Linear Algebra Review

Vector Norms $\|\boldsymbol{x}\|_2 = \sqrt{\sum_{i=1}^n |x_i|^2}$ (Euclidean norm)
$\|\boldsymbol{x}\|_1 = \sum_{i=1}^n |x_i|$ (absolute value sum/taxi norm)
$\|\boldsymbol{x}\|_\infty = \max_{i=1}^n |x_i|$ (largest component by abs. value)

Matrix Norms $\|\mathbf{A}\|_2 = \max_i \sqrt{\mathrm{eig}_i(\mathbf{A}^T\mathbf{A})}$
$\|\mathbf{A}\|_1 = \max_{1\le j\le n} \sum_{i=1}^m |a_{ij}|$ (max abs. value column sum)
$\|\mathbf{A}\|_\infty \max_{1\le i\le m} \sum_{j=1}^n |a_{ij}|$ (max abs. value row sum)
$\|\mathbf{A}\|_F = \sqrt{\sum_{i=1}^m \sum_{j=1}^n |a_{ij}|^2} = \sqrt{\mathrm{tr}(\mathbf{A}^*\mathbf{A})}$. (square root of the sum of the squares of every element in $\mathbf{A}$).

Systems of Linear Equations

Just Some Basic Stuff $\mathbf{XA} = \mathbf{B} \iff \mathbf{A}^T\mathbf{X}^T = \mathbf{B}^T$
$(\mathbf{UL})^{-1} = \mathbf{L}^{-1}\mathbf{U}^{-1}$
$\det \mathbf{L} = \prod_i^n \mathrm{L}_{ii} \quad \det \mathbf{U} = \prod_i^n \mathrm{U}_{ii}$

Forward Substitution $\mathbf{L}\boldsymbol{x} = \boldsymbol{b}$
$\mathbf{L} \in \mathbb{R}^{n\times n}$ lower triangular, non-singular
$x_i = \frac{1}{\mathrm{L}_{ii}}\left(b_i - \sum_{j=1}^{i-1} \mathrm{L}_{ij}\, x_j\right)$ Time cost: $\mathcal{O}(n^2)$

Back Substitution $\mathbf{U}\boldsymbol{x} = \boldsymbol{b}$
$\mathbf{U} \in \mathbb{R}^{n\times n}$ upper triangular, non-singular
$x_i = \frac{1}{\mathrm{U}_{ii}}\left(b_i - \sum_{j=i+1}^{n} \mathrm{U}_{ij}\, x_j\right)$ Time cost: $\mathcal{O}(n^2)$

LU Decomposition without Pivoting $\mathbf{A} \in \mathbf{R}^{n\times n}$ diagonally dominant $\implies \exists\ \mathbf{A} = \mathbf{LU}$
Algorithm: $\mathbf{A}^{(j)} \equiv \mathbf{A}$ on algorithm's jth iteration.
Start with $j = 1$ and $\mathbf{A}^{(1)} = \mathbf{A}$. Perform row subtraction on $A^{(j)}$'s rows $j+1$ to n; record subtraction coefficient in $\mathbf{L}$. Increment $j \to j+1$, repeat for all n rows.
Computational complexity: $\mathcal{O}(n^3)$

LU Decomposition with Partial Pivoting $\mathbf{A} \in \mathbf{R}^{n\times n}$ non-singular $\implies \exists\ \mathbf{PA} = \mathbf{LU}$
Algorithm: $\mathbf{A}^{(j)} \equiv \mathbf{A}$ on algorithm's jth iteration.
Start with $j = 1$ and $\mathbf{A}^{(1)} = \mathbf{A}$. Find largest element A_{ij} in column j from rows j to n. Switch jth row with row containing largest element. Record switch in $\mathbf{P}$. Perform row subtraction on $A^{(j)}$'s rows $j+1$ to n; record subtraction coefficient in $\mathbf{L}$. Increment $j \to j+1$, repeat for all n rows.
Computational complexity: $\mathcal{O}(n^3)$

Applications of LU Decomposition To find inverse of $\mathbf{A} \in \mathbb{R}^{n\times n}$: Solve $\mathbf{AA}^{-1} = \mathbf{I}$
To find $\det \mathbf{A}$: Find $\mathbf{PA} = \mathbf{LU}$, use $\det \mathbf{A} = (-1)^s \cdot \det \mathbf{U}$ where s is number of row switches in permutation matrix $\mathbf{P}$

Cholesky Decomposition $\mathbf{A} \in \mathbb{R}^{n\times n}$, $\mathbf{A}$ symmetric and positive definite.
$\mathbf{A} = \mathbf{LL}^T$ where $\mathbf{L}$ is lower-diagonal.
Algorithm: For $k = 1$ initialize $\mathbf{L}_1 = \sqrt{\mathrm{A}_{11}}$. For $k = 2, \ldots, n$, solve $\mathbf{L}_{k-1}\boldsymbol{l}_k = \boldsymbol{a}_k$ for $\boldsymbol{l}_k$. Solve $\mathrm{L_{kk}} = \sqrt{\mathrm{A}_{kk} - \boldsymbol{l}_k^T\boldsymbol{l}_k}$. Assemble $\mathbf{L}_k = \begin{bmatrix}\mathbf{L}_{k-1} & \mathbf{0}_k\boldsymbol{l}_k^T & \mathrm{L}_{kk}\end{bmatrix} \in \mathbb{R}^{k\times k}, \mathbf{0}_k \in \mathbb{R}^k$. Finish with $\mathbf{L} = \mathbf{L}_n$.
Notation: Matrix $\mathbf{L}_k$: $k\times k$ principle sub-matrix of $\mathbf{L}$. Vectors $\boldsymbol{a}_k, \boldsymbol{l}_k$: first $k-1$ entries in column k of $\mathbf{A}$ and $\mathbf{L}^T$.

Solving Common Linear Systems 1. $\mathbf{AX} = \mathbf{B}$ where $\mathbf{A} \in \mathbb{R}^{n\times n}$ and $\mathbf{B}, \mathbf{X} \in \mathbb{R}^{n\times m}$
Write $\mathbf{B} = (\boldsymbol{b}_1, \ldots, \boldsymbol{b}_m)$, LU decomposition $\mathbf{A} = \mathbf{LU}$. Solve $\mathbf{L}\boldsymbol{y}_i = \boldsymbol{b}_i$ with forward substitution, solve $\mathbf{U}\boldsymbol{x}_i = \boldsymbol{y}_i$ with back substitution, reconstruct $\mathbf{X} = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m)$.

2. $\mathbf{XA} = \mathbf{B}$ where $\mathbf{A} \in \mathbb{R}^{n\times n}$ and $\mathbf{B}, \mathbf{X} \in \mathbb{R}^{m\times n}$
Apply equality $\mathbf{XA} = \mathbf{B} \iff \mathbf{A}^T\mathbf{X}^T = \mathbf{B}^T$. Solve the system $\tilde{\mathbf{A}}\tilde{\mathbf{X}} = \tilde{\mathbf{B}}$ using (1).

3. $\mathbf{AXB} = \mathbf{C}$ where $\mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{X} \in \mathbb{R}^{n\times n}$
Let $\mathbf{Y} = \mathbf{XB}$ and solve $\mathbf{LY} = \mathbf{C}$ for $\mathbf{Y} = (\boldsymbol{y}_1, \ldots, \boldsymbol{y}_n)$ using (1). Solve $\mathbf{XB} = \mathbf{Y}$ for $\mathbf{X} = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n)$ using (2).

4. $\mathbf{AX} = \mathbf{B}$; $\mathbf{A} \in \mathbb{R}^{n\times n}$ symmetric; $\mathbf{B}, \mathbf{X} \in \mathbb{R}^{n\times m}$
Let $\mathbf{B} = (\boldsymbol{b}_1, \ldots, \boldsymbol{b}_m)$, Cholesky decomposition $\mathbf{A} = \mathbf{LL}^T$. Solve $\mathbf{L}\boldsymbol{y}_i = \boldsymbol{b}_i$ with forward substitution, solve $\mathbf{V}^T\boldsymbol{x}_i = \boldsymbol{y}_i$ with back substitution, reconstruct $\mathbf{X} = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m)$.

Non-Linear Equations

Bisection Method *Parameters:* Function f, tolerance ϵ, original interval $[a, b]$, variable interval $[\alpha, \beta]$ with midpoint c.
Algorithm: Start with $\alpha = a, \beta = b$. Calculate midpoint $c = \alpha + \frac{\beta-\alpha}{2}$. If $\operatorname{sign} f(\alpha) = \operatorname{sign} f(c)$, let $\alpha = c$; if $\operatorname{sign} f(\alpha) \neq \operatorname{sign} f(c)$, let $\beta = c$. Repeat until $|\beta - \alpha| \le \epsilon$, then return root $x_0 = \alpha + \frac{\beta-\alpha}{2}$.
Notes: After k iterations, interval width is $\ell = \frac{b-a}{2^k}$. Tolerance ϵ requires $k \ge \log_2 \frac{b-a}{\epsilon}$ iterations.

Fixed Point Iteration *Algorithm:* Solve $f(x) = 0$ for $x = g(x)$. Choose tolerance ϵ and initial x_0, find $x_{j+1} = g(x_j)$, repeat until $|x_{j+1} - x_j| < \epsilon$.
α fixed point of $x_{j+1} = g(x_j)$ if $\alpha = g(\alpha)$
Convergence: Fixed point $\alpha = g(\alpha)$ convergent if $|g'(\alpha)| \le 1$
Fixed point $\alpha = g(\alpha)$ has order of convergence p if $g^{(k)}(\alpha) = 0$ for $k = 1, \ldots, p-1$ and $g^{(p)}(\alpha) \neq 0$

Newton's Method Iteration function: $g(x) = x - \frac{f(x)}{f'(x)}$
Algorithm: Choose tolerance ϵ and initial guess x_0, find $x_{j+1} = x_j - \frac{f(x_j)}{f'(x_j)}$, repeat until $|x_{j+1} - x_j| < \epsilon$.
Convergence: If α is a simple zero (i.e. $f'(\alpha) \neq 0$), convergence is quadratic if $f''(\alpha) \neq 0$ and cubic if $f''(\alpha) = 0$.
Secant Method: $x_{j+1} = x_j - f(x_j)\frac{x_j - x_{j-1}}{f(x_j) - f(x_{j-1})}$

Polynomial Roots Given $p_n(x) = a_n x^n + \cdots + a_1 x + a_0$, construct $n\times n$ matrix

$$\mathbf{A}_p = \begin{bmatrix} 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 1 \\ -\frac{a_0}{a_n} & -\frac{a_1}{a_n} & -\frac{a_2}{a_n} & \cdots & -\frac{a_{n-1}}{a_n} \end{bmatrix}$$

The polynomial's p_n's roots are $\mathbf{A}_p$'s eigenvalues $\mathrm{eig}(\mathbf{A}_p)$

Linear Least Squares

Problem Given vector of data points $\boldsymbol{b} \in \mathbb{R}^m$ and model function $f = f(t, a_1, \ldots, a_m)$, find vector of parameters $\boldsymbol{x} = (a_1, \ldots, a_n) \in \mathbb{R}^n$ minimizing $\|\mathbf{A}\boldsymbol{x} - \boldsymbol{b}\|_2$ where $\mathbf{A} \in \mathbb{R}^{m\times n}$; $\operatorname{rank} \mathbf{A} = n$.
A_{ij} are coefficients of jth parameter a_j at ith data point.

Normal System $\mathbf{A}^T\mathbf{A}\boldsymbol{x} = \mathbf{A}^T\boldsymbol{b}$; solved with Cholesky decomposition.
$\boldsymbol{x} \in \mathbb{R}^n$ is desired vector of parameters
Unstable if $\mathbf{A}$'s columns are nearly linearly dependent.

QR Decomposition $\mathbf{A} \in \mathbb{R}^{m\times n} \implies \mathbf{A} = \mathbf{QR}$. $\mathbf{Q} \in \mathbb{R}^{m\times n}$ has orthogonal columns and $\mathbf{R} \in \mathbb{R}^{n\times n}$ is upper triangular.
Solve $\mathbf{R}\boldsymbol{x} = \mathbf{Q}^T\boldsymbol{b}$ for least squares parameter vector $\boldsymbol{x}$.

Householder Reflection $\mathbf{P} = \mathbf{I} - \frac{2}{\boldsymbol{w}^T \cdot \boldsymbol{w}} \boldsymbol{w}\boldsymbol{w}^T \quad \mathbf{P} = \mathbf{P}^T = \mathbf{P}^{-1}$
For $\mathbf{A} = [\boldsymbol{a}_1, \ldots, \boldsymbol{a}_n] \in \mathbb{R}^{m\times n}$ find $\boldsymbol{w} \in \mathbb{R}^m$ so $\mathbf{P}\mathbf{a}_1 = k\mathbf{e}_1$
$\boldsymbol{w} = \left[a_1 + \operatorname{sgn}\|\boldsymbol{a}\|_2, a_2, \ldots, a_m\right]^T \quad \|\boldsymbol{a}\|_2 = k$
$\mathbf{Q} = (\mathbf{P}_n\mathbf{P}_{n-1}\ldots\mathbf{P}_1)^T = \cdots = \mathbf{P}_1\mathbf{P}_2\ldots\mathbf{P}_n$
Givens Rotation $\mathbf{Q}^T$ Givens rotation such that $\mathbf{Q}^T\mathbf{A} = \mathbf{R}$
$$\begin{bmatrix}\cos\phi & -\sin\phi\\ \sin\phi & \cos\phi\end{bmatrix}^T \begin{bmatrix}\mathrm{A}_{11} & \mathrm{A}_{12}\\ \mathrm{A}_{21} & \mathrm{A}_{22}\end{bmatrix} = \begin{bmatrix}r & \mathrm{R}_{12}\\ 0 & \mathrm{R}_{22}\end{bmatrix}$$
$r = \sqrt{A_{11}^2 + A_{21}^2} \quad \cos\phi = \frac{A_{11}}{r} \quad \sin\phi = \frac{A_{21}}{r}$

Eigenvalues and Eigenvectors

Power Method Used to find largest eigenvalue of $\mathbf{A} \in \mathbb{R}^{n\times n}$
Algorithm: Pick initial vector $\boldsymbol{z}_0 \in \mathbb{R}^n$ and tolerance ϵ. For $k = 0, 1, \ldots$ find $\boldsymbol{y}_{k+1} = \mathbf{A}\boldsymbol{z}_k$ and normalize $\boldsymbol{z}_{k+1} = \frac{\boldsymbol{y}_{k+1}}{\|\boldsymbol{y}_{k+1}\|}$.
Stop when $\|\mathbf{A}\boldsymbol{z}_k - \rho_k\boldsymbol{z}_k\| \le \epsilon$ where $\rho(\boldsymbol{x}, \mathbf{A}) = \frac{\boldsymbol{x}^H\mathbf{A}\boldsymbol{x}}{\boldsymbol{x}^H\boldsymbol{x}}$.
ρ_k is the approximation for $\mathbf{A}$'s largest eigenvalue.

Eigenvalue Reduction *Problem:* For eigenvalue-eigenvector pair $\lambda_i, \boldsymbol{x}_i$ of matrix $\mathbf{A} \in \mathbb{R}^{n\times n}$ where $\|\boldsymbol{x}_i\| = 1$, find $\mathbf{B} \in \mathbb{R}^{n\times n}$, so $\operatorname{eig}(\mathbf{B}) = \operatorname{eig}(\mathbf{A}) \setminus \{\lambda_i\}$ and $\lambda_i \to 0$ i.e. λ_i is replaced by 0.
For symmetric matrices: $\mathbf{B} = \mathbf{A} - \lambda_i\boldsymbol{x}_i\boldsymbol{x}^T$
For non-symmetric matrices: $\mathbf{B} = \mathbf{Q}\mathbf{A}\mathbf{Q}^T$ for orthogonal $\mathbf{Q}$ such that $\mathbf{Q}\boldsymbol{x}_i = k\boldsymbol{e}_i$; $\boldsymbol{e}_i$ unit vector corresponding to λ_i.

Inverse Iteration Used to find smallest eigenvalue of $\mathbf{A} \in \mathbb{R}^{n\times n}$.
Algorithm: Use power method to find largest eigenvalue ψ_{max} of $\mathbf{A}^{-1}$; $\lambda_{min} = \frac{1}{\psi_{max}}$ is smallest eigenvalue of $\mathbf{A}$.

QR Iteration To find eigenvalues of non-symmetric matrix $\mathbf{A} \in \mathbb{R}^{n\times n}$.
Algorithm: Start with $\mathbf{A}_0 = \mathbf{A}$. For $k = 0, 1, \ldots$ find QR decomposition $\mathbf{A}_k = \mathbf{Q}_k\mathbf{R}_k$ and calculate $\mathbf{A}_{k+1} = \mathbf{R}_k\mathbf{Q}_k$.

If $\operatorname{eig}(\mathbf{A}) \in \mathbb{R}$, then for large k, $\mathbf{A}_k$ becomes upper-triangular and $\mathbf{A}_k$'s diagonal entries approach $\mathbf{A}$'s eigenvalues.

If $\mathbf{A}$ has $m < n$ complex eigenvalues, then for large k, $\mathbf{A}_k$ becomes quasi-upper triangular, $\mathbf{A}_k$'s diagonal entries approach $\mathbf{A}$'s real eigenvalues, and a $m \times m$ sub-matrix whose eigenvalues are $\mathbf{A}$'s complex eigenvalues remains on $\mathbf{A}_k$'s diagonal.

Symmetric and Tridiagonal Matrices For all symmetric $\mathbf{A} \in \mathbb{R}^{n\times n}$ there exists permutation matrix $\mathbf{P} \in \mathbb{R}^{n\times n}$ such that $\mathbf{T} = \mathbf{P}\mathbf{A}\mathbf{P}^T$ is tridiagonal.
$\mathbf{T}$ irreducible $\implies$ $\mathbf{T}$ has no zeros on upper tridiagonal.
$$\mathbf{T} = \begin{bmatrix} a_1 & b_1 & & & & & \\ b_1 & a_1 & b_2 & & & & \\ & \ddots & \ddots & \ddots & & & \\ & & b_{n-2} & a_{n-1} & b_{n-1} & b_{n-1} & a_n \end{bmatrix}$$
$\mathbf{T}_i \equiv \mathbf{T}$'s $i \times i$ principle sub-matrix.

Sturm Sequence $f_i(\lambda) = \det(\mathbf{T} - \lambda\mathbf{I}) \equiv \mathbf{T}_i$'s characteristic polynomial.
Sturm sequence: $f_{i+1}(\lambda) = (a_{i+1} - \lambda)f_i(\lambda) - b_i^2 f_{i-1}(\lambda)$,
$f_0(\lambda) = 1$ and $f_1(\lambda) = a_1 - \lambda \quad i = 0, \ldots, n$
For λ_0, $s(\lambda_0)$ is the number of sign agreements between successive terms in the sequence $f_i(\lambda_0), i = 0, \ldots, n$.
Adjacent terms with same sign and *interior* zeroes of $f_i(\lambda_0)$ count as sign agreements.
The number of sign agreements $s(\lambda_0)$ is the number of $\mathbf{T}$'s eigenvalues that are strictly larger than λ_0.

Polynomial Interpolation

Given n points $(x_i, y_i), i = 0, 1, \ldots, n$, all x_i unique, find a polynomial of degree $\le n$ such that $p(x_i) = y_i$ for all i.

Classic Form Interpolation polynomial: $p_n(x) = a_nx^n + \cdots + a_1x + a_0$ $a_nx_0^n + \cdots + a_1x_0 + a_0 = y_0$
$a_nx_1^n + \cdots + a_1x_1 + a_0 = y_1 \qquad \implies \mathbf{V}\boldsymbol{a} = \boldsymbol{y}$
$\vdots$
$a_nx_n^n + \cdots + a_1x_n + a_0 = y_n$
Find parameter vector $\boldsymbol{a} \in \mathbb{R}^n$ by solving $\mathbf{V}\boldsymbol{a} = \boldsymbol{y}$.

Lagrange Polynomial Interpolation Lagrange polynomials: $l_i(x) = \prod_{j\ne i}^n \frac{x - x_j}{x_i - x_j}, \quad l_i(x_j) = \delta_{i,j}$
Interpolation polynomial: $p_n(x) = \sum_{i=0}^n y_il_i(x)$
If $\omega(x) = (x - x_0)\cdots(x - x_n)$ then $l_i(x) = \frac{\omega(x)}{(x - x_i)\omega'(x_i)}$

Newton Polynomial Interpolation Divided difference for function f and points (x_i, y_i) is the leading coefficient of polynomial p_n interpolating f at (x_i, y_i).
$[x_0, \ldots, x_k]f = \frac{[x_1, \ldots, x_k]f - [x_0, \ldots, x_{k-1}]f}{x_k - x_0} \qquad [x_k]f = f(x_k)$
$\lim_{x_1\to x_0}[x_0, x_1]f = \lim_{x_1\to x_0} \frac{f(x_1) - f(x_0)}{x_1 - x_0} = f'(x_0)$
If $x_0 = \cdots = x_n$, then $[x_0, \ldots, x_n]f = \frac{f^{(n)}(x_0)}{n!}$
Interpolation polynomial: $p_n(x) = [x_0]f + (x - x_0)[x_0, x_1]f + \cdots + (x - x_0)\cdots(x - x_{n-1})[x_0, \ldots, x_n]f$

Error Estimation for Lagrange and Newton When interpolating $f(x), f'(x), \ldots$ on the interval $I = [a, b]$ at m points $x_1, \ldots, x_m$ with nth degree polynomial p_n.
$f(x) - p_n(x) = \frac{f^{(n+1)}(\xi)}{(n+1)!}\omega(x), \qquad \xi \in [a, b]$
$\omega(x) = (x - x_1)^{k_1}(x - x_2)^{k_2}\cdots(x - x_n)^{k_n}$
$k_i = 1$ for $f(x_i)$, $k_i = 2$ for $f(x_i)$ and $f'(x_i)$, etc...
$\max_{x\in[a,b]} |f(x) - p_n(x)| \le \frac{1}{(n+1)!} \max_{\xi\in[a,b]} \left|f^{(n+1)}(\xi)\right| \max_{x\in[a,b]} |\omega(x)|$

Differential Equations

To convert $a_ny^{(n)} + \cdots + a_1y' + a_0y = 0$ to a linear system:
Let $\tilde{y}_1 = y,\ \tilde{y}_2 = y',\ \ldots,\ \tilde{y}_n = y^{(n-1)}$
$\mathbf{Y} = [\tilde{y}_1, \ldots, \tilde{y}_n] \quad \mathbf{Y}' = \boldsymbol{F}(x, \mathbf{Y})$
Explicit Euler: $y_{n+1} = y_n + hf(x_n, y_n)$
Implicit Euler: $y_{n+1} = y_n + hf(x_{n+1}, y_{n+1})$
Trapezoid: $y_{n+1} = y_n + \frac{h}{2}\left[f(x_n, y_n) + f(x_{n+1}, y_{n+1})\right]$

Runge-Kutta RK2 $k_1 = hf(x_n, y_n)$
$k_2 = hf(x_n + \frac{1}{2}h, y_n + \frac{1}{2}k_1)$
$y_{n+1} = y_n + k_2 + \mathcal{O}(h^3)$

Runge-Kutta RK4 $k_1 = hf(x_n, y_n)$
$k_2 = hf(x_n + \frac{1}{2}h, y_n + \frac{1}{2}k_1)$
$k_3 = hf(x_n + \frac{1}{2}h, y_n + \frac{1}{2}k_2)$
$k_4 = hf(x_n + h, y_n + k_3)$
$y_{n+1} = y_n + \frac{1}{6}k_1 + \frac{1}{3}k_2 + \frac{1}{3}k_3 + \frac{1}{6}k_4 + \mathcal{O}(h^5)$