ECE 204 *Numerical methods*

Sections 001, 002

MIDTERM EXAMINATION

Douglas Wilhelm Harder dwharder@uwaterloo.ca EIT 4018 x37023

2019-02-25T10:15:00P1H15M

Rooms: E7-4043, 4053, 4417, 4433, 4437

1. **You may rip off the last page as soon as you sit down.**
2. There are 39 marks.
3. No aides.
4. Turn off all electronic media and store them under your desk.
5. You may ask only one question during the examination: “May I go to the washroom?”
6. Asking **any** other question **will** result in a deduction of 5 marks from the exam grade.
7. If you think a question is ambiguous, write down your assumptions and continue.
8. **Do not leave during first half hour or after there are only 15 minutes left.**
9. Do not stand up until all exams have been picked up.
10. If a question only asks for an answer, you do not have to show your work to get full marks; however, if your answer is wrong and no rough work is presented to show your steps, no part marks will be awarded.
11. The questions are in the order of the course material.

1 [3] List the six *tools* that we will use in this course for approximating solutions numerically.

2 [2] Sum the following two double-precision floating-point numbers and write the result in the same format:

```
8ac7000000000000    8adf000000000000
```

3 [2] Multiply the following two floating-point numbers and write the result in the same format:

```
-501050    -512100
```

4 [4] Author a function that implements fixed-point iteration where the function keeps iterating until the difference between two successive approximations is less than `eps`, but not iterating beyond the given maximum number of iterations. If the maximum number of iterations is reached without the first requirement being met, throw a `std::runtime_error` exception. You do not have to check if $f(x)$ ever returns an infinity or a not-a-number.

```
double fixed_point( double f( double x ), double x0, double eps,
                    unsigned int max_iterations ) {
```

5 [4] Demonstrate, using 2nd-order Taylor series, that the error of this approximation of the derivative

$$y^{(1)}(t) \approx \frac{-y(t+2h)+4y(t+h)-3y(t)}{2h}$$

is O(h^2).

6 [3] Your accurate sensor is relaying information back via a voltage that is then converted to a digital signal using an analog-to-digital converter. The sensor sends back a reading each 30 ms. The last three readings are 15, 4, –1. If the current time of the last reading is 415970 ms since the system was turned on, what is a reasonable estimation as to when the reading was most recently zero.

First explain how you will go about finding that time (1 mark) and the find that time (2 marks).

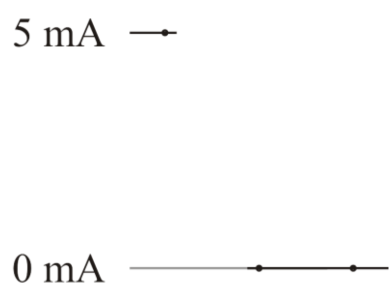
7 [3] Explain what tools we used and how we got from there to the formula for Simpson's rule,

$$\int_a^b f(x)\,\mathrm{d}x = \frac{1}{6}\left(f(a)+4f\left(\frac{a+b}{2}\right)+f(b)\right)(b-a),$$

and, and similarly, how we got to the approximation

$$\int_{\frac{a+b}{2}}^{b} y(t)\,\mathrm{d}t = \frac{1}{12}\left(-y(a)+8y\left(\frac{a+b}{2}\right)+5y(b)\right)\frac{b-a}{2}.$$

8 [4] Suppose you have three samples read by a sensor that samples one reading per second, but between the first and second readings, it is known that the approximately steady current is abruptly switched off. Initially the current is 5 mA, and for the two next readings, the current is 0 mA. Find an approximation of the charge passing a point during these two seconds using the composite trapezoid rule and Simpson's rule. What is the maximum possible error of the approximation given by the composite trapezoidal rule, and what is the maximum possible error for the approximation given by Simpson's rule?



9 [3] Given a very noisy power signal (watts) where the last five readings are 9 W (most recent), 8 W, 8 W, 9 W and 10 W (least recent) taken at 100 ms apart, and you know that the concavity is essentially zero on this interval, what is a reasonable approximation of the integral of the energy (joules) used during the last 100 ms? Find the approximation as a real number.

10 [1] Assuming that a function is continuous and differentiable, what is it about the bisection method and the constrained secant method that guarantee convergence to a root, while Newton's method and the second method may not converge to a root even if there is a root in the neighborhood of the initial approximations of the root.

11 [2] Apply one iteration of the constrained secant method to find a better approximation of the root of the polynomial $x^2 - 3$ on the interval [1, 2].

12 [2] Demonstrate that if the absolute error for an approximation x_0 of a root r is $|x_0 - r|$, show that after one iteration of Newton's method, the absolute error is now proportional to $|x_0 - r|^2$ assuming that the second derivative is bounded between the approximation x_0 and the root r.

13 [6] Newton's method in one variable has a single initial point x_0, while the secant method requires two initial points x_0 and x_1. Newton's method finds a tangent line at the initial point to find the next approximation, while the secant method uses a secant line through the two points to find the next approximation.

Now, Newton's method in two dimensions has a single initial point (x_0, y_0) and then finds two tangent planes at that point, with the simultaneous root of those two tangent planes being the next approximation to the root. Generalize the secant method so that it, too, may converge to a solution for a system of two equations in two unknowns. You need only explain your steps to how you would take the initial points and how you would find the next approximation.

YOU MAY RIP THIS LAST PAGE OFF

Floating-point representations

±EEMNNN ±M.NNN × 10^{EE-49}

seeeeeeeeeeebbbbbb…b $(-1)^s$ 1.bbbbbb…b × $10^{\text{eeeeeeeeeee} - 01111111111}$

where $01111111111_2 = 1023$.

Fixed-point theorem: Solving $x = f(x)$, choose x_0 and let $x_k \leftarrow f(x_{k-1})$.

Gaussian elimination with partial pivoting is the Gaussian elimination algorithm but always swapping appropriate rows so that the largest entry is in the row that will be used to eliminate that term in all subsequent rows.

$$f(x+h) = \left(\sum_{k=0}^{n} \frac{1}{k!} f^{(k)}(x) h^k \right) + \frac{1}{(n+1)!} f^{(n+1)}(\xi) h^{n+1} \quad \text{where } x < \xi < x+h.$$

$$f(x) = \left(\sum_{k=0}^{n} \frac{1}{k!} f^{(k)}(x_0)(x-x_0)^k \right) + \frac{1}{(n+1)!} f^{(n+1)}(\xi) h^{n+1} \quad \text{where } x_0 < \xi < x.$$

Averaging noisy values with zero bias mitigates the effect, while differentiating noisy values magnifies the effect.

```
double horner( double a[], unsigned int degree; double x ) {
    double result{a[0]};
    for ( std::size_t k{1}; k <= degree; ++k ) {
        result += result*x + a[k];
    }
    return 0;
}
```

Formula of interest:

$$f^{(1)}(x) = \frac{f(x+h) - f(x-h)}{2h} - \frac{1}{6} f^{(3)}(\xi) h^2 \qquad y^{(1)}(t) = \frac{y(t) - y(t-h)}{h} + \frac{1}{2} y^{(2)}(\tau_-) h$$

$$y^{(1)}(t) = \frac{3y(t) - 4y(t-h) + y(t-2h)}{2h} + \frac{1}{3} y^{(3)}(\tau) h^2$$

$$f^{(2)}(x) = \frac{f(x+h) - 2f(x) + f(x-h)}{h^2} - \frac{1}{12} f^{(4)}(\xi) h^2 \qquad y^{(2)}(t) = \frac{y(t) - 2y(t-h) + y(t-2h)}{h^2} + y^{(3)}(\tau) h$$

$$\int_a^b f(x)\,\mathrm{d}x = \left(\frac{1}{2} f(a) + \frac{1}{2} f(b) \right)(b-a) - \frac{1}{12} f^{(2)}(\xi)(b-a)^3$$

$$\int_a^b f(x)\,\mathrm{d}x = \frac{1}{6}(f_0 + 4f_1 + f_2)(b-a) - \frac{1}{2880} f^{(4)}(\xi)(b-a)^5$$

$$\int_a^b f(x)\,\mathrm{d}x = \frac{1}{8}(f_0 + 3f_1 + 3f_2 + f_3)(b-a) - \frac{1}{6480} f^{(4)}(\xi)(b-a)^5$$

$$\int_a^b f(x)\,\mathrm{d}x = \frac{1}{2}\left(f(a) + 2\left(\sum_{k=1}^{n-1} f(a+kh) \right) + f(b) \right) h - f^{(2)}(\xi) \frac{b-a}{12} h^2$$

$$\int_a^b f(x)\,\mathrm{d}x = \frac{1}{3}\left(f_0 + 4\sum_{k=1}^{\frac{n}{2}} f_{2k-1} + 2\sum_{k=1}^{\frac{n}{2}-1} f_{2k} + f_n \right) h - f^{(4)}(\xi) \frac{b-a}{180} h^4$$

$$\int_a^b f(x)\,\mathrm{d}x = \frac{3}{8}\left(f(a) + 3\left(\sum_{k=1}^{\frac{n}{3}} f(a+(3k-2)h) \right) + 3\left(\sum_{k=1}^{\frac{n}{3}} f(a+(3k-1)h) \right) + 2\left(\sum_{k=1}^{\frac{n}{3}-1} f(a+3kh) \right) + f(b) \right) h - f^{(4)}(\xi) \frac{b-a}{80} h^4$$

YOU MAY RIP THIS LAST PAGE OFF

Goal	Estimation
Estimate $y(t_n)$	$0.6\,y_n + 0.4\,y_{n-1} + 0.2\,y_{n-2} - 0.2\,y_{n-4}$
Estimate $y(t_n + h)$	$0.8\,y_n + 0.5\,y_{n-1} + 0.2\,y_{n-2} - 0.1\,y_{n-3} - 0.4\,y_{n-4}$
Estimate the rate of change of y over time	$\frac{0.2y_n + 0.1y_{n-1} - 0.1y_{n-3} - 0.2y_{n-4}}{h}$
Estimate the integral $\int_{t_n-4h}^{t_n} y(t)\,\mathrm{d}t$	$(4h)(0.2y_n + 0.2y_{n-1} + 0.2y_{n-2} + 0.2y_{n-3} + 0.2y_{n-4})$
Estimate the integral $\int_{t_n-h}^{t_n} y(t)\,\mathrm{d}t$	$h(0.5y_n + 0.35y_{n-1} + 0.2y_{n-2} + 0.05y_{n-3} - 0.1y_{n-4})$

Goal	Estimation
Estimate $y(t_n)$	$\frac{1}{35}(31y_n + 9y_{n-1} - 3y_{n-2} - 5y_{n-3} + 3y_{n-4})$
Estimate $y(t_n + h)$	$1.8y_n - 0.8y_{n-2} - 0.6y_{n-3} + 0.6y_{n-4}$
Estimate the rate of change of y over time at time t_n	$\frac{54y_n - 13y_{n-1} - 40y_{n-2} - 27y_{n-3} + 26y_{n-4}}{70h}$
Estimate the acceleration of y over time at time t_n	$\frac{2y_n - y_{n-1} - 2y_{n-2} - y_{n-3} + 2y_{n-4}}{7h^2}$
Estimate the integral $\int_{t_n-4h}^{t_n} y(t)\,\mathrm{d}t$	$(4h)\frac{11y_n + 26y_{n-1} + 31y_{n-2} + 26y_{n-3} + 11y_{n-4}}{105}$
Estimate the integral $\int_{t_n-h}^{t_n} y(t)\,\mathrm{d}t$	$h\frac{230y_n + 137y_{n-1} + 64y_{n-2} + 11y_{n-3} - 22y_{n-4}}{420}$

Method	Requirements	Iteration step	Rate of convergence	Is convergence guaranteed?
Bisection	An interval $[a, b]$ with $f(a)$ having the opposite sign of $f(b)$	Let $c \leftarrow \frac{a+b}{2}$ and update whichever endpoint has the same sign as $f(c)$.	$O(h)$	Yes
Bracketed secant	An interval $[a, b]$ with $f(a)$ having the opposite sign of $f(b)$	Let $c \leftarrow \frac{af(b) - bf(a)}{f(b) - f(a)}$ and update whichever endpoint has the same sign as $f(c)$.	$O(h)$	Yes
Secant	Two initial approximations x_0 and x_1 with $\lvert f(x_0)\rvert > \lvert f(x_1)\rvert$	Let $x_k \leftarrow \frac{x_{k-2} f(x_{k-1}) - x_{k-1} f(x_{k-2})}{f(x_{k-1}) - f(x_{k-2})}$.	$O(h^\phi)$	No
Newton's	An initial approximation x_0	Let $x_k \leftarrow x_{k-1} - \frac{f(x_{k-1})}{f^{(1)}(x_{k-1})}$.	$O(h^2)$	No

Given a function $f(x, y)$ and an approximation to a root (x_k, y_k), we can solve

$$\begin{pmatrix} \frac{\partial}{\partial x} f(x_k, y_k) & \frac{\partial}{\partial y} f(x_k, y_k) \\ \frac{\partial}{\partial x} g(x_k, y_k) & \frac{\partial}{\partial y} g(x_k, y_k) \end{pmatrix} \begin{pmatrix} \Delta x_k \\ \Delta y_k \end{pmatrix} = \begin{pmatrix} -f(x_k, y_k) \\ -g(x_k, y_k) \end{pmatrix}$$

and then let $x_{k+1} \leftarrow x_k + \Delta x_k$, $y_{k+1} \leftarrow y_k + \Delta y_k$.