CM271.CS371.AMATH341 Final Exam Review

LaTeXer: W. Kong

1 Summary

Name	Formula / Algorithm	Notes
Forward Substitution (FS) [GE]	$x_j = \left(b_j - \sum_{k=1}^{j-1} A_{jk}x_k\right)/A_{jj}$	Used for solving the system $Ax = b$ where A is a matrix. Does this through lower triangular form.
Backward Substitution (BS) [GE]	$x_j = \left(b_j - \sum_{k=j+1}^{n} A_{jk}x_k\right)/A_{jj}$	Does the above except through upper triangular form.
Pivoting	$A_{pj} \leftrightharpoons A_{jj}$	Switches the largest value in a column to the pivoting row before placing in upper/lower triangular form.
LU Decomposition	$A = LU$ so $LUx = b$. Solve $Ly = b$ for y by FS. Solve $Ux = y$ for x by BS.	Solves a system $Ax = b$ by factorization.
Iterative Methods	In the form $x^{(k)} = Tx^{(k-1)} + c$	Be wary of convergence criteria
Jacobi Iterative Method	$x_i^{(k)} = \frac{\left[\sum_{\substack{j\neq i\\ j=1}}^{n}\left(-a_{ij}x_j^{(k-1)}\right)\right]+b_i}{a_{ii}}$	Isolate $x_i's$ in terms of x_j, $i \neq j$, and substitute x_j values from the initial guess.
Gauss-Seidel Iterative Method	$x_i^{(k)} = \frac{-\left[\sum_{j=1}^{i-1} a_{ij}x_j^{(k)}\right]-\left[\sum_{j=i+1}^{n} a_{ij}x_j^{(k-1)}\right]+b_i}{a_{ii}}$	Similar idea as the above except now, you use the $x_i's$ computed thus far in the guesses.
Condition Number	$K(A) = \\|A\\|_p\\|A^{-1}\\|_p$	$K(A) \geq 0$ and defined for all invertible matrices. If $K(A) \approx 1$, then the system is well conditioned. If $K(A) \gg 1$, then it is ill conditioned
Vandermonde Approach	Solve $\begin{bmatrix} x_1^0 & \cdots & x_1^{n-1} \\ \vdots & \ddots & \vdots \\ x_n^0 & \cdots & x_n^{n-1}\end{bmatrix}\begin{bmatrix} a_1 \\ \vdots \\ a_n\end{bmatrix} = \begin{bmatrix} y_1 \\ \vdots \\ y_n\end{bmatrix}$.	Creates a linear system for n points (x_j, y_j) for $j = 1, ..., n$ using any of the above methods. Produces coefficients for a polynomial of degree at most $n-1$: $p(x) = a_1 + a_2x + ... + a_nx^{n-1}$
Langrangian Polynomial Interpolation	$P(x) = \sum_{k=0}^{n} L_{n,k}(x)\cdot y_k$ where $L_{n,k}(x) = \prod_{\substack{i\neq k\\ i=0}}^{n}\frac{(x-x_i)}{(x_k-x_i)}$	Interpolates a degree at most n polynomial for the points (x_i, y_i) for $i = 0, ..., n$ where $f(x_k) = P(x_k) = y_k$.

Name	Formula / Algorithm	Notes
Hermite Interpolation	$H_{2n+1}(x) = \sum_{j=0}^{n} f(x_j)H_{n,j}(x) + \sum_{j=0}^{n} f'(x_j)\hat{H}_{n,j}(x)$ where $H_{n,j}(x) = \left[1 - 2(x-x_j)L'_{n,j}(x_j)\right] L^2_{n,j}(x)$ and $\hat{H}_{n,j}(x) = (x-x_j)L^2_{n,j}(x)$	Does the above but ensures that the first derivative agrees as well. Also f must be $\mathcal{C}^1$ on the domain and will produce a polynomial of at most $2n+1$. Has error" $$\frac{(x-x_0)^2...(x-x_n)^2}{(2n+2)!} f^{(2n+2)}(\xi)$$ for some ξ in the domain.
Piecewise Interpolation	Uses either a 1,2-degree or 3-degee (see below) spline over a uniform partition	For 1 degree, this has error: $E \leq \frac{1}{8}M_2 h^2_{max}$ for M_2 an upper bound on f'' and $h_{max} = \max_i h_i = \max_i (x_{i+1} - x_i)$.
Natural Cubic Spline Piecewise Interpolation	Solve $$\begin{bmatrix} 1 & & & & \\ h_1 & 2(h_1+h_2) & h_2 & & \\ & & \ddots & & \\ & & h_{n-2} & 2(h_{n-2}+h_{n-1}) & h_{n-1} \\ & & & & 1 \end{bmatrix} \vec{c} = \begin{bmatrix} 0 \\ z_2 \\ \vdots \\ z_{n-1} \\ 0 \end{bmatrix}$$ where $z_i = 3\left(\frac{a_{i+1}-a_i}{h_i} - \frac{a_i - a_{i-1}}{h_{i-1}}\right)$, $h_i = x_{i+1} - x_i$, $b_i = \frac{a_{i+1}-a_i}{h_i} - \frac{h_i}{3}(c_{i+1} + 2c_i)$, $d_i = \frac{c_{i+1}-c_i}{3h_i}$ and $a_i = y_i$.	Assumes the following: (1) Derivatives at endpoints are zero (2) Agrees with the function at function values, interpolant points, and first and second derivatives. Each piecewise cubic has the form $$\begin{aligned} p_i(x) &= a_i + b_i(x-x_i) \\ &\quad + c_i(x-x_i)^2 + d_i(x-x_i)^3 \end{aligned}$$
Discrete Least Squares	For a interpolating function $f(\alpha_1, ..., \alpha_m)$. We solve the linear system obtained by the equations $\frac{\partial}{\partial \alpha_j} \sum_{i=1}^{n} (y_i - f(\alpha_1, ..., \alpha_m))^2$ for $j = 1, ..., m$ by isolating the $\alpha_i' s$ on one side.	This is for a set of n points (x_i, y_i).
Bezier Curves	The equation of the curve is $P(t) = \sum_{i=0}^{n} P_i B_{i,n}(t)$ where the $P_i' s$ are k-dimensional control points. Note $B_{m,n}(t) = \binom{n}{m} t^m (1-t)^n$ and $\frac{d}{dt}B_{i,n}(t) = n\left(B_{i-1,n-1}(t) - B_{i,n-1}(t)\right)$.	A closed Bezier curve is when $P_0 = P_n$.
Error of Interpolating Piecewise Polynomial	$$E \leq \frac{h^{n+1}}{(n+1)!} \sup_{c \in [a,b]} \left\| f^{(n+1)}(c) \right\|$$	
Trapezoidal Rule	$\int_a^b f(x)\,dx = \frac{h}{2}\left[f(x_0) - f(x_1)\right] - \frac{h^3}{12} f''(\xi), a = x_0, b = x_1$	Gives exact results for functions with second derivative equal to zero.
Simpson's Rule	$\int_{x_0}^{x_2} f(x)\,dx = \frac{h}{3}\left[f(x_0) + 4f(x_1) + f(x_2)\right] - \frac{h^5}{90} f^{(4)}(\xi_3)$	
Midpoint Rule	This is like approximating with rectangles with midpoint crossing the function at the midpoint of its interval.	

Name	Formula / Algorithm	Notes
Composite Trapezoidal Rule	$$\int_a^b f(x)\,dx = \frac{h}{2}\left[f(a) + 2\sum_{j=1}^{n-1} f(x_j) + f(b)\right] - \frac{(b-a)}{12}h^2 f^{(2)}(\mu), a = x_0, b = x_1$$	Requires $f \in \mathcal{C}^2[a,b]$.
Composite Simpson's Rule	$$\int_a^b f(x)\,dx = \frac{h}{3}\left[f(a) + 2\sum_{j=1}^{(\frac{n}{2})-1} f(x_{2j}) + 4\sum_{j=1}^{\frac{n}{2}} f(x_{2j-1}) + f(b)\right] - \frac{(b-a)}{180}h^4 f^{(4)}(\mu), a = x_0, b = x_1$$	Requires $f \in \mathcal{C}^4[a,b]$ and n is even.
Composite Midpoint Rule	$$\int_a^b f(x)\,dx = 2h\sum_{j=1}^{\frac{n}{2}} f(x_{2j}) - \frac{(b-a)}{6}h^2 f^{(2)}(\mu), a = x_0, b = x_1$$	Requires $f \in \mathcal{C}^2[a,b]$ and n is even.
Gaussian Quadrature	$$\int_a^b f(x)dx \approx \frac{b-a}{d-c}\sum_{i=0}^{n} w_i f\left(\frac{b-a}{d-c}t_i + \frac{ad-bc}{d-c}\right)$$	$n+1$ terms
Legendre Polynomial	$$g_k(x) = \frac{2k-1}{k}xg_{k-1}(x) - \frac{k-1}{k}g_{k-2}(x)$$	$g_0(x) = 1, g_1(x) = x$. Note that the roots of g_n produce the $x_i's$ in the formula $\int_b^a f(x)dx = \sum_{i=1}^{n} w_i f(x_i)$. Use the fact that the approximation is accurate for polynomials of degree $2n+1$ to solve a linear system for the $w_i's$.
Monte-Carlo Integration	$$\int_0^1 f(x)dx \approx \frac{1}{n}\sum_{i=1}^{n} f(x_i)$$	Where the x_i are random points in [0,1] If interval not of length 1 then use $\frac{1}{b-a}\int_a^b f(x)dx \approx \int_0^1 f(x)dx$

Name	**Formula / Algorithm**	**Notes**
Bisection Method	$mid = \frac{a+b}{2}$	If f(a)f(mid)<0 then [a,mid]. f(mid)f(b)<0 then [mid,b]
Secant Method	$x_{k+1} = x_k - f(x_k)\dfrac{x_k - x_{k-1}}{f(x_k) - f(x_{k-1})}$	
Newton's Method	$x_{k+1} = x_k - \dfrac{f(x_k)}{f'(x_k)}$	Requires: initial estimate, continuous function, first derivative
Linear Convergence	$\|x_k - x^*\| \leq r^k \|x_0 - x^*\|$	for some $0 \leq r \leq 1$
Quadratic Convergence	$\exists c > 0, \forall k > k_0, \|x_{k+1} - x^*\| \leq c\|x_k - x^*\|^2$	
Two point difference formula	$f'(x) = \frac{f(x+h)-f(x)}{h} - \frac{h}{2}f''(c)$	$\frac{h}{2}f''(c)$ is the truncation error
Three point difference formula	$f'(x) = \frac{f(x+h)-f(x-h)}{2h} - \frac{h^2}{6}f'''(c)$	$\frac{h^2}{6}f'''(c)$ is the truncation error