{2}} \frac{h^3}{3} f''(\xi_j) \\ &= 2h \sum_{j=1}^{\frac{n}{2}} f(x_{2j-1}) + \frac{h^2(b-a)}{6} f''(\eta).\end{aligned}$$

22 Lecture 22

22.1 Gaussian Quadrature Rule

We compute a polynomial that has $2n-1$ degrees of precision using n points. To do this, we first consider using n quadrature points over the interval $[-1,1]$. To find the polynomial for this, we create the *Legendre polynomials* $P_0(x),\dots,P_n(x)$:

- $P_n(x)$ is a *monomial* of degree "n".
- $\int_{-1}^{1} P(x)P_n(x)\,\mathrm{d}x = 0$ for any polynomial $P(x)$ of degree less than "n".

Thus we have that

$$\begin{aligned} P_0(x) &= 1 \\ P_1(x) &= x \\ P_2(x) &= x^2 - \frac{1}{3}. \end{aligned}$$

Observe that in the above, we have that

$$\begin{aligned} \int_{-1}^{1} P_0(x)P_1(x)\,\mathrm{d}x &= 0 \\ \int_{-1}^{1} P_0(x)P_2(x)\,\mathrm{d}x &= 0 \\ \int_{-1}^{1} P_1(x)P_2(x)\,\mathrm{d}x &= 0 \end{aligned}$$

Another way to obtain these monomials is to solve Legendre's equation:

$$(1-x^2)y'' - 2xy' + n(n+1)y = 0.$$

Note. The roots of the Legendre polynomial $P_n(x)$ are the Gaussian quadrature points $x_1, x_2, \dots, x_n$.

If $x_1, x_2, \dots, x_n$ are roots of the Legendre polynomial $P_n(x)$, then

$$c_j = \int_{-1}^{1} \prod_{\substack{i=1 \\ i\neq j}}^{n} \frac{x - x_i}{x_j - x_i}\,\mathrm{d}x$$

23 Lecture 23

We now generalize the Gaussian quadrature from the interval $[-1, 1]$ to $[a, b]$. In other words, we want to solve for

$$\int_a^b f(x)\,\mathrm{d}x = \int_{-1}^1 g(t)\,\mathrm{d}t.$$

Observe that the linear map $t = \frac{x-\frac{b+a}{2}}{\frac{b-a}{2}} = \frac{2x-b-a}{b-a}$ suffices. Solving for x yields $x = \frac{b-a}{2}t + \frac{b+a}{2}$. Thus we may rewrite our integral as

$$\int_a^b f(x)\,\mathrm{d}x = \int_{-1}^1 \frac{b-a}{2} f\left(\frac{b-a}{2}t + \frac{b+a}{2}\right)\,\mathrm{d}t.$$

23.1 Direct Methods for Solving Systems of Linear Equations

Consider the following system:

$$\begin{aligned}
x_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n &= b_1 \\
x_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n &= b_2 \\
x_{31}x_1 + a_{32}x_2 + \cdots + a_{3n}x_n &= b_3 \\
&\vdots \\
x_{n1}x_1 + a_{n2}x_2 + \cdots + a_{nn}x_n &= b_n
\end{aligned}$$

We can express it as a matrix of the form

$$A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n_1} & a_{n_2} & \cdots & a_{nn} \end{bmatrix}$$

We have a few methods to yield the solutions:

- Direct methods: Give exact solutions in a finite number of steps
- Iterative methods: Start with an initial guess x^0 and produce approximations $x^1, \ldots, x^n$ which will converge to x^*

23.1.1 Linear Systems of Equations and Gaussian Elimination

The idea is we transform our equation $Ax = b$ into the equation $\tilde{A}x = \tilde{b}$, which has the same solutions but is easier to solve. We have a few *elementary row operations* that we can perform:

- $\lambda E_i \to E_i$ for $\lambda \neq 0$
- $E_i + \lambda E_j \to E_i$
- $E_j \leftrightarrow E_i$

Once we get our system of equations into an upper-triangular matrix, we can use backwards-substitution to solve for x_j for $j = 1, \ldots, n$.

24 Lecture 24

Downsides of Gaussian Elimination

- For it to work, we need $a_{jj} \neq 0$ at every step.
- If the condition number is large, then we may have large inaccuracies (instability).

24.1 Gaussian Elimination with Partial Pivoting

We always assume that $Ax = b$ has a unique solution, so A^{-1} exists and partial pivoting will give us the solution.

At the j^{th} step, if $|a_{jj}| < \max_{j+1 \leq \ell \leq n} |a_{\ell j}|$, then swap the row at which the maximum is found with the j^{th} row. This ensures that $a_{jj} \neq 0$. Adding partial pivoting also makes our solution *stable*.

24.2 Solving Linear Systems by Matrix Factorization

Our goal is still to solve $Ax = b$. We want to factor $A = LU$, where L is a lower triangular matrix and U is an upper triangular matrix. We then set $Ux = y$, so $Ly = b$. Thus we can solve for y via forwards substitution, and solve for x via backwards substitution.

25 Lecture 25

For Gaussian Elimination, the number of operations needed is on the order of $\mathcal{O}(\frac{n^3}{3})$. For LU factorization, the number of operations needed is also on the order of $\mathcal{O}(\frac{n^3}{3})$. However, if we have multiple equations $Ax_i = b_i$, it is much more efficient to first LU factorize and then solve, as the complexity becomes $\mathcal{O}(n^2)$ for solving each equation.

Theorem. If Gaussian elimination can be performed on a system $Ax = b$ without row interchanges, then A has an LU factorization.

What if we need row interchanges in our Gaussian elimination?

Then we can find a matrix P (permutation matrix) such that we can perform Gaussian elimination on PA without row exchanges. Furthermore, $Ax = b \iff PAx = Pb$.