CPSC 303: DETAILED LIST OF SKLLS FOR FINAL EXAM, 2024 (IN PROGRESS)

JOEL FRIEDMAN

Contents

Disclaimer: The material may sketchy and/or contain errors, which I will elaborate upon and/or correct in class. For those not in CPSC 303: use this material at your own risk...

The following is a list of skills you should have for the final exam. **It is currently a work in progress: more material will be added.**

NOTE:

(1) skills listed in green will NOT be examinable this year (2024).
(2) skills listed in red represent modifications of skills added or edited the week of April 15-19.

(1) **Intro to ODE's, Numerical Methods for ODE's:**
  (a) **Group Homework 1, Problem 2**: Know how to build derivative approximation schemes using linear algebra and Taylor's theorem. (Later: understand that transposes of Vandermonde matrices show up here.)
  (b) **Group Homework 1, Problems 3,4**: Solve separable ODE's (e.g., $y' = h(t)g(y)$), including those with initial values. Know that some ODE's, such as $y' = y^2$ (under certain initial conditions), should have solutions $y(t)$ that tend to infinity as t approaches some finite time. Know that if $y' = f(y)$ and $z' = g(z)$ have the same initial condition (i.e., $y(t_0) = y_0$ and $z(t_0) = y_0$), and if $f(y) \le g(y)$ for y near y_0, then $z(t) \ge y(t)$ for all t near t_0 with $t > t_0$.
  (c) Know that the equation $\mathbf{y}' = \mathbf{f}(t, \mathbf{y})$ with initial condition $\mathbf{y}(t_0) = \mathbf{y}_0$ has a solution for t near t_0 if $\mathbf{f}$ is continuous; moreover, this local solution is unique if $\mathbf{f}$ is differentiable in $\mathbf{y}$ near $\mathbf{y}_0$.

Research supported in part by an NSERC grant.

(d) Know that some ODE's don't have unique solutions (not even locally), such as $y' = |y|^{1/2}$.
(e) Know all solutions to $y' = |y|^{1/2}$.
(f) Know how to convert the ODE $z'' = g(t, z, z')$ into a vector ODE $\mathbf{y}' = \mathbf{f}(t, \mathbf{y})$ (namely by setting $\mathbf{y} = (z', z)$).
(g) Know that Newton's law $\mathbf{F}_{ij} = -\mathbf{F}_{ji}$ implies the conservation of momentum $\sum_i m_i \mathbf{v}_i = \sum_i m_i \dot{\mathbf{x}}_i$.
(h) Know how to solve an equation with constant coefficients, $(d/dt - r_1)(d/dt - r_2)y = 0$ as $y(t) = C_1 e^{r_1 t} + C_2 e^{r_2 t}$ if $r_1 \neq r_2$, or $y(t) = C_1 e^{r_1 t} + C_2 t e^{r_1 t}$ if $r_1 = r_2$.
(i) Know that the exact solution to $\mathbf{y}' = A\mathbf{y}$ and $\mathbf{y}(t_0) = \mathbf{y}_0$ is $\mathbf{y}(t) = e^{A(t-t_0)}\mathbf{y}_0$, where $e^{A(t-t_0)}$ is obtained using the power series $e^x = 1 + x + x^2/2 + \cdots$, with $A(t - t_0)$ in place of x.
(j) **Group Homework 2, Problem 2**: know that MATLAB has a function that computes e^A for a matrix, A, and that e^A is given by $\sum_{i=0}^{\infty} A^i/i!$.
(k) **Group Homework 2, Problem 3**: know that under Euler's method, $y' = |y|^{1/2}$ will numerically report very different values for $y(2)$ given $y(t_0) = 0$ as opposed to $y(t_0) = 10^{-20}$. (Due to the non-uniqueness of the solution near $t = 0$.)
(l) **Group Homework 2, Problem 4**: know that two time translations of a function yield another time translation, and two time reversals of a function yield a time translation; know how certain types of ODE's behave under time translation and reversal.
(m) **Group Homework 3, Problems 3 and 4**: In exact arithmetic, Euler's method for $y' = |y|^{1/2}$ with $y_0 = -h^2$ takes you to $y_1 = y_2 = \cdots = 0$, but this may not happen in double precision when h is not a power of $(1/2)$. Similarly for y_0 values that in exact arithmetic eventually take you to $y_i = y_{i+1} = \cdots = 0$.
(n) Know that the infinite sum $I + A + A^2/2 + \ldots$ converges to e^A of matrices converges in the sense of norms. Know that $\|A\|_\infty$ equal the largest sum of the absolute values of elements of a row (e.g., for $A = [a, b; c, d]$, $\|A\|_\infty$ equals $\max(|a| + |b|, |c| + |d|)$ (see 01_26 notes, and the rigorous proof in Homework 6, Problem 3).
(o) **Homework 5, Problems 2,3**: If $U' = u$, then he central force problem $m\ddot{\mathbf{x}} = -mu(\|x\|)\mathbf{x}/\|\mathbf{x}\|$ has energy $(1/2)m\|\dot{\mathbf{x}}\|^2 + mU(\|\mathbf{x}\|)$ (which is independent of time). This can be used to test the numerial solution. For Newton's law $u(r) = 1/r^2$, $U(r) = -1/r$, the (Explicit) Trapezoidal method does much better, for a given step size, than Euler's method.

(2) **Recurrence Relations (and the ODE Analog), Normal and Subnormal Numbers (in Double Precision):**
(a) Know how to solve to solve recurrence relations $(\sigma - r_1)(\sigma - r_2)x_n = 0$, e.g., the Fibonacci recurrence relation $(\sigma^2 - \sigma - 1)x_n = 0$. Know that when writing the above three-term recurrence as $\mathbf{y}_{n+1} = A\mathbf{y}_n$ with $\mathbf{y}_n = (x_{n+1}, x_n)$, then A will have eigenvectors $(r, 1)$ where $r = r_1, r_2$.
(b) **Group Homework 4, Problem 2**: The explicity trapezoidal method does better than Euler's method at solving $y' = Ay$ and $y(t_0) = y_0$;

in the case there, to report $y(1)$ given $y' = y$ and $y(0) = 1$ (so $y(1) = e$), Euler's method reports $(1 + 1/N)^N$ and trapezoidal reports $(1 + 1/N + 1/(2N^2))^N$, and one can demonstrate that the latter is a better approximation of e.

(c) You should know how to solve recurrence relations $(\sigma - r_1)(\sigma - r_2)x_n = 0$, and understand the analogy with solving $(d/dt - r_1)(d/dt - r_2)y = 0$ from ODE's above.

(d) **Group Homework 4, Problems 3,4, Group Homework 5, Problems 4**: You should know how to solve homogeneous ODE's with constant coefficients, and homogeneous linear recurrences with constant coefficients; how to solve inhomogeneous equations of the form $(d/dt - r_1)(d/dt - r_2)y = \mathrm{poly}(t)$ (namely you solve the homogenous equation and "guess" $y(t)$ is a polyonmial of the same degree[1]; similarly how to solve $(\sigma - r_1)(\sigma - r_2)x_n = \mathrm{poly}(n)$[2]. Also **Group Homework 5, Problem 5**: know what happens when $r_1 = r_2$, and that this is the same as the limit when r_1 is fixed and $r_2 \to r_1$.

(e) **Group Homework 4, Problem 5**: You should know that recurrence equations with a solution x_n which tends to 0 in exact arithmetic may cycle due to finite precision; the case at hand is $x_{n+2} = (3/2)x_{n+1} - (1/2)x_n$, which cycles at integer multiples of 2^{-1074}.

(f) You should know how normal and subnormal numbers are stored; in particular, you should know that 2^{-1074} is the smallest positive subnormal number, and you should understand the lack of precision when working with subnormal numbers. You should know what are the smallest and largest normal numbers in double precision (2^{-1022} and $(2 - 2^{-52})2^{1023}$, and that a larger number is declared to be `Inf`.

(3) **Interpolation (including Condition Number)**

(a) **Group Homework 6, Problems 2,5**: Know what happens in monomial interpolation on $(x_i, f(x_i))$ for $i = 0, 1, 2$, when x_0, x_1, x_2 tend to each other; know how the relationship with derivative approximation schemes (e.g., [A&G], Chapter 14). Know that the condition number is order $1/\epsilon^2$ when the differences between the x_i are proportional to ϵ. **Group Homework 7, Problem 6**: know that this condition number can change when passing to an equivalent system.

(b) **Group Homeowrk 7, Problems 2,3**: Know that if p is linear function with $p(2), p(2 + \epsilon)$ fixed, or a quadratic function with with $p(2), p(2 + \epsilon), p(2 + 2\epsilon)$ fixed, then for any fixed c, and $\epsilon \to 0$, then $p(2 + c\epsilon)$ is independent of ϵ (and know how to prove this); know that as $\epsilon \to 0$, Lagrange interpolation is far more accurate than monomial interpolation.

(c) **Group Homework 6, Problems 3,4**: Know how to compute $\|A\|_\infty$ of a matrix, and how to find $\mathbf{x}$ (with entries ± 1) such that $\|A\mathbf{x}\|_\infty = \|A\|_\infty \|\mathbf{x}\|_\infty$.

---

[1] This only works when r_1, r_2 are different from 0, but the examples this year avoided this case. If one of r_1, r_2 is 0, you have to guess a polynomial of one degree higher, and if $r_0 = r_2 = 0$, you have to guess a polynomial of two degrees higher.

[2] This only works when r_1, r_2 are different from 1, but the examples this year avoided this case. If one of r_1, r_2 is 1, you have to guess a polynomial of one degree higher, and if $r_0 = r_2 = 1$, you have to guess a polynomial of two degrees higher.

(d) **Group Homework 7, Problem 4,5**: For any A, know how to find $\mathbf{x}_{\text{approx}}, \mathbf{x}_{\text{true}}$, or equivalently $\mathbf{b}_{\text{approx}} = A\mathbf{x}_{\text{approx}}$ and $\mathbf{b}_{\text{true}} = A\mathbf{x}_{\text{true}}$, so that the relative error of $\mathbf{x}_{\text{approx}}, \mathbf{x}_{\text{true}}$ is $\kappa_\infty(A)$ (i.e., the ∞-condition number, i.e. $\|A\|_\infty \|A^{-1}\|_\infty$) time that of $\mathbf{b}_{\text{approx}}, \mathbf{b}_{\text{true}}$.

(e) **Group Homework 8, Problem 2**: Know how the formula for divided differences $p[x_0, x_1, x_2]$ when p is a quadratic polynomial.

(f) Know how to compute $f[x_0, \ldots, x_n]$ in quadratic time in n; know the "generalized mean-value theorem," $f[x_0, \ldots, x_n] = f^{(n)}(\xi)/n!$ for some ξ contained in any interval containing $x_0, \ldots, x_n$.

(g) Know that if f is interpolated at $x_0, \ldots, x_n$ by a polynomial of degree $p(x)$ (of degree at most n), then (the "error in polynomial interpolation theorem" states that)

$$f(x) - p_n(x) = \frac{f^{(n+1)}(\xi)}{(n+1)!} \prod_{i=0}^{n} (x - x_i)$$

where ξ lies in any interval containing $x_0, \ldots, x_n$ and x (see class 03/11 or [A&G] bottom page 314); know how Taylor's theorem results from this. (Class 03/11)

(h) Know that Lagrange interpolation yields the formula

$$f[x_0, \ldots, x_n] = \sum_{i=0}^{n} f(x_i) \prod_{j \neq i} \frac{1}{x_j - x_i}.$$

(i) Know the that (see the table on page 312 of [A&G]):
   (i) Monomial interpolation takes $O(n^3)$ flops to construct, and $O(n)$ evaluation cost (per evaluation);
   (ii) Lagrange interpolation takes $O(n^2)$ flops to construct, and $O(n)$ evaluation cost (per evaluation);
   (iii) Newton interpolation takes $O(n^2)$ flops to construct, and $O(n)$ evaluation cost (per evaluation).

(4) **Splines** These refer to functions whose values at $A = x_0 < x_1 < \ldots < x_n = B$ are given, often as $f(x_0), f(x_1), \ldots, f(x_n)$.

(a) Know that minimizing $\text{Energy}_1(u) = \int (u'(x))^2\, dx$ (or $\text{Length}(u) = \int \sqrt{1 + (u'(x))^2}\, dx$ gives rise to a piecewise linear function (subject to $u(x_i)$ given for $i = 0, 1, \ldots, n$). Hence the energy is not miniminized at a function in $C^1[A, B]$.

(b) By contrast, minimizing $\text{Energy}_2(u) = \int (u''(x))^2\, dx$ gives a piecewise cubic function that is in $C^2[A, B]$ (i.e., it is twice differentiable at the x_i).

(c) **Group Homework 8, Problem 3**: know how minimizing $\text{Energy}_{2,w}(u) = \int w(x)(u''(x))^2\, dx$ gives rise to cubic splines when $w(x)$ is constant, and to a more generally formula when w is not constant.

(d) **Group Homework 9, Problem 2**: know that a cubic spline is not generally three times differentiable across the $x_0, \ldots, x_n$.

(e) The cubic splines over $A = x_0 < x_1 < \ldots < x_n = B$ are described as $v(x)$ which is piecewise cubic polynomials: $s_i(x)$ for $x_i \le x \le x_{i+1}$, for $i = 0, \ldots, n-1$, $s_i(x) = a_i + b_i(x - x_i) + c_i(x - x_i)^2 + d_i(x - x_i)^3$, hence $4n$ parameters. Enforcing $s(x_i) = f(x_i)$, $s(x_{i+1}) = f(x_{i+1})$ and that v', v'' is continuous across $x_1, \ldots, x_{n-1}$ gives $4n-2$ parameters, leaving 2 parameters.

(f) The $\mathrm{Energy}_{2,w}(u) = \int w(x)(u''(x))^2\,dx$ minimizer occurs at the cubic spline v where $v''(x_0) = v''(x_n) = 0$, the *natural spline*; these two conditions determine a unique cubic spline.

(g) For each i, there are formulas for a_i, b_i, d_i in terms of $f(x_i), f_(x_i), c_i, c_{i+1}, h_i$, where $h_i = x_{i+1} - x_i$. One can solve for $\mathbf{c} = (c_1, \ldots, c_{n-1})$ (where $c_0 = c_n = 0$) by a matrix equation. For $h_1 = \cdots = h_{n-1}$, the matrix equation is $(2I + (1/2)N_{\mathrm{rod},n-1})\mathbf{c} = 3\Phi$, where the i-th component of Φ is $f[x_{i-1}, x_i, x_{i+1}]$, and N_{rod} is the tridiagonal matrix with 0's on the diagonal and 1's on all the off-diagonal entries (adjacent to the diagonal).

(h) The matrix $\|N_{\mathrm{rod},n-1}\|_\infty \leq 2$ (equal when $n \geq 3$), and so one can solve for $\mathbf{c}$ as $(3/2)(I - N/4 + (N/4)^2 - \cdots)\Phi$. Since $\|(N/4)^k\|_\infty \leq (1/2)^k$, the above infinite series converges like a geometric series, when measuring with the ∞-norm.

(i) The matrix $N_{\mathrm{rod},n} = A_{P_{n-1}}$, the adjacency matrix of P_{n-1}, the path of length $n-1$; this can be used to compute powers of $N_{\mathrm{rod},n}$.

(j) Another way to understand powers of $N_{\mathrm{rod},n}$ is to write $N_{\mathrm{rod},n} = S_{n,1} + S_{n,-1}$, where $S_{n,1}, S_{n,-1}$ are the "upward shift" and "downward shift" matrices. (These are also the adjacency matrices of directed path graphs.) It is easier to consider powers of the closely related $N_{\mathrm{ring},n} = C_{n,1} + C_{n,-1}$, where $C_{n,1}, C_{n,-1}$, which are "cyclic shifts" up and down by 1; these cyclic shifts are inverses of each other, and give a simple formula for powers of $N_{\mathrm{ring},n}$. Also $N_{\mathrm{ring},n} = A_{C_n}$, the adjacency matrix of C_n, the cycle of length n.

(k) The reason that $N_{\mathrm{rod},n} = S_{n,1} + S_{n,-1}$ doesn't give a simple k-th power formula is that $S_{n,1}, S_{n,-1}$ don't commute. (Although $[S_{n,1}, S_{n,-1}]$ has all but two of its entries 0.)

(l) **Group Homework 10, Problem 2**: even if the $h_1, h_2, \ldots, h_{n-1}$ are not all equal, we still get a system $(2I + (1/2)N)\mathbf{c} = 3\Phi$ as above with $\|N\|_\infty \leq 2$, and so a similar expansion holds.

(5) **The Heat Equation**

(a) Solving the heat equation $u_t = cu_{xx}$ and $u(0,t) = u(1,t) = 0$ by discrete approximation gives $u(x, t+H) = (1-2\rho)u(x,t) + \rho(u(x+h,t) + u(x-h,t)$ where $\rho = cH/h^2$; in other words, with x-step size h, and t-step size H, we get $U(i,j)$ approximates $u(ih, jH)$ and $U(\,\cdot\,, j+1) = (1 - 2\rho + \rho N_{\mathrm{rod},n-2})U(\,\cdot\,, j)$.

(b) If we fix $\rho > 1/2$ and take $h \to 0$ and $\rho = cH/h^2$ (or $H = \rho h^2/c$), this method becomes unstable, roughly when $(|1-2\rho| + 2\rho)^k 2^{-53}$ is close to 1.

(c) The above method becomes a higher order method (of error $O(h^4)$ instead of $O(h^2)$ when $\rho = 1/6$.

(d) The fact that $\rho = 1/6$ yields more accurate results can also be seen by looking at $u(x,t) = sin(\pi x)e^{-\pi^2 ct}$. For the numerical approximation $u(x, t+H) = (1-2\rho)u(x,t) + \rho(u(x+h,t) + u(x-h,t))$ gives $u(x, Hm) \approx sin(\pi x)(1 - 2\rho(1 - \cos(\pi h)))^m$ for fixed ρ, which is within $O(h^4)$ exactly when $\rho = 1/6$ (for other ρ, this is accurate to within $O(h^2)$.

(6) **More on ODE's**

(a) Stiff ODE's: The solution to $y' = ay$ with $a < 0$ is decreasing in time, but with Euler's method we get $y_m = (1 + ah)^m y_0$ which is only decreasing when $-1 < ah < 0$. Hence the solution with Euler's method is "unstable" unless $h > -1/a$.

(b) Stiff systems of ODE's: In solving $\mathbf{y}' = A\mathbf{y}$, the eigenvalues of A can similarly make Euler's method unstable for eigenvalues λ with negative real part unless $|\lambda h + 1| \le 1$.

(c) A second type of stiffness occurs for $\mathbf{y}' = A\mathbf{y}$ with A has complex eigenvalues. This problem shows up in $x'' = -x$, whose solutions are linear combinations of $\sin(t), \cos(t)$, but the eigenvalues of $A = [0, -1; 1, 0]$ are $\lambda = \pm\sqrt{-1}$. For stability one needs $|\sqrt{-1}h + 1| \le 1$ or $\sqrt{1 + h^2} \le 1$, which is impossible. Said otherwise, Euler's method with step size h increases the invariant $(x')^2 + x^2$ by a factor of $\sqrt{1 + h^2}$ per iteration of Euler's method.

(d) There are many higher order Runga-Kutta methods, which generalize Euler's method, including a popular method that is accurate to order 4. However, higher order methods tend to have worse stability properties, in the above sense for Euler's method with stiff ODE's.

(e) A system of the form $\mathbf{z}'' = \mathbf{g}(\mathbf{z})$ is time reversible, so if $\mathbf{z}(t)$ is a solution, then so is $\mathbf{z}(T - t)$ for any t. This gives another way to check numerical solutions. So solve $\mathbf{z}'' = \mathbf{g}(\mathbf{z})$ by writing it as $\mathbf{y}' = f(\mathbf{y})$ with $\mathbf{y} = (\mathbf{z}', \mathbf{z})$: if you solve $\mathbf{y}' = f(\mathbf{y})$ with $\mathbf{y}(0) = (\mathbf{z}'_0, \mathbf{z}_0)$, then solving the same equation with $\mathbf{y}(0) = (-\mathbf{z}'(T), \mathbf{z}(T))$ should give $\mathbf{y}(T) = (-\mathbf{z}'_0, \mathbf{z}_0)$.

Department of Computer Science, University of British Columbia, Vancouver, BC V6T 1Z4, CANADA.

Email address: jf@cs.ubc.ca

URL: http://www.cs.ubc.ca/~jf