Gerŝgorin Circle Theorem [1] [2]

$$R^i = \left\{ z \in \mathbb{C} \;\middle|\; |z - a_{ii}| \leqslant \sum_{j=1, j\neq i}^{n} |a_{ij}| \right\}$$

Finding Eigenvalues Solve $\det(A - \lambda I) = 0$

Finding Eigenvectors Solve $(A - \lambda_i I)x = 0$. The eigenvectors are linearly independent if $\det(A) \neq 0$

Orthogonal Vectors $(\mathbf{v}^{(i)})^\top \mathbf{v}^{(j)} = 0,\ \forall\ i \neq j$

Orthonormal Vectors [3] $(\mathbf{v}^{(i)})^\top \mathbf{v}^{(i)} = 1,\ \forall\ i = 1, \dots, n$ and above

Orthogonal Matrices An $n \times n$ matrix whose columns form an orthonormal set in $\mathbb{R}^n$ is orthogonal.

Invertible/Orthogonal Matrix Properties

i Orthogonal Q is invertible with $Q^{-1} = Q^\top$

ii Invertible Q is orthogonal if $Q^\top = Q^{-1}$

iii $\forall \mathbf{x} \in \mathbb{R}^n, \|Q\mathbf{x}\|_2 = \|\mathbf{x}\|_2$

iv $Q^{-1}Q = Q^\top Q = I$

Positive-Definiteness A symmetric matrix is positive-definite if and only if all its eigenvalues are positive.

Similar Matrices $A \sim D \rightarrow \exists\, S \mid A = S^{-1}DS$

i $A \sim D$ with $A = S^{-1}DS$, where the columns of S consist of the eigenvectors, and the ith diagonal element of D is the eigenvalue of A that corresponds to the eigenvector in the ith column of S.

ii Similar matrices have the same eigenvalues.

Power Method Rate of convergence is $O(|\lambda_2/\lambda_1|^m)$. It is not known whether or not A has a single dominant eigenvalue nor how $\mathbf{x}^{(0)}$ should be chosen. $\mathbf{x}_u^{(k)}$ is not normalized.

$$\mathbf{x}^{(0)} \neq 0 \text{ given}$$

$$\left\{ \mathbf{x}_u^{(k)} = A\mathbf{x}^{(k-1)} \right.$$

The final values must be normalized.

$$\begin{aligned} \mathbf{x}^{(k)} &= \mathbf{x}_u^{(k)} / \mathbf{x}_{u,p_k}^{(k)} \\ \lambda^{(k)} &= \mathbf{x}_{p_k}^{(k)} / \mathbf{x}_{p_k}^{(k-1)} \\ \mathbf{v}^{(k)} &= \mathbf{x}^{(k)} / \mathbf{x}_{p_k}^{(k)} \end{aligned}$$

Symmetric Power Method Used when A is symmetric. Rate of convergence is $O(|\lambda_2/\lambda_1|^{2m})$. $\mathbf{x}_u^{(k)}$ is not normalized.

$$\mathbf{x}^{(0)} \neq 0 \text{ given}$$

$$\left\{ \mathbf{x}_u^{(k)} = A\mathbf{x}^{(k-1)} \right.$$

The final values must be normalized.

$$\begin{aligned} \mathbf{x}^{(k)} &= \mathbf{x}_u^{(k)} / \|\mathbf{x}_u^{(k)}\|_2 \\ \lambda^{(k)} &= \|\mathbf{x}^{(k)}\|_2 / \|\mathbf{x}^{(k-1)}\|_2 \\ \mathbf{v}^{(k)} &= \mathbf{x}^{(k)} / \|\mathbf{x}^{(k)}\|_2 \end{aligned}$$

Wielandt Deflation Delete the ith row and column of B.

$$B = A - \lambda_1 \mathbf{v}^{(1)} \mathbf{x}^\top \text{ where } \mathbf{x} = \frac{1}{\lambda_1 v_i^{(1)}} (a_{i1}, a_{i2}, \dots a_{in})$$

[1] a_{ij} is the i-th row and j-th column of the matrix A.

[2] The eigenvalues fall in the union of the circles. Remember $\rho(A) = \max|\lambda_i|$

[3] Simply normalize the orthogonal set to become orthonormal

Inverse Power Method Gives faster convergence. Used to determine the eigenvalue that is closest to q. A combined method to find all eigenvalues of a matrix is do deflate the original matrix, find the deflated matrix's eigenvalue through General Power Method, then use that eigenvalue as an initial guess to the Inverse Power Method to ensure we have found an eigenvalue of the original matrix, not the deflated matrix.

$$\mathbf{x}^{(0)} \neq 0 \text{ given, set } \mathbf{y}^{(0)} = \mathbf{x}^{(0)}$$

$$\left\{ \mathbf{x}_u^{(k)} = (A - qI)^{-1} \mathbf{x}^{(k-1)} \right.$$

The final values must be normalized.

$$\begin{aligned} \mathbf{x}^{(k)} &= \mathbf{x}_u^{(k)} / \mathbf{x}_{u,p_k}^{(k)} \\ \mathbf{y}^{(k)} &= (A - qI)^{-1} \mathbf{x}^{(k)} \\ \lambda^{(k+1)} &= 1/\mathbf{y}_{p_k}^{(k)} + q \end{aligned}$$

Householder Transformation Reduces a symmetric matrix to a similar tridiagonal matrix. Repeat for $k = 1, \dots, n-2$.

$$\left\{ \begin{aligned} q &= \sum_{j=k+1}^{n} (a_{j,k})^2 \\ \alpha &= -\operatorname{sign}(a_{k+1,k})\sqrt{q} \\ r &= \sqrt{\frac{1}{2}\alpha^2 - \frac{1}{2}\alpha a_{k+1,k}^{(k)}} \\ w_j^{(k)} &= \frac{a_{j,k}^{(k)}}{2r} \text{ for each } j = k+2, \dots, n \\ P^{(k)} &= I - 2w^{(k)} * (w^{(k)})^\top \\ A^{(k+1)} &= P^{(k)} A^{(k)} P^{(k)} \end{aligned} \right.$$

Rotation Matrices An identity matrix which differs in four elements, for some θ and some $i \neq j$.

$$p_{ii} = p_{jj} = \cos\theta \text{ and } p_{ij} = -p_{ji} = \sin\theta$$

QR Algorithm Finds all the eigenvalues of a symmetric, tridiagonal matrix (use Householder's if not tridiagonal). Do the following where $k = 1, \dots n-1$ for each iteration i needed. The diagonal of A contains the eigenvalues, which correspond to the eigenvectors in the respective column of Q.

$$\left\{ \begin{aligned} c_{k+1} &= \frac{a_{k,k}}{\sqrt{a_{k,k}^2 + a_{k+1,k}^2}} \\ s_{k+1} &= \frac{a_{k+1,k}}{\sqrt{a_{k,k}^2 + a_{k+1,k}^2}} \\ P_{k+1} &= \left(\begin{array}{c|cc|c} I_{k-1} & \multicolumn{2}{c|}{0} & 0 \\ \hline 0 & c_{k+1} & s_{k+1} & 0 \\ & -s_{k+1} & c_{k+1} & \\ \hline 0 & \multicolumn{2}{c|}{0} & I_{n-k-1} \end{array} \right) \end{aligned} \right.$$

$$\begin{aligned} Q^{(i)} &= P_2^\top \cdot \dots \cdot P_{n-1}^\top \\ A^{(i+1)} &= A_{n-1}^{(i)} \cdot Q^{(i)} \end{aligned}$$

Fixed Points $\mathbf{G} : D \subset \mathbb{R}^n \mapsto \mathbb{R}^n$ has fixed point $\mathbf{p} \in D$ if $\mathbf{G}(\mathbf{p}) = \mathbf{p}$. Given $f_1(x_1, \dots, x_n), \dots, f_n(x_1, \dots, x_n)$, solve the ith equation for x_i and set the solved x_i equal to $g_i(x_1, \dots, x_n)$. Apply Jacobi or Gauss-Seidel on each g_i to find the fixed point.

Fixed Point Theorem Assure that $\left|\frac{\delta g_i(\mathbf{x})}{\delta x_j}\right| \leqslant \frac{K}{n}$ for $j = 1, \dots, n$, $K < 1$. Then we can be sure that $\mathbf{G}(\mathbf{x})$ converges in some $x \in D$ such that $a_i \leqslant g_i(x_i, \dots, x_n) \leqslant b_i$ for some $a_i < b_i < \infty$. This works because $\mathbf{F}(\mathbf{p}) = \mathbf{p} - \mathbf{G}(\mathbf{p}) = 0 \Rightarrow \mathbf{G}(\mathbf{p}) = \mathbf{p}$.

Newton's Method If we can't solve the ith equation for x_i in the Fixed Point Method, we use Newton's Method, which exhibits quadratic convergence. Newton's method requires continuous g derivatives and that $A(\mathbf{x}) = \mathbf{J}(\mathbf{x})$ is nonsingular around the radius of the solution. Given a guess $\mathbf{x}^{(0)}$, we do the following.

$$\begin{aligned}\mathbf{x}^{(k)} = \mathbf{G}(\mathbf{x}^{(k-1)}) &= \mathbf{x}^{(k-1)} - A(\mathbf{x}^{(k-1)})^{-1}\mathbf{F}(\mathbf{x}^{(k-1)}) \\ &= \mathbf{x}^{(k-1)} - \mathbf{J}(\mathbf{x}^{(k-1)})^{-1}\mathbf{F}(\mathbf{x}^{(k-1)})\end{aligned}$$

A weakness in Newton's is the requirement to compute and invert $\mathbf{J}(\mathbf{x})$, avoided by finding a $\mathbf{y}$ such that

$$\mathbf{J}(\mathbf{x}^{(k-1)})\mathbf{y} = -\mathbf{F}(\mathbf{x}^{(k-1)})$$

then $\mathbf{x}^{(k)} = \mathbf{x}^{(k-1)} + \mathbf{y}$. Newton's Method requires $n^2 + n$ functional evaluations, n^2 for the Jacobian matrix, n for the evaluation of $\mathbf{F}$, and $O(n^3)$ arithmetic operations to solve the linear system.

Broyden Quasi-Newton. Requires only n scalar functional evaluations per iteration and reduces the calculations to $O(n^2)$. Quadratic convergence is lost and becomes superlinear. Broyden's is not self-correcting (as Newton's is) to roundoff errors. Given a guess $\mathbf{x}^{(0)}$, we first compute the first of A_k, which will be used in place of $\mathbf{J}(\mathbf{x}^{(k)})$ to determine the next $\mathbf{x}$. We must also compute $\mathbf{x}^{(1)}$ using one iteration of Newton's.

$$\begin{aligned}A_0 &= \mathbf{J}(\mathbf{x}^{(0)}) \\ A_1 &= A_0 + \frac{[\mathbf{F}(\mathbf{x}^{(1)}) - \mathbf{F}(\mathbf{x}^{(0)}) - \mathbf{J}(\mathbf{x}^{(0)})(\mathbf{x}^{(1)} - \mathbf{x}^{(0)})](\mathbf{x}^{(1)} - \mathbf{x}^{(0)})^\top}{\|\mathbf{x}^{(1)} - \mathbf{x}^{(0)}\|_2^2} \\ &= A_0 + \frac{\mathbf{y}_1 - A_0\mathbf{s}_1}{\|\mathbf{s}_1\|_2^2}\mathbf{s}_1^\top\end{aligned}$$

$$\begin{cases}\mathbf{y}_k = \mathbf{F}(\mathbf{x}^{(k)}) - \mathbf{F}(\mathbf{x}^{(k-1)}) \\ \mathbf{s}_k = (\mathbf{x}^{(k)} - \mathbf{x}^{(k-1)}) \\ A_k = A_{k-1} + \dfrac{\mathbf{y}_k - A_{k-1}\mathbf{s}_k}{\|\mathbf{s}_k\|_2^2}\mathbf{s}_k^\top \\ \mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} - A_k^{-1}\mathbf{F}(\mathbf{x}^{(k)})\end{cases}$$

Sherman-Morrison Formula Broyden's formula still requires a matrix inversion to be calculated, so this formula permits A_i^{-1} to be computed from A_{i-1}^{-1}, as long as A is nonsingular. This computation of A requires only $O(n^2)$ calculations, as we bypass calculating A_i and the necessity of solving the linear system. Using the property

$$\left(A + \mathbf{x}\mathbf{y}^\top\right)^{-1} = A^{-1} - \frac{A^{-1}\mathbf{x}\mathbf{y}^\top A^{-1}}{1 + \mathbf{y}^\top A^{-1}\mathbf{x}}$$

We obtain

$$A_i^{-1} = A_{i-1}^{-1} + \frac{(\mathbf{s}_i - A_{i-1}^{-1}\mathbf{y}_i)\mathbf{s}_i^\top A_{i-1}^{-1}}{\mathbf{s}_i^\top A_{i-1}^{-1}\mathbf{y}_i}$$

Steepest Descent Given a system of nonlinear equations $f_1(x_1, \ldots, x_n) = \ldots = f_1(x_n, \ldots, x_n) = 0$, we have a solution when $g(x_1, \ldots, x_n) = \sum_{i=1}^n [f_i(x_1, \ldots, x_n)]^2 = 0$. Steepest Descent linearly converges to a local minimum and will converge even for bad initial guesses. Given a coordinate function $\mathbf{F}(\mathbf{x})$, initial guess $\mathbf{x}^{(0)}$, and step size α, do the following, assuming $g(x_1, \ldots, x_n) = \sum_{i=1}^n [f_i(x_1, \ldots, x_n)]^2$.

$$x^{(k)} = x^{(k-1)} - \alpha\nabla g(x^{(k)})$$

α can be found by direct search until $g(\mathbf{x}) > g(\mathbf{x}^{(k)})$.

$$\begin{cases}\alpha = \frac{\alpha}{2} \\ x = x^{(k)} - \alpha\frac{\nabla g(x^{(k)})}{\|\nabla g(x^{(k)})\|_2}\end{cases}$$

Or α can be found by quadratic interpolation.

Considering $h(\alpha) = g(\mathbf{x}^{(k)} - \alpha\nabla g(\mathbf{x}^{(k)}))$, we construct a quadratic polynomial $P(x)$ using $\alpha_1 < \alpha_2 < \alpha_3$ to guess the minimum to $h(\hat{\alpha})$ for $\hat{\alpha} \in [\alpha_1, \alpha_3]$. We set $\alpha_1 = 0$, $\alpha_3 = c$ for some reasonable c, and $\alpha_2 = \frac{\alpha_3}{2}$. The minimum to P lies on α_2 or α_3 since $P(\alpha_3) = h(\alpha_3) < h(\alpha_1) = P(\alpha_1)$. Steepest Descent has the tendency to zig-zag around the solution.

Forward Euler's Method Is a first-order version of Runge-Kutta. For a mesh size n and initial guess $\mathbf{x}^{(0)}$

$$b = -\frac{1}{n}\mathbf{F}(x)$$

$$\begin{cases}A = J(\mathbf{x}^{(k-1)}) \\ \mathbf{x}^{(k)} = \mathbf{x}^{(k-1)} + A^{-1}b\end{cases}$$

Runge-Kutta For n iterations and initial guess $\mathbf{x}^{(0)}$. Does not require a good guess of $\mathbf{x}^{(0)}$ and converges quickly, though for RK4, it requires for linear systems to be solved when computing the $\mathbf{k}$ values, so n steps requires solving $4n$ linear systems. One step may be enough to get an accurate solution.

The generalized $\mathbf{k}_i$ values, where α_i are weights and $h = \frac{1}{n}$

$$\mathbf{k}_i = -h[\mathbf{J}(\mathbf{x}^{(k)}) + \alpha_{i-1}\mathbf{k}_{i-1})]^{-1}\mathbf{F}(\mathbf{x}^{(0)})$$

For the fourth-order Runge-Kutta problem ($n = 4$), the $\mathbf{k}_i$ values are as below, where $\alpha = (0, \frac{1}{2}, \frac{1}{2}, 1)$.

$$\begin{aligned}h &= 1/n \\ \mathbf{b} &= -h\mathbf{F}(\mathbf{x})\end{aligned}$$

$$\begin{cases}\mathbf{k}_1 = (\mathbf{J}(\mathbf{x}^{(i)}))^{-1}\mathbf{b} \\ \mathbf{k}_2 = (\mathbf{J}(\mathbf{x}^{(i)} + \dfrac{1}{2}\mathbf{k}_1))^{-1}\mathbf{b} \\ \mathbf{k}_3 = (\mathbf{J}(\mathbf{x}^{(i)} + \dfrac{1}{2}\mathbf{k}_2))^{-1}\mathbf{b} \\ \mathbf{k}_4 = (\mathbf{J}(\mathbf{x}^{(i)} + \mathbf{k}_3))^{-1}\mathbf{b} \\ \mathbf{x}^{(i+1)} = \mathbf{x}^{(i)} + \dfrac{1}{6}(\mathbf{k}_1 + 2\mathbf{k}_2 + 2\mathbf{k}_3 + \mathbf{k}_4)\end{cases}$$