Instructions

- You may rip off the last three pages as soon as you sit down.
- There are 54 marks available. It will be marked out of 45.
- No aides.
- Turn off all electronic media and store them under your desk.
- You may ask only one question during the examination: "May I go to the washroom?"
- Asking any other question will result in a deduction of 5 marks from the exam grade.
- If you think a question is ambiguous, write down your assumptions and continue.
- Do not leave during first hour or after there are only 15 minutes left.
- Do not stand up until all exams have been picked up.
- There are questions on both sides of the pages.
- 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.
- Answer the questions in the spaces provided. If you require additional space to answer a question, please use the provided blank page and refer to this page in your solutions.

1. **[4]** List the tools described at the start of the course. -1 for each missing or incorrectly listed tool with no minimum grade.

2. **[4]** Show that the error of the approximation of the second derivative

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

is equal to $-\frac{1}{12}f^{(4)}(\xi)h^2$ where $x - h \leq \xi \leq x + h$. You must show and explain each step in your calculations. You should use 3rd-order Taylor series for $f(x+h)$ and $f(x-h)$. Show where you use the intermediate-value theorem.

3. **[3]** Assume that we want to apply Euler's method for approximating the solution to an initial-value problem from t_0 to t_f by breaking the interval into n equally-sized sub-intervals where $t_k = t_0 + hk$ where $h = \frac{t_f - t_0}{n}$. To do the error analysis, we must sum all the errors to get

$$\sum_{k=1}^{n} \frac{1}{2} y^{(2)}(\tau_k) h^2$$

First, describe the bound on each of the τ_k, and then, second, show how this formula can be simplified to

$$\frac{(t_f - t_0)}{2} y^{(2)}(\tau) h$$

and describe how we know that $t_0 \leq \tau \leq t_f$.

4. [**4**] Write down the system of linear equations as a numeric augmented matrix that must be solved to find $\Delta\mathbf{u}_0$ when trying to find the minimum of the function

$$2x^2 + 3y^2 + z^4 + xy - yz + x - z + 1$$

where $\mathbf{u} = \begin{pmatrix} x \\ y \\ z \end{pmatrix}$ and when using the gradient and Newton's method starting with the initial approximation $\mathbf{u}_0 = \begin{pmatrix} 3 \\ 2 \\ 1 \end{pmatrix}$. All the values in the augmented matrix must be numbers, but do not take any steps towards solving this system of linear equations.

5. [**3**] Show that one step of Newton's method for finding a root of a real-valued function of a real variable is $O(h^2)$ where $h = x - r$ is the error of the current approximation with respect to the root r. You should use

$$f(r) = f(x) + f^{(1)}(x)(r - x) + \frac{1}{2}f^{(2)}(\xi)(r - x)^2$$

where r is a root of f. You want to find that C such that $x_1 - r = C(x_0 - r)^2$. You must show your work.

6. [**2**] Given three approximations to a root (x_0, y_0), (x_1, y_1) and (x_2, y_2), for Muller's method, we must shift the x values and interpolate $(x_0 - x_2, y_0)$, $(x_1 - x_2, y_1)$ and $(0, y_2)$; while for the inverse quadratic interpolation, we only interpolate (y_0, x_0), (y_1, x_1) and (y_2, x_2). Provide a justification as to why shifting is not necessary. You may (and probably should) use diagrams.

7. **[3]** Apply one step of Newton's method to find a better approximation of the root of $\cos(x)$ given the initial approximation of the root $x_0 = \frac{\pi}{4}$. Is this a better or worse approximation of the root of the cosine function? Justify your answer. For your justification, you may assume $\pi \approx 3.2$, so $\frac{\pi}{2} \approx 1.6$ and $\frac{\pi}{4} = 0.8$.

8. **[3]** If $A = D + A_{off}$, then the initial approximation of a solution to $A\mathbf{x} = \mathbf{b}$ is $\mathbf{x}_0 \leftarrow D^{-1}\mathbf{b}$. If $A = \begin{pmatrix} 2 & 0 & 0 \\ 1 & 2 & 0 \\ 0 & 1 & 2 \end{pmatrix}$ and $\mathbf{b} = \begin{pmatrix} 1 \\ -1 \\ 1 \end{pmatrix}$, apply one step of the Gauss-Seidel method to find $\mathbf{x}_1$.

9. **[2]** Newton's method in two dimensions finds two tangent planes at a point $\mathbf{x}_k$ (one tangent plane for each function) and then finds the simultaneous zero of those tangent planes as the next approximation. How could you generalize the secant method, which in one dimension requires two initial approximations of the root, so that we can use such a method to find a simultaneous root of two functions of two real variables? This is only two marks, so you simply have to describe the general idea with a justification why you are proposing it.

10. **[2]** Use Euler's method to approximate $y(0.3)$ with $h = 0.1$ for the initial-value problem

$$y^{(1)}(t) = y(t) + t - 1$$

with the initial condition $y(0) = 1$.

11. **[2]** Use one step of Heun's method to approximate $y(0.1)$ with $h = 0.1$ for the initial-value problem

$$y^{(1)}(t) = -50y(t)$$

with the initial condition $y(0) = 1$. Given what you know about analytic solutions to initial-value problems from calculus, is this in any way a reasonable approximation? Why or why not? If it is, justify your answer, and if it is not, suggest what could you do to get a better approximation of $y(0.1)$ while still using Heun's method?

12. **[1]** Circle the correct answer: All the methods we saw for approximating a solution to a single first-order initial-value problem can be used to approximate solutions to a system of first-order initial-value problems, only instead of real arithmetic, we would generally use vector arithmetic, and if we used the absolute value, we could substitute the 2-norm. True or False?

13. **[2]** Convert this third-order initial value problem into a system of first-order initial-value problems:

$$\begin{aligned} y^{(3)}(t) &= \sin(y(t)) + y^{(1)}(t)\cos(y^{(2)}(t)) + 1 \\ y(0) &= 1 \\ y^{(1)}(0) &= 2 \\ y^{(2)}(0) &= 3 \end{aligned}$$

14. **[3]** Suppose you have the boundary-value problem

$$\begin{aligned} u^{(2)}(x) &= \sin(u(x)) + \cos(u^{(1)}(x)) + 1 \\ u(0) = 1 \qquad & u(10) = 2 \end{aligned}$$

and you wanted to approximate the solution by converting it into an initial value problem with $u(0) = 1$ and $u^{(1)}(0) = s$ where you can choose the initial slope. Using the ideas presented in class (remembering that the numbers are nice):

(a) What would you use as an initial slope?

(b) Suppose you used the Dormand-Prince method with this initial slope and you found that your approximation of $u(10)$ was 5.0. What would be your second initial slope?

(c) Suppose you used this second initial slope you calculated, and you found that your approximation of $u(10)$ was 0.5. What would your next initial slope be?

15. [**2**] Write down the system of linear equations that must be solved to approximate, when $h = 1$, a solution to the boundary-value problem

$$\begin{aligned} u^{(2)}(x) + 3u^{(1)}(x) + 2u(x) &= x(5 - x) \\ u(0) &= 1 \\ u(5) &= 2 \end{aligned}$$

Do not attempt to solve this system of linear equations.

16. [**3**] Suppose that you have the heat equation with $\alpha = 1$, $t_0 = 0$, $a = 0$, $b = 4$, and $u_{init}(x) = x(4 - x)$ while $u_a(t) = u_b(t) = 0$. Approximate the solution at $t = 0.1$ using $\Delta t = 0.1$ with $h = 1$. Does your approximation make sense? That is, does the approximation make based on what should happen in reality?

17. [**2**] Assume you know that there is a minimum on the interval $[0, 10]$ for a given function $f(x)$. Assume $\phi^{-1} = 0.618$ and $\phi^{-2} = 0.382$. What are the first two interior points you would check, and then assume that $f(m_1) < f(m_2)$. What would your next four points be (including the new end-points and the new interior points)?

18. **[3]** What is the function of one variable s that you would have to minimize if you were to apply gradient descent to the function $f(x, y) = x^2 + 3y^2 - xy + x - y + 1$ if the initial approximation was $x = 1$ and $y = 1$.

19. **[6]** Propose two different methods that could be used to approximate finding an inflection point. If any tools are used, you only have to explain how to use the tools, you don't have to find the exact formulas. For example, you could write "find the interpolating polynomial between the points ..." There are at least three methods that can be generalized.

USE THIS PAGE IF ADDITIONAL SPACE IS REQUIRED
Clearly state the question number being answered and refer the marker to this page.

YOU MAY RIP THESE LAST THREE PAGES OFF

Floating-point representations: ±EENMMM represents $\pm\mathtt{N.MMM} \times 10^{\mathtt{EE}-49}$ and the 64 bits

$$\mathtt{seeeeeeeeeeebbbbbb\cdots b}$$

represents

$$(-1)^{\mathtt{s}}\mathtt{1.bbbbbb\cdots b} \times 2^{\mathtt{eeeeeeeeeee}-01111111111}$$

where 0b01111111111 = 1023 = 0x3ff. Recall 1 is +491000 or 0x3ff0000000000000.
Fixed-point theorem: To approximate a solution to $x = f(x)$, choose x_0 and let $x_k \leftarrow f(x_{k-1})$.
Gaussian elimination with partial pivoting: This is the Gaussian elimination algorithm but always swapping appropriate rows so that the largest entry is in the pivot position (the row that will be used to eliminate that term in all subsequent rows).
n^{th}-order Taylor series: If h is small, expanding around x yields:

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

where $x \le \xi \le x+h$. Otherwise, if x is close to x_0, expanding around x_0 yields:

$$f(x) = \left(\sum_{k=0}^{n} \frac{1}{k!} f^{(k)}(x_0) (x-x_0)^k\right) + \frac{1}{(n+1)!} f^{(n+1)}(\xi) (x-x_0)^{n+1}$$

where $x_0 \le \xi \le x$.

```
double horner( double a[], unsigned int const degree, double const x ) {
    // The coefficient of xˆk is a[k]
    double result{ a[degree] };

    for ( std::size_t k{degree - 1}; k < degree; --k ) {
        result = result*x + a[k];
    }

    return result;
}
```

Noise: Averaging noisy values with zero bias mitigates the effect, while differentiating noisy values magnifies the effect. Use interpolating polynomials if the data is accurate and precise, but use least squares best-fitting polynomials if the data is accurate but not precise (that is, the data has significant noise). If the data is not accurate, we cannot recover the underlying signal.
Evaluating interpolating polynomials: For interpolating between t_k and t_{k-1} where t_k is the time of the most recent data point, shift and scale to $\ldots, -2.5, -1.5, -0.5$ and 0.5 to ensure that $-0.5 < \delta < 0.5$ to evaluate the polynomial at the point $\frac{t_{k-1}+t_k}{2} + \delta h$ where h is the time step between readings. Note, you do not have to know these formulas explicitly; rather, you must understand the idea behind deriving these. For example, why to we shift and scale so that our choice of δ is such that $|\delta| < 0.5$.
Derivatives:
Centered three-point:

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

Backward two-point:

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

Backward three-point:

$$y^{(1)}(t) = \frac{3y(t) - 4y(t-h) + y(t-2h)}{2h} + \frac{1}{3} y^{(3)}(t) h^2 + \mathrm{O}\left(h^3\right)$$

Second derivatives:

Centered three-point:

$$f^{(2)}(x) = \frac{f(x+h) - 2f(x) + f(x-h)}{h^2} - \frac{1}{12} f^{(4)}(\xi)\, h^2$$

Backward three-point:

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

Backward four-point:

$$y^{(2)}(t) = \frac{2y(t) - 5y(t-h) + 4y(t-2h) - y(t-3h)}{h^2} + \frac{11}{12} y^{(4)}(t)\, h^2 + \mathrm{O}\left(h^3\right)$$

Integrals:
Two-point (trapezoidal rule):

$$\int_{x_{k-1}}^{x_k} f(x)\,\mathrm{d}x = \left(\frac{1}{2} f(x_{k-1}) + \frac{1}{2} f(x_k)\right) h - \frac{1}{12} f^{(2)}(\xi)\, h^3$$

Centered four-point:

$$\int_{x_{k-1}}^{x_k} f(x)\,\mathrm{d}x = \left(-\frac{1}{24} f(x_{k-2}) + \frac{13}{24} f(x_{k-1}) + \frac{13}{24} f(x_k) - \frac{1}{24} f(x_{k+1})\right) h - \frac{11}{720} f^{(4)}(t_k)\, h^5 + \mathrm{O}\left(h^6\right)$$

Simpson's rule:

$$\int_{x_{k-1}}^{x_{k+1}} f(x)\,\mathrm{d}x = \left(\frac{1}{6} f(x_{k-1}) + \frac{4}{6} f(x_k) + \frac{1}{6} f(x_{k+1})\right) (2h) - \frac{1}{90} f^{(4)}(\xi)\, h^5$$

Backward three-point (half Simpson's rule):

$$\int_{t_{k-1}}^{t_k} y(t)\,\mathrm{d}x = \left(\frac{5}{12} y(t_k) + \frac{8}{12} y(t_{k-1}) - \frac{1}{12} y(t_{k-2})\right) h - \frac{1}{24} y^{(3)}(t_k)\, h^4 + \mathrm{O}\left(h^5\right)$$

Backward four-point:

$$\int_{t_{k-1}}^{t_k} y(t)\,\mathrm{d}x = \left(\frac{9}{24} y(t_k) + \frac{19}{24} y(t_{k-1}) - \frac{5}{24} y(t_{k-2}) + \frac{1}{24} y(t_{k-3})\right) h + \frac{19}{720} y^{(4)}(t_k)\, h^5 + \mathrm{O}\left(h^6\right)$$

As Simpson's rule spans two time intervals, it is less useful, but it is interesting with its comparison with the trapezoidal rule applied twice versus one application of Simpson's rule.
Any integral formula can be applied repeatedly on the interval $[a, b]$ by dividing the interval into n equally-spaced sub-intervals of width $h = \frac{b-a}{n}$ and then setting $x_k = a + kh$ or $t_k = a + kh$.

Least squares: In general, if we want to find the best approximation of an n-dimensional vector $\mathbf{y}$ by a linear linear combination of m vectors $\mathbf{v}_1, \ldots, \mathbf{v}_m$ (where $m < n$), we create the matrix $V = (\mathbf{v}_1 \cdots \mathbf{v}_m)$ and solve $V^T V\vec{\alpha} = V^T\mathbf{y}$. More specific to this course, having shifted and scaled the n most recent t-values onto $0, -1, -2, \ldots, -n+1$, with y values $\mathbf{y} = (y_k, y_{k-1}, y_{0-2}, \ldots, y_{k-n+1})$, we solve $V^T V\vec{\alpha} = V^T\mathbf{y}$ for the coefficients of the least-squares best-fitting polynomial, generally of degree one (linear or $\alpha_1 t + \alpha_0$) or two (quadratic or $\alpha_2 t^2 + \alpha_1 t + \alpha_0$). We can find the $2 \times n$ or $3 \times n$ matrix to calculate $\vec{\alpha} = \left(V^T V\right)^{-1} V^T\mathbf{y}$.

Value being estimated	Linear estimation
$y(t_k)$	α_0
$y(t_k + h)$	$\alpha_0 + \alpha_1$
$y^{(1)}(t_k)$	α_1/h
$\int_{t_k-h}^{t_k} y(t)\mathrm{d}t$	$(\alpha_0 - \alpha_1/2)h$
$\int_{t_k}^{t_k+h} y(t)\mathrm{d}t$	$(\alpha_0 + \alpha_1/2)h$

Value being estimated	Quadratic estimation
$y(t_k)$	α_0
$y(t_k + h)$	$\alpha_0 + \alpha_1 + \alpha_2$
$y^{(1)}(t_k)$	α_1/h
$y^{(2)}(t_k)$	$2\alpha_2/h^2$
$\int_{t_k-h}^{t_k} y(t)\mathrm{d}t$	$(\alpha_0 - \alpha_1/2 + \alpha_2/3)h$
$\int_{t_k}^{t_k+h} y(t)\mathrm{d}t$	$(\alpha_0 + \alpha_1/2 + \alpha_2/3)h$

References to both binary search and interpolation search are not applicable to this course. Instead, they are introduced into the course to demonstrate parallels between the binary search and bisection method, and interpolation search and the bracketed secant method.

Bisection: Given an interval $[a, b]$ with $f(a)$ and $f(b)$ having opposite signs, let $m \leftarrow \frac{a+b}{2}$ and update whichever endpoint has the same sign as $f(m)$. $O(h)$.

Bracketed secant: Given an interval $[a, b]$ with $f(a)$ and $f(b)$ having opposite signs, let $c \leftarrow \frac{af(b)-bf(a)}{f(b)-f(a)}$ and update whichever endpoint has the same sign as $f(c)$. $O(h)$.

Secant: Given two initial approximations x_0 and x_1 with $|f(x_0)| > |f(x_1)|$, let $x_2 \leftarrow \frac{x_0 f(x_1)-x_1 f(x_0)}{f(x_1)-f(x_0)}$. $O(h^{1.618})$.

Muller's: Given three initial approximations, interpolate $(x_0 - x_2, y_0)$, $(x_1 - x_2, y_1)$ and $0, y_2)$ and find the smaller root of the interpolating quadratic, call this δ and set $x_3 = x_2 + \delta$. $O(h^{1.839})$.

Inverse quadratic interpolation: Given three initial approximations, interpolate (y_0, x_0), (y_1, x_1) and (y_2, x_2) and find let x_3 be the constant coefficient of this interpolating quadratic polynomial. $O(h^{1.839})$.

Newton's: Given an initial approximation x_0, let $x_1 \leftarrow x_0 - \frac{f(x_0)}{f^{(1)}(x_0)}$. $O(h^2)$.

Fixed-point iteration for systems of linear equations: Given a square $n \times n$ matrix A, if D_A is the matrix corresponding to the diagonal entries of A, and A_{off} consists of all the off-diagonal entries of A (so $A = D_A + A_{\text{off}}$), the we can rewrite $A\mathbf{x} = \mathbf{b}$ as the equation $\mathbf{x} = D_A^{-1}(\mathbf{b} - A_{\text{off}}\mathbf{x})$. Let the entries of $\mathbf{x}_k = (x_{k,1}, x_{k,2}, ..., x_{k,n})$.
Jacobi: Set $x_k \leftarrow D_A^{-1}(\mathbf{b} - A_{\text{off}}\mathbf{x}_{k-1})$.
Gauss-Seidel: Set $\mathbf{x}_k \leftarrow \mathbf{x}_{k-1}$, and then for i from 1 to n, update $x_{k,i} \leftarrow D_A^{-1}(b_i - A_{\text{off};i,\cdots}\mathbf{x}_k)$ where $A_{\text{off};i,\cdots}$ is the i^{th} row of A_{off}.
Successive over-relaxation: Given the approximation x_{k-1} and having iterated an algorithm once more to get the approximation x_k, we can move an addition $100\alpha\%$ in the direction of the newer approximation by updating $x_k \leftarrow (1+\alpha)x_k - \alpha x_{k-1}$.
Newton's method in two dimensions: Given functions $f(x,y)$ and $g(x,y)$ and an approximation to a simultaneous 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 $x_{k+1} \leftarrow x_k + \Delta x_k$ and $y_{k+1} \leftarrow y_k + \Delta y_k$.
Newton's method in n dimensions: More generally, approximating $\mathbf{f}(\mathbf{x}) = \mathbf{0}$, given an approximation $\mathbf{x}_k$, solve $\mathbf{J}(\mathbf{f})(\mathbf{x}_k)\Delta\mathbf{x}_k = -\mathbf{f}(\mathbf{x}_k)$ and then let $\mathbf{x}_{k+1} \leftarrow \mathbf{x}_k + \Delta\mathbf{x}_k$.

Initial-value problems (IVPs): Given the ordinary-differential equation (ODE) and initial value

$$y^{(1)}(t) = f(t, y(t)) \text{ and } y(t_0) = y_0,$$

we will approximate $y_{k+1} \approx y(t_{k+1})$.

Given the system of ODEs and initial values

$$\mathbf{y}^{(1)}(t) = \mathbf{f}(t, \mathbf{y}(t)) \text{ and } \mathbf{y}(t_0) = \mathbf{y}_0$$

we will approximate $\mathbf{y}_{k+1} \approx \mathbf{y}(t_{k+1})$.

Euler's method:

$$y_{k+1} \leftarrow y_k + hf(t_k, y_k) \quad \mathbf{y}_{k+1} \leftarrow \mathbf{y}_k + h\mathbf{f}(t_k, \mathbf{y}_k)$$

Heun's method:

$$\begin{array}{ll} s_0 \leftarrow f(t_k, y_k) & \mathbf{s}_0 \leftarrow \mathbf{f}(t_k, y_k) \\ s_1 \leftarrow f(t_k + h, y_k + hs_0) & \mathbf{s}_1 \leftarrow \mathbf{f}(t_k + h, \mathbf{y}_k + h\mathbf{s}_0) \\ y_{k+1} \leftarrow y_k + h\frac{s_0+s_1}{2} & \mathbf{y}_{k+1} \leftarrow \mathbf{y}_k + h\frac{\mathbf{s}_0+\mathbf{s}_1}{2} \end{array}$$

The 4^{th}-order Runge-Kutta method:

$$\begin{array}{ll} s_0 \leftarrow f(t_k, y_k) & \mathbf{s}_0 \leftarrow \mathbf{f}(t_k, y_k) \\ s_1 \leftarrow f\left(t_k + \frac{1}{2}h, y_k + \frac{1}{2}hs_0\right) & \mathbf{s}_1 \leftarrow \mathbf{f}\left(t_k + \frac{1}{2}h, \mathbf{y}_k + \frac{1}{2}h\mathbf{s}_0\right) \\ s_2 \leftarrow f\left(t_k + \frac{1}{2}h, y_k + \frac{1}{2}hs_1\right) & \mathbf{s}_2 \leftarrow \mathbf{f}\left(t_k + \frac{1}{2}h, \mathbf{y}_k + \frac{1}{2}h\mathbf{s}_1\right) \\ s_3 \leftarrow f(t_k + h, y_k + hs_2) & \mathbf{s}_3 \leftarrow \mathbf{f}(t_k + h, \mathbf{y}_k + h\mathbf{s}_2) \\ y_{k+1} \leftarrow y_k + h\frac{s_0+2s_1+2s_2+s_3}{6} & \mathbf{y}_{k+1} \leftarrow \mathbf{y}_k + h\frac{\mathbf{s}_0+2\mathbf{s}_1+2\mathbf{s}_2+\mathbf{s}_3}{6} \end{array}$$

A single step of these three methods are $O(h^2)$, $O(h^3)$ and $O(h^5)$, respectively; however, multiple steps are one order less: $O(h)$, $O(h^2)$ and $O(h^4)$, respectively.

For Euler's method, the error of one step may be found from Taylor series:

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

where

$$y(t_{k+1}) = y_k + f(t_k, y_k)h + \frac{1}{2}y^2(\tau)h^2$$

assuming that y_k is exact.

The adaptive Euler-Heun method Given an ODE and an initial value together with a maximum absolute error per unit step ϵ_{step}, start with an initial step size h and determine both minimum and maximum step sizes h_{min} and h_{max}, respectively. Also, let $k \leftarrow 0$.

1. If $h < h_{min}$, set $h \leftarrow h_{min}$, and if $h > h_{max}$, set $h \leftarrow h_{max}$.
2. Given y_k, calculate $s_0 \leftarrow f(t_k, y_k)$ and $s_1 \leftarrow f(t_k + h, y_k + hs_0)$.
3. Estimate $y(t_k + h)$ with
   - $y \leftarrow y_k + hs_0$ (the worse approximation), and
   - $z \leftarrow y_k + h\frac{s_0+s_1}{2}$ (the better approximation).
4. Let $a \leftarrow \frac{h\epsilon_{step}}{2|y-z|}$, and
   - if $a \geq 1$ or $h = h_{min}$, let $t_{k+1} \leftarrow t_k + h$ and let $y_{k+1} \leftarrow z$ and then set $h \leftarrow 0.9ah$ and $k \leftarrow k + 1$, and we will continue with the next step;
   - otherwise $a < 1$ and we will try again.
5. If $0.9a \leq 0.5$, set $h \leftarrow 0.5h$ (don't shrink h by more than a factor of two),
   else if $0.9a \geq 2$, set $h \leftarrow 2h$ (don't grow h by more than a factor of two),
   else set $h \leftarrow 0.9ah$.

Given a system of ODEs and initial values with a similar set-up:

1. If $h < h_{min}$, set $h \leftarrow h_{min}$, and if $h > h_{max}$, set $h \leftarrow h_{max}$.
2. Given y_k, calculate $\mathbf{s}_0 \leftarrow \mathbf{f}(t_k, \mathbf{y}_k)$ and $\mathbf{s}_1 \leftarrow \mathbf{f}(t_k + h, \mathbf{y}_k + h\mathbf{s}_0)$.
3. Estimate $\mathbf{y}(t_k + h)$ with
   - $\mathbf{y} \leftarrow \mathbf{y}_k + h\mathbf{s}_0$ (the worse approximation), and
   - $\mathbf{z} \leftarrow \mathbf{y}_k + h\frac{\mathbf{s}_0+\mathbf{s}_1}{2}$ (the better approximation).
4. Let $a \rightarrow \frac{h\epsilon_{step}}{2\|\mathbf{y}-\mathbf{z}\|_2}$ where $\|\cdot\|_2$ is the 2-norm (or Euclidean norm), and
   - if $a \geq 1$ or $h = h_{min}$, let $t_{k+1} \leftarrow t_k + h$ and let $\mathbf{y}_{k+1} \leftarrow \mathbf{z}$ and then set $h \leftarrow 0.9ah$ and $k \leftarrow k + 1$, and we will continue with the next step;
   - otherwise $a < 1$ and we will try again.
5. If $0.9a \leq 0.5$, set $h \leftarrow 0.5h$ (don't shrink h by more than a factor of two),
   else if $0.9a \geq 2$, set $h \leftarrow 2h$ (don't grow h by more than a factor of two),
   else set $h \leftarrow 0.9ah$.

Boundary-value problems (BVPs) Given a 2$^{\text{nd}}$-order ODE $u^{(1)}(x) = f(x, u(x), u^{(1)}(x))$ with two boundary conditions $u(a) = u_a$ and $u(b) = u_b$, we can approximate a solution to this BVP as follows. First, create the IVP $u^{(1)}(x) = f(x, u(x), u^{(1)}(x))$ with the two initial conditions $u(a) = u_a$ and $u^{(1)}(a) = s$ where s is an initial slope we can choose. Let us use any technique you wish (preferably Dormand Prince), and let $u_s(x)$ be an approximation to the solution of this IVP with the initial slope s. Then the approximation of $u(b)$ using the this technique and initial slope is $u_s(b)$. Proceed as follows:

1. Let $s_0 = \frac{u_b-u_a}{b-a}$ and find $u_{s_0}(b)$. If $u_{s_0}(b) = u_b$, we are done, otherwise, continue.
2. Let $s_1 = \frac{2u_b-u_{s_0}(b)-u_a}{b-a}$ and find $u_{s_1}(b)$. If $u_{s_1}(b) = u_b$, we are done, otherwise, continue.
3. Define the function $f(s) = u_b - u_s(b)$, which is a function of a single variable s, and use s_0 and s_1 as the first two approximations for the secant method.

Linear boundary-value problems (BVPs) Given a 2nd-order linear ODE (LODE) $c_2(x)u^{(2)}(x) + c_1(x)u^{(1)}(x) + c_0(x)u(x) = g(x)$ with two boundary conditions $u(a) = u_a$ and $u(b) = u_b$, we convert the LODE into a linear finite-difference equation and let $x_k = a + hk$ for $h = \frac{b-a}{n}$, so if we define $p_k = 2c_2(x_k) - hc_1(x_k)$, $q_k = -4c_2(x_k) + 2h^2c_0(x_k)$ and $r_k = 2c_2(x_k) + hc_1(x_k)$, we have

$$p_k u_{k-1} + q_k u_k + r_k u_{k+1} = 2h^2 g(x_k)$$

For Dirichlet boundary conditions, the corresponding linear equations are:

$$q_1 u_1 + r_1 u_2 = 2h^2 g(x_1) - p_1 u_a$$

$$p_{n-1} u_{n-2} + q_{n-1} u_{n-1} = 2h^2 g(x_{n-1}) - r_{n-1} u_b$$

For Neumann boundary conditions, the corresponding linear equations are:

$$\left(q_1 + \frac{4}{3}p_1\right) u_1 + \left(r_1 - \frac{1}{3}p_1\right) u_2 = 2h^2 g(x_1) + \frac{2}{3}hp_1 u_a^{(1)}$$

$$\left(p_{n-1} - \frac{1}{3}r_{n-1}\right) u_{n-2} + \left(q_{n-1} + \frac{4}{3}r_{n-1}\right) u_{n-1} = 2h^2 g(x_{n-1}) - \frac{2}{3}hr_{n-1} u_b^{(1)}$$

after which

$$u_0 \leftarrow -\frac{2}{3}hu_a^{(1)} + \frac{4}{3}u_1 - \frac{1}{3}u_2$$

$$u_n \leftarrow \frac{2}{3}hu_b^{(1)} + \frac{4}{3}u_{n-1} - \frac{1}{3}u_{n-2}$$

and this simplifies for insulated boundary conditions where $u_a^{(1)} = 0$ or $u_b^{(1)} = 0$.

Heat equation: For the heat equation $\frac{\partial u}{\partial t} = \alpha \nabla^2 u$, we have $u_{init}(x)$, $u_a(t)$ and $u_b(t)$, and we convert the partial-differential equation (PDE) into a finite-difference equation with $x_k \leftarrow a + kh$ and $t_\ell \leftarrow t_0 + \ell\Delta t$ so that $u_{k,0} \leftarrow u_{init}(x_k)$,

$$u_{k,\ell+1} \leftarrow u_{k,\ell} + \frac{\alpha\Delta t}{h^2}\left(u_{k-1,\ell} - 2u_{k,\ell} + u_{k+1,\ell}\right)$$

where $u_{0,\ell} \leftarrow u_a(t_\ell)$ and $u_{n,\ell} \leftarrow u_b(t_\ell)$.

Wave equation: For the heat equation $\frac{\partial^2 u}{\partial t^2} = c^2 \nabla^2 u$, we have $u_{init}(x)$, $\dot{u}_{init}(x)$, $u_a(t)$ and $u_b(t)$, and we convert the PDE into a finite-difference equation with $x_k \leftarrow a + kh$ and $t_\ell \leftarrow t_0 + \ell\Delta t$ so that $u_{k,0} \leftarrow u_{init}(x_k)$,

$$u_{k,1} \leftarrow u_{k,0} + \dot{u}_{init}(x_k)\Delta t + \frac{1}{2}\left(\frac{c\Delta t}{h}\right)^2 \left(u_{k-1,0} - 2u_{k,0} + u_{k+1,0}\right)$$

$$u_{k,\ell+1} \leftarrow 2u_{k,\ell} - u_{k,\ell-1} + \left(\frac{c\Delta t}{h}\right)^2 \left(u_{k-1,\ell} - 2u_{k,\ell} + u_{k+1,\ell}\right)$$

where $u_{0,\ell} \leftarrow u_a(t_\ell)$ and $u_{n,\ell} \leftarrow u_b(t_\ell)$.

On a grid $x_i = a_x + ih$, $y_j \leftarrow a_y + jh$ and $z_k \leftarrow a_z + kh$ we convert the PDE into the linear finite difference equation:

$$6u_{i,j,k} - u_{i-1,j,k} - u_{i+1,j,k} - u_{i,j-1,k} - u_{i,j+1,k} - u_{i,j,k-1} - u_{i,j,k+1} = 0$$

replacing any points on the boundary with their boundary value.

Newton's: Use Newton's method on $f^{(1)}(x)$.
Golden ratio search: To minimize, given an interval $[a, b]$ with a maximum on that interval, let $m_1 \leftarrow b - (b-a)/\phi$ and $m_2 \leftarrow a + (b-a)/\phi$ and update $a \leftarrow m_1$ if $f(m_2) < f(m_1)$ and update $b \leftarrow m_2$ otherwise.
Successive parabolic interpolation: Formula not necessary: given three points, find the interpolating polynomial $ax^2 + bx + c$ and let the next point be $-\frac{b}{2a}$.
Newton's: Use Newton's method on $\vec{\nabla} f$.
Gradient descent: Calculate or approximate $\vec{\nabla} f(\mathbf{x}_k)$, and then use a one-dimensional algorithm to find a minimum of $f(\mathbf{x}_k - s\vec{\nabla} f(\mathbf{x}_k))$ and when we find the value s_k that gives us the minimum, set $\mathbf{x}_{k+1} \leftarrow \mathbf{x}_k - s_k \vec{\nabla} f(\mathbf{x}_k)$.