# Math 373: 01 — Fall 2000
# MW8 SC–205
# Prof. Bumby

## Study Guide for Final Exam

**Rootfinding** The first main topic of the course was the solution of $f(x) = 0$ for some function $f$ that we assume can be computed accurately. Sometimes this is written as a *fixed-point* problem $x = g(x)$. A fixed-point problem becomes a rootfinding problem by collecting terms, e.g. writing $f(x) = x - g(x)$. A rootfinding problem may also be turned into a fixed-point problem: one way to do this is to isolate a term in the expression for $f(x)$ and solve for the $x$ in this term in terms of the rest of $f(x)$; a more subtle way to get a fixed-point problem is to let $g(x)$ be the expression arising from Newton's method $g(x) = x - f(x)/f'(x)$.

When phrased as a fixed-point problem, it is natural to **iterate** the function $g$ in the hope that one gets a convergent sequence by picking a value of $x_0$ and setting $x_{n+1} = g(x_n)$ for $n = 0, 1, 2, \ldots$.

One sure way to solve $f(x) = 0$ is to locate an interval $I$ on which $f$ changes sign and repeatedly **bisect** $I$, keeping the half where there is a change of sign. The flow of this method is determined only by the sign of $f$ at each new point, but the size of $f$ may be useful as a check for mistakes and a warning about limited accuracy in the computation of $f(x)$.

**Exercise 1**. The function

$$f(x) = x^3 - x\cos(50x) - 1$$

appeared on the midterm exam. A graph given on the exam showed six changes of sign between 0.5 and 1.0. Bisection will work on any interval on which the function changes sign. This function is negative at both 0.5 and 1.0, but it is positive 0.8, so we could take $I = [0.5, 0.8]$ or $[0.8, 1.0]$. To find all six roots, you would need to separately start from intervals such as $[0.68, 0.69]$, $[0.69, 0.70]$, $[0.7, 0.8]$, $[0.8, 0.9]$, $[0.90, 0.95]$ and $[0.95, 1.00]$. The initial intervals were chosen to have short terminating decimal expansions, although later steps of the method will not be particularly suited to the decimal representation. Long before you have the answer to a reasonable calculator accuracy of 10 decimal places, you will be forced to round-off the values of the endpoints. At any rate, a bisection program can find all roots easily to this accuracy. **Do this**.

Our first treatment of rootfinding avoided the question of how accurately the function $f$ could be computed. However, it is fairly clear that the cosine can be computed to about the same absolute accuracy as its argument. However, if the argument is $50x$, this will be two fewer places than the accuracy of $x$. This means that we would need to keep about 12 decimal places of $x$ in order to recognize the roots to 10 decimal places. You will see this by the difference between function values being much larger than the difference between the arguments.

It is more troublesome to have $f(x)$ being small over a large interval, since higher accuracy will be needed to detect a change of sign. This can happen if $f(x) = f'(x) = 0$ at some point (a multiple root) or is two roots are very close. In exercise 1, the two roots near 0.7 are harder to isolate than the other four roots, and need to be treated with more care in any rootfinding method.

In the computation of a function, the largest term sets the scale for the computation, so you will compute the function to a fixed accuracy relative to the largest term. The presence of a double root means that a function whose terms are all roughly 1 in size will be about $10^{-10}$ on an interval of length about $10^{-5}$, so high accuracy rootfinding will be impossible unless some way is found to compute the function more accurately.

This also affects Newton's method. This method shows quadratic convergence, which means that, *for some scale*, the distance to root is squared at each step. When this distance is less than one, it means that the number of decimal places of accuracy doubles at each step. This is great! If you have 5 decimal place accuracy, you can expect 10 place accuracy at the next iteration, and one more iteration would give 20 places if you are able to compute to that accuracy. However, this depends on the scale. For Newton's method, it is easy to see that the *unit* interval has a value of approximately $2\min(f'(x))/\max(f''(x))$.

**Exercise 2**. Find starting values such that Newton's method will converge to each of the roots of the function in Exercise 1. Keep track of the values of $f(x)$ and $f'(x)$ and use this to estimate the rate of convergence. How many additional steps would be needed to get 100 place accuracy for the root near 0.97 (assuming the use of a system allowing that accuracy) after you have 10 place accuracy? (Although $f''$ appears in the error estimate used to find the scale, it appears only through its maximum value, which is clearly about 2500. Use of this bound will give a conservative estimate on the scale for quadratic convergence.)

The formula for the expression to be iterated in Newton's method depends on the form of the function $f$ and not just on the root. Different expressions obtained by Newton's method should have comparable rates of convergence, but there may be a difference in ease of computation.

If you have a linearly convergent iteration that converges to a fixed point of a function $g$, Steffensen's method obtains an iteration that is quadratically convergent. Steffensen's method produces a sequence of triples $(p_0^{(i)}, p_1^{(i)}, p_2^{(i)})$ starting from $p_0^{(0)}$ where $p_1^{(i)} = g(p_0^{(i)})$, $p_2^{(i)} = g(p_1^{(i)})$ and

$$p_0^{(i+1)} = p_0^{(i)} - \frac{(p_1^{(i)} - p_0^{(i)})^2}{p_2^{(i)} - 2p_1^{(i)} + p_0^{(i)}}.$$

The values of $p_1^{(i)} - p_0^{(i)}$ should be recorded to keep track of the rate of convergence. On an appropriate scale, there is quadratic convergence, so this quantity is squared from one value of $i$ to the next.

**Exercise 3**. Find an interval on which iterating $g(x) = (x^4 - 1)/5$ converges to a root of $x^4 - 5x - 1 = 0$, and apply Steffensen's method to give a rapidly convergent process for finding that root. Then, find a method that will converge to the other real root of this equation.

**Interpolation** The Lagrange form of the interpolation formula gives a quick proof that a polynomial of degree $n$ can be found that interpolates the values at $n + 1$ different points. However, it has some computational limitations. Divided differences lead to computationally superior expressions for the values of the interpolating polynomial because the terms in the Newton series usually decrease in size and any loss of accuracy in the computation of the coefficients (which are the divided differences) is balanced by the polynomial it multiplies being uniformly small on the interval of interest.

Traditionally, interpolation formulas were given based on values at equally spaced points, but this is not an essential part of either the Lagrange or Newton formula. In the last problem on the midterm exam, a six decimal place table of a function was given at intervals of 0.1 from 1.0 to 2.0, omitting 1.3 and the task was to reconstruct the missing value. The intended answer was to use an interpolation formula using values at 1.1, 1.2, 1.4, and 1.5 since the polynomial $\prod(x - x_i)$ appearing in the error term is smaller at $x = 1.3$ for this choice of $x_i$ than for other available choices of four points.

**Exercise 4**. Extend the computation to find an interpolating polynomial of degree 5 by adding the value of the function at two more points. Notice the effect of roundoff on the accuracy of the coefficients, but also record the size of each new polynomial term at $x = 1.3$ and show that the contribution of roundoff error remains around $10^{-6}$.

**Exercise 5**. Use the same table, but a different interpolating polynomial, to estimate the function at $x = 1.73$ to four decimal places. If you choose too high a degree, you will see new terms having a negligible effect on the value.

**Numerical Differentiation** One application of interpolation formulas is that they lead to formulas estimating the derivative of a function at a point in terms of values at nearby points. Again, the traditional formulas are based on equally spaced points, but the methods for deriving the formulas have no such limitation. The symmetrical 3 and 5 point formulas (equations 4.5 and 4.6 of the text) are the most accurate for the effort used in their computation, but they require that the function be tabulated on both sides of the node at which the derivative is sought.

**Exercise 6**. Continuing with the data from the last question from the midterm exam, estimate $f'(1.3)$ and $f'(1.7)$ using equation 4.6. Also, use equation 4.7 with $h = -0.1$ to estimate $f'(2.0)$.

If a procedure is given for evaluating the function at arbitrary points, then the truncation error in these formulas can be made small by making the step size $h$ small. However, if $f(x)$ is only computed to a fixed accuracy $\epsilon$, there will be roundoff error of about $\epsilon/h$. Thus, if it is not possible to decrease $\epsilon$, the roundoff error will begin to dominate the truncation error when $h$ is made very small. The combined error will be smallest when roundoff and truncation errors are about the same size. For the example taken from the exam, we have $\epsilon = 10^{-6}$ and $h = 0.1$, so seven point formulas would not give much improvement over the five point formulas used in exercise 6. Likewise, a more finely spaced table would give little benefit unless the accuracy of the values was also increased. In the homework problems (3b and 4b) from Section 4.1, the five place accuracy of the given values gave a roundoff error that threatened to be as large as $10^{-4}$ with $h = 0.1$. Indeed, the observed difference in the value of $f'(8.3)$ is slightly larger than the truncation error for the formula used.

**Exercise 7**. Repeat the exercise from section 4.1 using an 8 decimal place table derived from the exact formula for $f(x)$ and note the effect on the error in the value of $f'(x)$ for the given values.

**Power Series** Another way to approximate a function is through the Taylor Series. The most useful series are the series for $e^x$, $\cos(x)$, and $\sin(x)$ in powers of $x$, i.e. with $x_0 = 0$ in Theorem 1.14. The generality of that statement of the theorem is of very little practical use since the same same effect can be obtained by changing the function to $f(x_0 + x)$. The important part of Taylor series is the ability to find them easily for some functions and the fact that the error estimate is always *something like* the next term. In particular, the error term for the series of $\cos(x)$ can be used to give a much easier estimate on the value of $\cos(t^2)$ than comes from the error estimate for the Taylor series of that function, even though the main part of the series will be identical.

Since the Taylor polynomials are polynomials they can be easily computed to high accuracy. They form the basis of many methods that appear to give the values of transcendental functions on demand. As suggested by a problem on the midterm exam, one way to compute trig functions everywhere is to use the series to get desired accuracy on an interval like $[-\pi/12, \pi/12]$, and use the addition formula together with the known exact values at some special points to compute the functions at other points. This requires that an accurate value of $\pi$ be computed *once*, and stored in the program for computing the trig functions.

**Exercise 8**. Maple gives $\pi = 3.14159265358979323 85$ and $\sqrt{3} = 1.7320508075688772935$ when set for 20 digit accuracy. Since $\cos\pi/6 = \sqrt{3}/2$ and $\sin\pi/6 = 1/2$ and

$$\cos(\pi/6 + x) = \cos\pi/6\cos(x) - \sin\pi/6\sin(x)$$

set up a process to evaluate $\cos(0.4)$ in terms of $\sin(x)$ and $\cos(x)$ at $x = 0.4 - \pi/6$ (whose absolute value is less the 1/8), and evaluate these quantities by a Taylor polynomial that will give 20 place accuracy. If you don't have a convenient way to do 20 digit addition and multiplication, you can skip the details of the computation, but you should describe the process for obtaining these values.

**Numerical Integration** Integrating an interpolation formula gives an expression that estimates the integral of a function on an interval in terms of function values at selected points. Typically, the points are chose equally spaced, although this is usually not optimal. Moreover, using higher degree interpolation formulas does not produce significant improvement in accuracy unless the functions are very tamely smooth.

A better idea is to use a composite rule. The interval is divided into a collection of very small intervals and the simple rule is used on each small interval. Accuracy follows from the step size being small even though the error term involves only a modest power of the step size. A calculator will typically use Simpson's rule with either 64 or 128 subintervals to get close to full accuracy quickly.

**Exercise 9** What are the main term and error term for Simpson's rule using the values of $f(k/128)$ to estimate

$$\int_0^1 f(x)\,dx.$$

In particular, if you use Theorem 4.4, you should identify the parameters in the statement of that theorem in terms of the description given here.

If a program is written to compute integrals, there is no difficulty including an extrapolation step giving the Romberg method. If the integrand is smooth, this will lead to excellent results with little effort.

Functions whose higher derivatives do not remain small require a different approach. Adaptive methods vary the step size to meet projections on the error.

Both of these approaches are beyond the scope of the exam, although they are likely to be among the more practical topics considered in the course.

In all integration methods, it s most important that programs use the data correctly. When summarized by formulas, it is not always clear what different quantities represent. One approach to writing programs is to compute an *average*, which is converted to an integral at the end of the program. All formulas have the property that the average of a function is approximated by a weighted average of function values, so there is a clear check against inserting spurious factors.

**Differential Equations** The simplest method for numerical solution of differential equations is Euler's method. This simply follows the tangent line at each point for a fixed interval of the independent variable. A complete error analysis of this method shows that the error in the value at any point is proportional to the step size. However, there is also a possibility that the error will depend exponentially on the difference between the initial and final points.

A fundamental higher order method is the fourth order Runge-Kutta method. For equations of the form $dy/dt = f(t)$, this reduces to Simpson's rule, but the exact form of the error term is very complicated in general. Typically, this method is used with refinements of the step size until the difference in value produced by refinement is small enough that one feels that the more accurate value is trustworthy. Since the method is fourth-order, halving the step size should cut the error by a factor of 16. Unfortunately, the exponential dependence on the difference between the initial and final points remains for all methods.

Predictor-Corrector methods use the computed values of $dy/dt$ as the data to construct interpolation polynomials which are integrated from the last tabulated value of the solution to the current value. The result of this integration can be computed once-and-for-all as long as the step size is fixed, leading to simple formulas. When a new value of $y$ is predicted, the formula for $dy/dt$ is modified, and the integral is recomputed. This gives two estimates for each point and the difference between them can be used to estimate the error. This works because the error estimate depends only on a fixed high-order derivative of $y$ with respect to $t$ and not on a mix of partial derivatives of the right side of the equation, as appears in the Runge-Kutta formula. Of course, this is only truncation error. The errors in the previously computed values of $dy/dt$ give an error in the main term of the interpolation formula, and contribute to the compounding effect giving the exponential dependence on the interval of values of the independent variable.

**Exercise 10** A frequent example in the text is

$$\frac{dy}{dt} = \frac{y}{t} - \left(\frac{y}{t}\right)^2$$

with $y(1) = 1$ on $1 \leq t \leq 2$. The exact solution is $y = t/(1 + \ln t)$. Start from $h = .1$, compute three RK4 steps and then shift to a predictor-corrector method. Use the difference between predictor and corrector at $t = 2$ as a guide to the global error in the method. Compare with the error to the true solution. Repeat with $h = 0.05$. What is the effect on the error? What step size do you expect to need to get an accuracy of $10^{-8}$?