Linear algebra

Vector spaces, matrices and norms

Vectors and vector spaces

- A vector $\underline{v}$ is an element of a vector space

* In this module, we will consider complex-valued vectors $\underline{x} \in \mathbb{C}^n$, which have the form

$$\underline{x} = [x_1, x_2 \dots x_n]^T, \quad \text{where } x_i \in \mathbb{C}$$

- A vector space V is a set of vectors that is closed under addition and scalar multiplication

  i.e. if $\underline{v}_1, \underline{v}_2 \in V$, then $a\underline{v}_1 + b\underline{v}_2 \in V$ for any scalars a, b

  (The scalars are often real/complex, but could also be chosen over some other field)

- A subspace of a vector space is a subset that obeys the rules of a vector space

- The inner product btwn two equal-length vectors yields a complex scalar, defined as

$$\underline{x}^H \underline{y} = \sum_i \bar{x}_i y_i$$

* $\underline{x}^H \underline{y} = \overline{\underline{y}^H \underline{x}}$ and $\underline{x}^H \underline{x}$ is real and non-negative ($\underline{x}^H \underline{x} = 0$ iff $\underline{x} = 0$)

Matrices

- A matrix $\underline{\underline{A}}$ is a linear operator that performs a linear transformation from one vector to another.

  For a matrix $\underline{\underline{A}}: \mathbb{C}^n \to \mathbb{C}^m$ where $\underline{\underline{A}}\underline{x} = \underline{b}$, the vector $\underline{x} \in \mathbb{C}^n$ is mapped to the vector $\underline{b} \in \mathbb{C}^m$

- An $n \times n$ matrix comes from the space $\mathbb{C}^{n \times n}$, so we often denote $\underline{\underline{A}} \in \mathbb{C}^{n \times n}$

- The product of two matrices $\underline{\underline{C}} = \underline{\underline{A}}\,\underline{\underline{B}}$ can be expressed using index notation

$$C_{ij} = \sum_k A_{ik} B_{kj} = A_{ik} B_{kj}$$

  Treating a column vector as a $n \times 1$ matrix, we can compute the matrix-vector product $\underline{b} = \underline{\underline{A}}\underline{x}$.

$$b_i = \sum_j A_{ij} x_j = x_1 \begin{bmatrix} \uparrow \\ \underline{a}_1 \\ \downarrow \end{bmatrix} + x_2 \begin{bmatrix} \uparrow \\ \underline{a}_2 \\ \downarrow \end{bmatrix} + \dots \quad \leftarrow \underline{a}_i \text{ are columns of } \underline{\underline{A}}$$

- The conjugate transpose / Hermitian transpose of a matrix is defined as

$$(\underline{\underline{A}}^H)_{ij} = \bar{A}_{ji}$$

  Alternatively, using the transpose, $\underline{\underline{A}}^H = \overline{\underline{\underline{A}}^T} = \bar{\underline{\underline{A}}}^T$ (conjugate transpose = transpose for real-valued matrices)

- A matrix $\underline{\underline{M}} \in \mathbb{C}^{n \times n}$ is Hermitian if

$$\underline{\underline{M}}^H = \underline{\underline{M}}$$

- A Hermitian matrix $\underline{\underline{M}} \in \mathbb{C}^{n \times n}$ has the following properties:

  (i) All eigenvalues are real, i.e. $\lambda_i \in \mathbb{R}$

  If $(\lambda, \underline{x})$ is an eigenpair of $\underline{\underline{M}}$, then $\underline{\underline{M}}\underline{x} = \lambda \underline{x}$.

  Premultiplying by $\underline{x}^H$, $\quad \underline{x}^H \underline{\underline{M}} \underline{x} = \lambda \underline{x}^H \underline{x}$

  Taking the complex conjugate $\quad (\underline{x}^H \underline{\underline{M}} \underline{x})^H = \underline{x}^H \underline{\underline{M}} \underline{x} = \bar{\lambda} \underline{x}^H \underline{x}$ (use $\underline{\underline{M}}^H = \underline{\underline{M}}$)

  Comparing the above eqns, they can only hold when $\bar{\lambda} = \lambda$,

  i.e. all the eigenvalues are real.

(ii) the eigenvectors are orthogonal, i.e. $\underline{x}_i^H \underline{x}_j = 0$ if $i \neq j$.

consider two eigenpairs of $\underline{A}$, $(\lambda_1, \underline{x}_1), (\lambda_2, \underline{x}_2)$, i.e. $\underline{A}\underline{x}_1 = \lambda_1 \underline{x}_1$, $\underline{A}\underline{x}_2 = \lambda_2 \underline{x}_2$, where $\lambda_1 \neq \lambda_2$

premultiplying by $\underline{x}_2^H / \underline{x}_1^H$, $\quad \underline{x}_2^H \underline{A} \underline{x}_1 = \lambda_1 \underline{x}_2^H \underline{x}_1 \qquad \underline{x}_1^H \underline{A} \underline{x}_2 = \lambda_2 \underline{x}_1^H \underline{x}_2$

Taking the complex conjugate for left eqn, $\quad (\underline{x}_2^H \underline{A} \underline{x}_1)^H = \underline{x}_1^H \underline{A} \underline{x}_2 = \lambda_1 \underline{x}_1^H \underline{x}_2$ (use $\underline{A}^H = \underline{A}$)

Comparing the above eqns, since $\lambda_1 \neq \lambda_2$, they can only hold if $\underline{x}_1^H \underline{x}_2$

i.e. the eigenvectors are orthogonal.

- If the eigenvalues of the matrix $\underline{A} \in \mathbb{C}^{n \times n}$ are λ_i, then

(i) the eigenvalues of the inverse $\underline{A}^{-1}$ are $1/\lambda_i$

$$\underline{A}\underline{x}_i = \lambda_i \underline{x}_i \quad \rightarrow \quad \underline{A}^{-1}\underline{x}_i = \frac{1}{\lambda_i}\underline{x}_i$$

(ii) the eigenvalues of the transpose $\underline{A}^T$ are λ_i Note that $\det(\underline{A}^T) = \det(\underline{A})$,

$\det(\underline{A} - \lambda_i \underline{I}) = 0$; $0 = \det(\underline{A}^T - \lambda \underline{I}) = \det((\underline{A} - \lambda \underline{I})^T) = \det(\underline{A} - \lambda \underline{I}) \rightarrow \lambda = \lambda_i$

(iii) the eigenvalues of the conjugate transpose $\underline{A}^H$ are $\bar{\lambda}_i$. Note that $\det(\underline{A}^H) = \det(\bar{\underline{A}}^T) = \overline{\det(\underline{A})}$

$\det(\underline{A} - \lambda_i \underline{I}) = 0$; $0 = \det(\underline{A}^H - \lambda \underline{I}) = \det((\underline{A} - \bar{\lambda} \underline{I})^H) = \overline{\det(\underline{A} - \bar{\lambda} \underline{I})} = \det(\underline{A} - \bar{\lambda} \underline{I}) \rightarrow \lambda = \bar{\lambda}_i$

- For a Hermitian matrix $\underline{M} \in \mathbb{C}^{n \times n}$, it is positive definite / semi-positive definite if

$$\underline{x}^H \underline{A} \underline{x} > 0 \quad \forall \underline{x} \in \mathbb{C}^n \setminus \underline{0} \qquad \underline{x}^H \underline{A} \underline{x} \geq 0 \quad \forall \underline{x} \in \mathbb{C}^n \setminus \underline{0}$$

The eigenvalues of a Hermitian positive definite / semipositive definite matrix are strictly +ve / +ve.

$$\lambda_i > 0 \qquad \lambda_i \geq 0$$

(say $(\lambda_i, \underline{x}_i)$ is an eigenpair of $\underline{M}$. $\underline{x}_i^H \underline{A} \underline{x}_i = \underline{x}_i^H (\lambda_i \underline{x}_i) = \lambda_i \underline{x}_i^H \underline{x}_i$, and $\underline{x}_i^H \underline{x}_i$ is real and positive)

- Note that any Hermitian positive definite matrix is invertible, since $\lambda_i > 0 \rightarrow \det(\underline{M}) = \prod_i \lambda_i \neq 0$

- For a matrix $\underline{A} \in \mathbb{C}^{m \times n}$, the matrix $\underline{A}^H \underline{A}$ is semi-positive definite, i.e. $\underline{x}^H \underline{A}^H \underline{A} \underline{x} \geq 0$.

$$0 \leq \|\underline{A}\underline{x}\|_2^2 = (\underline{A}\underline{x})^H (\underline{A}\underline{x}) = \underline{x}^H \underline{A}^H \underline{A} \underline{x}$$

so the eigenvalues of $\underline{A}^H \underline{A}$ are non-negative. ← if the eigenvalues of $\underline{A}$ are λ_i, then the eigenvalues of $\underline{A}^H \underline{A}$ are $\lambda_i \bar{\lambda}_i = |\lambda_i|^2$

- A matrix $\underline{Q} \in \mathbb{C}^{n \times n}$ is a unitary matrix if $\underline{Q}^H = \underline{Q}^{-1}$ i.e. $\underline{Q}^H \underline{Q} = \underline{Q} \underline{Q}^H = \underline{I}$.

↳ $|\det(\underline{Q})| = 1$ (since $\det(\underline{Q}^H \underline{Q}) = \det(\underline{Q}^H)\det(\underline{Q}) = |\det(\underline{Q})|^2 = \det(\underline{I}) = 1$)

↳ Transformation preserves the Euclidean norm of a vector $\|\underline{Q}\underline{x}\|_2 = \|\underline{x}\|_2$ and matrix $\|\underline{Q}\underline{A}\|_2 = \|\underline{A}\|_2$

$(\|\underline{Q}\underline{x}\|_2^2 = (\underline{Q}\underline{x})^H (\underline{Q}\underline{x}) = \underline{x}^H \underline{Q}^H \underline{Q} \underline{x} = \underline{x}^H \underline{I} \underline{x} = \underline{x}^H \underline{x} = \|\underline{x}\|_2^2)$

- The rank of a matrix $\underline{A} \in \mathbb{C}^{m \times n}$ is the no. of linearly independent rows or columns, and satisfies

$$\text{rank}(\underline{A}) \leq \min(m, n)$$

It is full rank if $\text{rank}(\underline{A}) = \min(m, n)$ and rank deficient if $\text{rank}(\underline{A}) < \min(m, n)$

- A matrix in which most entries are zero is a sparse matrix — the no. of non-zeroes on each row of a sparse matrix will be roughly the same and independent of the matrix size.

↳ The no. of entries in a dense matrix is $n \times n$

↳ The no. of entries in a sparse matrix is cn

Norms

- denotify a norm of a vector $\underline{x}$ by $\|\underline{x}\|$, a norm is a scalar that satisfies

  ↳ $\|\underline{x}\| > 0$ when $\underline{x} \neq 0$

  ↳ $\|\underline{x}\| = 0$ when $\underline{x} = 0$

  ↳ $\|k\underline{x}\| = |k|\|\underline{x}\|$

  ↳ $\|\underline{x} + \underline{y}\| \leq \|\underline{x}\| + \|\underline{y}\|$

  * for orthogonal vectors $\underline{x}, \underline{y}$, i.e. $\underline{x}^H\underline{y} = 0$,
  $\|\underline{x} \pm \underline{y}\|^2 = \|\underline{x}\|^2 \pm \|\underline{y}\|^2$

Vector norms.

- A particular family of norms are known as ℓ_p norms, defined as

$$\|\underline{x}\|_p = \left(\sum_{i=1}^{n} |x_i|^p\right)^{1/p}, \quad \text{where } |x| = \sqrt{\text{Re}[x]^2 + \text{Im}[x]^2}$$

  ↳ ℓ-1 norm: $\|\underline{x}\|_1 = |x_1| + \ldots + |x_n| = \sum_{i=1}^{n} |x_i|$ [taxi-cab norm]

  ↳ ℓ-2 norm: $\|\underline{x}\|_2 = \left(|x_1|^2 + \ldots + |x_n|^2\right)^{1/2} = \left(\sum_{i=1}^{n} |x_i|^2\right)^{1/2} = \left(\underline{x}^H\underline{x}\right)^{1/2}$ [Euclidean norm]

  ↳ ℓ-∞ norm: $\|\underline{x}\|_\infty = \max_i |x_i|$ [max-norm]

- We can define norms that involve a matrix $\underline{\underline{A}}$ if $\underline{\underline{A}}$ is positive definite.

$$\|\underline{x}\|_A = \sqrt{\underline{x}^H \underline{\underline{A}}\, \underline{x}} \quad \text{[energy norm]}$$

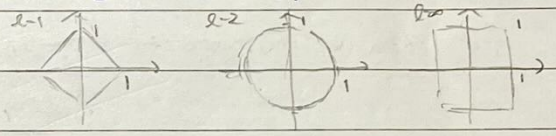
If $\underline{\underline{A}} = \underline{\underline{I}}$, we have the ℓ_2 norm

($\underline{\underline{A}}$ being positive definite implies that the energy $\underline{x}^H\underline{\underline{A}}\underline{x}$ is non-negative and only zero if $\underline{x} = 0$)

- It is useful to refer to the unit ball/sphere for a given norm.

$\{\underline{x} \in V : \|\underline{x}\| \leq 1\}$ $\{\underline{x} \in V : \|\underline{x}\| = 1\}$.

The "shape" depends on the chosen norm



Matrix norms induced by vector ℓ_p norms

- The norm of a matrix $\underline{\underline{A}}$ is defined as

$$\|\underline{\underline{A}}\| = \max_{\underline{x} \in \mathbb{C}^n \setminus 0} \frac{\|\underline{\underline{A}}\underline{x}\|}{\|\underline{x}\|} \quad \text{or} \quad \|\underline{\underline{A}}\| = \max_{\underline{x} \in \mathbb{C}^n \setminus 0, \|\underline{x}\|=1} \|\underline{\underline{A}}\underline{x}\|$$

i.e. the norm measures the "max one" by which the matrix $\underline{\underline{A}}$ can re-scale a vector $\underline{x}$

- We can express the defn of the operator norm as an inequality

$$\forall \underline{x} \quad \|\underline{\underline{A}}\|\,\|\underline{x}\| \geq \|\underline{\underline{A}}\underline{x}\|$$

It follows from this that (i) $\|\underline{\underline{A}}\underline{\underline{B}}\| \leq \|\underline{\underline{A}}\|\|\underline{\underline{B}}\|$ and (ii) $\|\underline{\underline{A}} + \underline{\underline{B}}\| \leq \|\underline{\underline{A}}\| + \|\underline{\underline{B}}\|$

(i) $\|\underline{\underline{A}}\underline{\underline{B}}\underline{x}\| \leq \|\underline{\underline{A}}\|\|\underline{\underline{B}}\underline{x}\| \leq \|\underline{\underline{A}}\|\|\underline{\underline{B}}\|\|\underline{x}\| \rightarrow \frac{\|\underline{\underline{A}}\underline{\underline{B}}\underline{x}\|}{\|\underline{x}\|} \leq \|\underline{\underline{A}}\|\|\underline{\underline{B}}\| \quad \forall \underline{x}$

since $\|\underline{\underline{A}}\underline{\underline{B}}\| = \max_{\underline{x} \in \mathbb{C}^n \setminus 0} \frac{\|\underline{\underline{A}}\underline{\underline{B}}\underline{x}\|}{\|\underline{x}\|}$, $\|\underline{\underline{B}}\underline{x}\| = \frac{\|\underline{\underline{B}}\underline{x}\|}{\|\underline{x}\|}$ for a particular $\underline{x} \rightarrow \|\underline{\underline{A}}\underline{\underline{B}}\| \leq \|\underline{\underline{A}}\|\|\underline{\underline{B}}\|$

(ii) $\|(\underline{\underline{A}}+\underline{\underline{B}})\underline{x}\| = \|\underline{\underline{A}}\underline{x} + \underline{\underline{B}}\underline{x}\| \leq \|\underline{\underline{A}}\underline{x}\| + \|\underline{\underline{B}}\underline{x}\| \leq \|\underline{\underline{A}}\|\|\underline{x}\| + \|\underline{\underline{B}}\|\|\underline{x}\| \rightarrow \frac{\|(\underline{\underline{A}}+\underline{\underline{B}})\underline{x}\|}{\|\underline{x}\|} \leq \|\underline{\underline{A}}\| + \|\underline{\underline{B}}\|$

since $\|\underline{\underline{A}}+\underline{\underline{B}}\| = \max_{\underline{x} \in \mathbb{C}^n \setminus 0} \frac{\|(\underline{\underline{A}}+\underline{\underline{B}})\underline{x}\|}{\|\underline{x}\|}$, $\|\underline{\underline{A}}+\underline{\underline{B}}\| = \frac{\|(\underline{\underline{A}}+\underline{\underline{B}})\underline{x}\|}{\|\underline{x}\|}$ for a particular $\underline{x} \rightarrow \|\underline{\underline{A}}+\underline{\underline{B}}\| \leq \|\underline{\underline{A}}\| + \|\underline{\underline{B}}\|$

- To quantify the "size" of the original vector $\underline{x}$ and the transformed vector $\underline{\underline{A}}\underline{x}$, we need to choose a norm

ℓ-1 norm: $\|\underline{\underline{A}}\|_1 = \max_{\underline{x} \in \mathbb{C}^n \setminus 0} \frac{\|\underline{\underline{A}}\underline{x}\|_1}{\|\underline{x}\|_1} = \max_j \sum_{i=1}^{n} |a_{ij}|$ ← ℓ-1 norm of the column of $\underline{\underline{A}}$ w the max. ℓ-1 norm

ℓ-2 norm: $\|\underline{\underline{A}}\|_2 = \max_{\underline{x} \in \mathbb{C}^n \setminus 0} \frac{\|\underline{\underline{A}}\underline{x}\|_2}{\|\underline{x}\|_2} = \sqrt{\lambda_{\max}(\underline{\underline{A}}^H\underline{\underline{A}})}$ ← sqrt of the largest eigenvalue of $\underline{\underline{A}}^H\underline{\underline{A}}$ or largest singular value of $\underline{\underline{A}}$

ℓ-∞ norm: $\|\underline{\underline{A}}\|_\infty = \max_{\underline{x} \in \mathbb{C}^n \setminus 0} \frac{\|\underline{\underline{A}}\underline{x}\|_\infty}{\|\underline{x}\|_\infty} = \max_i \sum_{j=1}^{n} |a_{ij}|$ ← ℓ-1 norm of the row of $\underline{\underline{A}}$ w the max ℓ-1 norm

* If $\underline{\underline{A}}$ is Hermitian, the eigenvalues of $\underline{\underline{A}}^H\underline{\underline{A}} = \underline{\underline{A}}\underline{\underline{A}}$ are the square of eigenvalues of $\underline{\underline{A}}$, so $\|\underline{\underline{A}}\|_2 = \lambda_{\max}(\underline{\underline{A}})$

- For the ℓ-2 norm of a matrix, $\|\underline{A}\|_2$,

$$\|\underline{A}\|_2^2 = \max \frac{\|\underline{A}\underline{x}\|_2^2}{\|\underline{x}\|_2^2} = \max \frac{(\underline{A}\underline{x})^H(\underline{A}\underline{x})}{\underline{x}^H\underline{x}} = \max \frac{\underline{x}^H\underline{A}^H\underline{A}\underline{x}}{\underline{x}^H\underline{x}} = \frac{\underline{x}^H\lambda_{max}(\underline{A}^H\underline{A})\underline{x}}{\underline{x}^H\underline{x}} = \lambda_{max}(\underline{A}^H\underline{A})$$

$$\therefore \|\underline{A}\|_2 = \sqrt{\lambda_{max}(\underline{A}^H\underline{A})}$$

* Note $\|\underline{A}^H\|_2 = \sqrt{\frac{1}{\lambda_{min}(\underline{A}^H\underline{A})}}$ since $\lambda(\underline{A}^H\underline{A}^{-1}) = \frac{1}{\lambda(\underline{A}^H\underline{A})}$, $\rightarrow \lambda_{max}(\underline{A}^H\underline{A}^{-H}) = \frac{1}{\lambda_{min}(\underline{A}^H\underline{A})}$

- For the ℓ-1 and ℓ-∞ norm of a matrix, $\|A\|_1$, $\|A\|_\infty$,

Recall that the operator norm $\|\underline{A}\|$ is the min. value of C that satisfies $\|\underline{A}\underline{x}\| \leq C\|\underline{x}\|$, and see $\|\underline{x}\|=1$

(i) ℓ-1 norm: $\|\underline{A}\underline{x}\|_1 = \|\sum_{j=1}^n x_j \underline{a}_j\|_1 \leq \sum_{j=1}^n |x_j| \|\underline{a}_j\|_1 \leq \max_{j\in[1,n]} \|\underline{a}_j\|_1$

(def. of $\underline{A}$; triangle inequality; since $\|\underline{x}\|_1=1$, $\sum_{j=1}^n |x_j| = 1$)

choose a vector $\underline{x}$ s.t. $x_i = \begin{cases} 1 & \text{if } i=j \\ 0 & \text{o/w} \end{cases}$. then $\|\underline{A}\underline{x}\|_1 = \max_{j\in[1,n]} \|\underline{a}_j\|_1 \rightarrow \|\underline{A}\|_1 = \max_{j\in[1,n]} \|\underline{a}_j\|_1$

(ii) ℓ-∞ norm: $\|\underline{A}\underline{x}\|_\infty = \max_{i\in[1,m]} \left|\sum_{j=1}^n x_j a_{ij}\right| \leq \max_{i\in[1,m]} \sum_{j=1}^n |x_j||a_{ij}| \leq \max_{i\in[1,m]} \sum_{j=1}^n |a_{ij}|$

(triangle inequality; since $\|\underline{x}\|_\infty = 1$, $x_i \leq 1$)

choose a vector $\underline{x}$ s.t. $x_i = \text{sign}(a_{ni})$, then $\|\underline{A}\underline{x}\|_\infty = \max_{i\in[1,m]} \sum_{j=1}^n |a_{ij}| \rightarrow \|\underline{A}\|_\infty = \max_{i\in[1,m]} \sum_{j=1}^n |a_{ij}|$

* Note that the vector and matrix ℓ-2 norms are invariant under rotation, i.e. for a unitary matrix $\underline{Q}$,

$$\|\underline{Q}\underline{x}\|_2 = \|\underline{x}\|_2 \qquad \|\underline{Q}\underline{A}\|_2 = \|\underline{A}\|_2$$

Frobenius norm

- The Frobenius norm of a matrix $\underline{A}$ is defined as

$$\|\underline{A}\|_F = \sqrt{\sum_{i,j} |a_{ij}|^2}$$

It can be shown that the Frobenius norm is also given by the following.

$$\|\underline{A}\|_F = \sqrt{\text{trace}(\underline{A}^H\underline{A})} = \sqrt{\sum_{i=1}^{\min(m,n)} \sigma_i^2(\underline{A})}$$

* Note that the Frobenius norm of a matrix is invariant under rotation, i.e. for a unitary matrix $\underline{Q}$,

$$\|\underline{Q}\underline{A}\|_F = \|\underline{A}\|_F.$$

Equivalence of norms

- Choosing a norm depends on what we want to measure

↳ eg. ℓ-1 norm for probabilities, ℓ-2 norm for distances

- On all the vector spaces that we consider, all norms are equivalent, i.e. $\exists$ consts $c_1>0, c_2>0$ s.t.

$$\forall \underline{x} \in V \quad c_1\|\underline{x}\|_\alpha \leq \|\underline{x}\|_\beta \leq c_2\|\underline{x}\|_\alpha$$

Stability and condition number

condition number.

- The condition no. of an invertible matrix $\underline{\underline{A}}$ is defined by

$$\kappa(\underline{\underline{A}}) = \|\underline{\underline{A}}\| \, \|\underline{\underline{A}}^{-1}\|$$

Note that $\kappa(\underline{\underline{A}}) \geq 1$ since $\kappa(\underline{\underline{A}}) = \|\underline{\underline{A}}\|\|\underline{\underline{A}}^{-1}\| \geq \|\underline{\underline{A}}\underline{\underline{A}}^{-1}\| = \|\underline{\underline{I}}\| = 1$

- For the ℓ_2 norm, $\|\underline{\underline{A}}\|_2 = \sqrt{\lambda_{max}(\underline{\underline{A}}^H \underline{\underline{A}})}$ and $\|\underline{\underline{A}}^{-1}\|_2 = \dfrac{1}{\sqrt{\lambda_{min}(\underline{\underline{A}}^H \underline{\underline{A}})}}$, so

$$\kappa_2(\underline{\underline{A}}) = \|\underline{\underline{A}}\|_2 \|\underline{\underline{A}}^{-1}\|_2 = \frac{\sqrt{\lambda_{max}(\underline{\underline{A}}^H \underline{\underline{A}})}}{\sqrt{\lambda_{min}(\underline{\underline{A}}^H \underline{\underline{A}})}}$$

If $\underline{\underline{A}}$ is Hermitian, then $\kappa_2(\underline{\underline{A}}) = \dfrac{|\lambda_{max}(\underline{\underline{A}})|}{|\lambda_{min}(\underline{\underline{A}})|}$

- For small κ, the shape dilates (stretch equally in all dir.)

  For large κ, the shape stretches in a particular dir. (in the limit collapses into a line)

Error bounds and stability.

- consider the problem where we have an error in making the observation $\underline{q}$

$$\underline{\underline{A}}(\underline{x} + \delta\underline{x}) = \underline{q} + \delta\underline{q}$$

where $\delta\underline{q}$ is the error in the observation and $\delta\underline{x}$ is the consequent error in the soln.

$\underline{\underline{A}}\underline{x} = \underline{q}$: $\|\underline{\underline{A}}\|\|\underline{x}\| \geq \|\underline{\underline{A}}\underline{x}\| = \|\underline{q}\| \rightarrow \dfrac{1}{\|\underline{x}\|} \leq \dfrac{\|\underline{\underline{A}}\|}{\|\underline{q}\|}$ [1]

$\underline{\underline{A}}\delta\underline{x} = \delta\underline{q}$: $\delta\underline{q} = \underline{\underline{A}}\delta\underline{x} \rightarrow \delta\underline{q} = \lambda_r \underline{\underline{A}}\underline{v}_r \rightarrow \|\delta\underline{x}\| = \|\lambda_r \underline{v}_r\| \leq \|\underline{\underline{A}}^{-1}\| \|\delta\underline{q}\|$ [2]

Combining [1], [2], $\dfrac{\|\delta\underline{x}\|}{\|\underline{x}\|} \leq \dfrac{\|\underline{q}\|}{\|\underline{q}\|} \|\underline{\underline{A}}\| \|\underline{\underline{A}}^{-1}\| = \kappa(\underline{\underline{A}}) \dfrac{\|\delta\underline{q}\|}{\|\underline{q}\|}$

rel. error in $\underline{x}$ ≤ $\kappa(\underline{\underline{A}})$ × rel. error in $\underline{b}$

i.e. for large condition no. $\kappa(\underline{\underline{A}})$, we can expect small errors in $\underline{b}$ to cause large errors in $\underline{x}$.

- consider the problem where we have an error in the matrix $\underline{\underline{A}}$ (e.g. floating pt. error in LU decomposition)

$$(\underline{\underline{A}} + \delta\underline{\underline{A}})(\underline{x} + \delta\underline{x}) = \underline{q}$$

where $\delta\underline{\underline{A}}$ is the error in the matrix and $\delta\underline{x}$ is the consequent error in the soln.

$(\underline{\underline{A}} + \delta\underline{\underline{A}})(\underline{x} + \delta\underline{x}) = \underline{\underline{A}}\underline{x} + \underline{x}\delta\underline{\underline{A}} + \delta\underline{\underline{A}}\delta\underline{x} = \underline{q} \rightarrow \delta\underline{x} = (\underline{x} + \delta\underline{x})(-\underline{\underline{A}}^{-1}\delta\underline{\underline{A}})$

$\therefore \|\delta\underline{x}\| = \|\lambda_r \underline{v}_r\| \leq \|\underline{\underline{A}}^{-1}\| \|\delta\underline{\underline{A}}\| \|\underline{x} + \delta\underline{x}\| \rightarrow \dfrac{\|\delta\underline{x}\|}{\|\underline{x} + \delta\underline{x}\|} \leq \|\underline{\underline{A}}\| \|\underline{\underline{A}}^{-1}\| \dfrac{\|\delta\underline{\underline{A}}\|}{\|\underline{\underline{A}}\|} = \kappa(\underline{\underline{A}}) \dfrac{\|\delta\underline{\underline{A}}\|}{\|\underline{\underline{A}}\|}$

i.e. for large condition no. $\kappa(\underline{\underline{A}})$, we can expect small errors in $\underline{\underline{A}}$ to cause large errors in $\underline{x}$.

- A matrix w/ a large condition no. $\kappa(\underline{\underline{A}})$ is said to be ill-conditioned ($\neq$ small determinant)

* Conditioning should not be confused w/ the determinant (i.e. large condition no. $\neq$ small determinant).

↳ $\underline{\underline{A}} = \begin{bmatrix} 1 & -1 & \cdots & -1 \\ 0 & 1 & \cdots & -1 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{bmatrix}$ $\det(\underline{\underline{A}}) = 1$, $\kappa_2(\underline{\underline{A}}) = n 2^{n-1}$

↳ $\underline{\underline{D}} = \text{diag}(0.1, 0.1, \ldots 0.1)$ $\det(\underline{\underline{D}}) = 10^{-n}$, $\kappa_2(\underline{\underline{D}}) = 1$

Interpolation and least squares methods

Polynomial interpolation.

- If we have n data pts in a 2D space (x,y), we can usually fit a polynomial w/ n coeffs

e.g. Given 6 data pts $\underline{f} = [f_0(x_0,y_0) \dots f_5(x_5,y_5)]^T$, we can interpolate the data pts w/ a polynomial of the form $f(x,y) = c_0 + c_1 x + c_2 y + c_3 xy + c_4 x^2 + c_5 y^2$

For each data pt $f(x_i,y_i)$, we have an eqn, so we can write our system of eqns in matrix form

$$\underbrace{\begin{bmatrix} 1 & x_0 & y_0 & x_0y_0 & x_0^2 & y_0^2 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & x_5 & y_5 & x_5y_5 & x_5^2 & y_5^2 \end{bmatrix}}_{\text{Vandermonde matrix } \underline{\underline{A}}} \underbrace{\begin{bmatrix} c_0 \\ \vdots \\ c_5 \end{bmatrix}}_{\underline{c}} = \underbrace{\begin{bmatrix} f_0 \\ \vdots \\ f_5 \end{bmatrix}}_{\underline{f}}$$

- In general, there are a few issues w/ solving for $\underline{c}$, i.e. finding the interpolating polynomial:
  - ↳ we can only solve the matrix eqn when $\underline{\underline{A}}$ is full rank, i.e. data pts are not colinear
  - ↳ the matrix $\underline{\underline{A}}$ is notoriously ill-conditioned, and the condition no. $\kappa(\underline{\underline{A}})$ grows w/ polynomial degree n.
  - ↳ if we choose equally-spaced interpolation pts, we get the Runge effect — large amplitude oscillations near the ends of the domain, which become more severe at high polynomial degrees.

Using a different set of basis functions

- So far, we have considered the monomial basis $\{1, x \dots x^{n-1}\}$, so we have polynomials of the form

$$f = c_{n-1}x^{n-1} + c_{n-2}x^{n-2} + \dots + c_1 x + c_0$$

where we would solve for the coefficients $\underline{c}$ by solving the matrix eqn.

$$\begin{bmatrix} 1 & x_0 & \cdots & x_0^{n-2} & x_0^{n-1} \\ \vdots & \vdots & & \vdots & \vdots \\ 1 & x_{n-1} & \cdots & x_{n-1}^{n-2} & x_{n-1}^{n-1} \end{bmatrix} \begin{bmatrix} c_0 \\ c_1 \\ \vdots \\ c_{n-2} \\ c_{n-1} \end{bmatrix} = \begin{bmatrix} f_0 \\ f_1 \\ \vdots \\ f_{n-2} \\ f_{n-1} \end{bmatrix}$$

- Instead, we could consider the Legendre polynomial basis $\{P_0(x), P_1(x) \dots P_{n-1}(x)\}$, where

$$(n+1)P_{n+1}(x) = (2n+1)xP_n(x) - nP_{n-1}(x), \quad \text{and} \quad P_0 = 1,\ P_1 = x$$

Note that the Legendre polynomials form an orthogonal basis on the interval $[-1,1]$, i.e.

$$\langle P_m, P_n \rangle = \int_{-1}^{1} P_m(x)P_n(x)\,dx = 0 \quad \text{if } m \neq n$$

Therefore, we have polynomials of the form

$$f = \alpha_{n-1}P_{n-1}(x) + \alpha_{n-2}P_{n-2}(x) + \dots + \alpha_1 P_1(x) + \alpha_0 P_0(x)$$

where we would solve for the coefficients $\underline{\alpha}$ by solving the matrix eqn.

$$\begin{bmatrix} P_0(x_0) & P_1(x_0) & \cdots & P_{n-2}(x_0) & P_{n-1}(x_0) \\ \vdots & \vdots & & \vdots & \vdots \\ P_0(x_{n-1}) & P_1(x_{n-1}) & \cdots & P_{n-2}(x_{n-1}) & P_{n-1}(x_{n-1}) \end{bmatrix} \begin{bmatrix} \alpha_0 \\ \alpha_1 \\ \vdots \\ \alpha_{n-2} \\ \alpha_{n-1} \end{bmatrix} = \begin{bmatrix} f_0 \\ f_1 \\ \vdots \\ f_{n-2} \\ f_{n-1} \end{bmatrix}$$

- Using the Legendre polynomials instead of monomials as the basis significantly reduced the condition no. $\kappa(\underline{\underline{A}})$ of the Vandermonde matrix $\underline{\underline{A}}$.

Interpolation w/ non-equispaced points.

- The Lebesgue const. $\Lambda(T)$ bounds the interpolation error. For a function f, where P_n is the best approx of f over all polynomials of degree n and p_ℓ is the degree-n polynomial interpolant of f,

$$\|f - p_\ell\| \leq (\Lambda(T)+1)\|f - P_n\|$$

Note that the Lebesgue const $\Lambda(T)$ depends on the choice of interpolation pts T.

- For a good polynomial interpolant p_ℓ, we want the Lebesgue const $\Lambda(T)$ to be as small as possible.
- By using non-equispaced sampling pts — roots of orthogonal polynomials in particular (they tend to cluster close to the ends of the interval), we can mitigate the Runge effect (i.e. small $\Lambda(T)$)
- The Lebesgue const $\Lambda(T)$ grows exponentially in n for equispaced pts but only logarithmically in n for chebyshev pts (roots of chebyshev polynomials, defined as $T_n(x) = \cos(n\arccos(x))$, $x \in [-1,1]$).

Least squares method for data fitting of over-determined systems.

- To find a soln to the matrix eqn $\underline{\underline{A}}\underline{x} = \underline{b}$, the vector $\underline{b}$ must lie in the column space of $\underline{\underline{A}}$.
If $\underline{\underline{A}}$ is a $m \times n$ matrix w/ $m > n$ (skinny matrix), in general $\underline{b}$ does not lie in the column space of $\underline{\underline{A}}$.
- For an approx soln $\hat{\underline{x}}$, we can define the residual vector $\underline{r}$ as

$$\underline{r} = \underline{\underline{A}}\hat{\underline{x}} - \underline{b}$$

we want to choose $\hat{\underline{x}}$ s.t. the norm of the residual vector $\|\underline{r}\|$ is min. for some norm, i.e.

$$\hat{\underline{x}}^* = \underset{\hat{\underline{x}} \in \mathbb{C}^n}{\operatorname{argmin}} \|\underline{r}(\hat{\underline{x}})\| = \underset{\hat{\underline{x}} \in \mathbb{C}^n}{\operatorname{argmin}} \|\underline{\underline{A}}\hat{\underline{x}} - \underline{b}\|$$

- If we use the l-2 norm, we seek $\hat{\underline{x}}^* = \underset{\hat{\underline{x}} \in \mathbb{C}^n}{\operatorname{argmin}} \|\underline{\underline{A}}\hat{\underline{x}} - \underline{b}\|_2$. Let $r(\hat{\underline{x}}) = \|\underline{\underline{A}}\hat{\underline{x}} - \underline{b}\|_2^2$. We have

$$r(\hat{\underline{x}}) = \|\underline{\underline{A}}\hat{\underline{x}} - \underline{b}\|_2^2 = \sum_{i=1}^{m} \left| \sum_{j=1}^{n} (a_{ij}\hat{x}_j) - b_i \right|^2 \rightarrow \text{least squares method}$$

Expanding and minimising $r(\hat{\underline{x}})$,

$$r(\hat{\underline{x}}) = \|\underline{\underline{A}}\hat{\underline{x}} - \underline{b}\|_2^2 = (\underline{\underline{A}}\hat{\underline{x}} - \underline{b})^H(\underline{\underline{A}}\hat{\underline{x}} - \underline{b}) = \hat{\underline{x}}^H\underline{\underline{A}}^H\underline{\underline{A}}\hat{\underline{x}} - \hat{\underline{x}}^H\underline{\underline{A}}^H\underline{b} - \underline{b}^H\underline{\underline{A}}\hat{\underline{x}} + \underline{b}^H\underline{b}$$

$$\therefore \frac{dr(\hat{\underline{x}})}{d\hat{\underline{x}}} = 2\underline{\underline{A}}^H\underline{\underline{A}}\hat{\underline{x}} - 2\underline{\underline{A}}^H\underline{b} = 0 \quad \rightarrow \quad \underline{\underline{A}}^H\underline{\underline{A}}\hat{\underline{x}} = \underline{\underline{A}}^H\underline{b}$$

Rearranging the least squares problem, we have

$$\hat{\underline{x}}^* = (\underline{\underline{A}}^H\underline{\underline{A}})^{-1}\underline{\underline{A}}^H\underline{b} = \underline{\underline{A}}^{\dagger}\underline{b}$$

where $\underline{\underline{A}}^{\dagger} = (\underline{\underline{A}}^H\underline{\underline{A}})^{-1}\underline{\underline{A}}^H$ is the pseudoinverse / penrose-Moore inverse.

* The inverse $(\underline{\underline{A}}^H\underline{\underline{A}})^{-1}$ can be computed when $\underline{\underline{A}}$ is full rank

- To solve the least squares problem $\underline{\underline{A}}^H\underline{\underline{A}}\hat{\underline{x}}^* = \underline{\underline{A}}^H\underline{b}$, if $\underline{\underline{A}}$ is full rank, we can apply LU factorisation or cholesky factorisation to $\underline{\underline{A}}^H\underline{\underline{A}}$, but $\underline{\underline{A}}^H\underline{\underline{A}}$ is notoriously ill-conditioned
- We can use more specialised methods like QR factorisation or SVD to manage the ill-conditioning of $\underline{\underline{A}}^H\underline{\underline{A}}$ (QR may have accuracy/stability problems and SVD is expensive)

* In practice, do not write your own least-squares solver → use specialised functions/libraries

Iterative methods for linear systems

Direct and iterative methods.

- To solve a system of the form $\underline{\underline{A}}\underline{x} = \underline{b}$, soln can be categorised as direct or iterative.

① Direct methods.

- A direct method computes a soln to $\underline{\underline{A}}\underline{x} = \underline{b}$ in a known/predictable no. of operations.
  W/O round-off errors, the soln is exact. (eg. LU factorisation)
- In general, direct methods cannot handle singular eqns
- For reasonably conditioned systems, direct solvers can be robust.
- Direct solvers do not scale well on parallel computers as substitution steps are serial.

② Iterative methods.

- An iterative method seeks an approx soln via a series of iterations, and is usually terminated when the error/residual falls below a prescribed threshold.
- Some iterative methods can solve singular eqns.
- Iterations can be terminated early if high accuracy is not req.
- Iterative methods use less memory and are suited to large parallel computers.

Power iteration for eigenvector w/ largest eigenvalue

- We can use power iteration to find the eigenvector associated w/ the largest absolute eigenvalue.
- A vector $\underline{x} \in \mathbb{C}^n$ can be expressed in terms of the n eigenvectors of an $n \times n$ matrix.

$$\underline{x} = \sum_{i=1}^{n} \alpha_i \underline{v}_i \quad \text{where } \alpha_i \in \mathbb{R}.$$

If we multiply $\underline{x}$ repeatedly by $\underline{\underline{A}}$,

$$\underline{\underline{A}}^k \underline{x} = \sum_{i=1}^{n} \alpha_i \lambda_i^k \underline{v}_i$$

∴ If the largest eigenvalue is distinct, the resulting vector will be aligned w/ the eigenvector of the largest eigenvalue (given $\underline{x}$ has a component in the dir. of the eigenvector)

- Similar methods inc. the inverse power method $\underline{\underline{A}}^{-1}\underline{x} = \frac{1}{\lambda}\underline{x}$ → eigenvector w/ smallest eigenvalue
  and the shifted inverse power method $(\underline{\underline{A}} - \alpha \underline{\underline{I}})^{-1} = \frac{1}{\lambda - \alpha}\underline{x}$ → eigenvector w/ eigenvalue closest to α.

* These methods can converge slowly

Rayleigh quotient for eigenvalues

- We can use the Rayleigh quotient $R(\underline{\underline{A}}, \underline{x})$ to find the eigenvalue λ^q corresponding to the given estimate of an eigenvector $\underline{x}$.

- Let $\underline{x}$ be an estimate of an eigenvector of the matrix $\underline{\underline{A}}$, so $\underline{\underline{A}}\underline{x} \approx \lambda \underline{x}$.

To find an estimate of the corresponding eigenvalue λ^q, we consider the minimisation problem in l_2 norm,

$$\lambda^q = \underset{\lambda \in \mathbb{C}}{\operatorname{argmin}} \|\underline{\underline{A}}\underline{x} - \hat{\lambda}\underline{x}\|_2$$

This is minimised when λ^q is equal the Rayleigh quotient $R(\underline{\underline{A}}, \underline{x})$,

$$\boxed{\lambda^q = R(\underline{\underline{A}}, \underline{x}) = \frac{\underline{x}^H \underline{\underline{A}}^H \underline{x}}{\underline{x}^H \underline{x}}}$$

- To prove this, consider any $\mu \in \mathbb{C}$,

$$\|\underline{\underline{A}}\underline{x} - \mu\underline{x}\|_2^2 = \underline{x}^H \underline{\underline{A}}^H \underline{\underline{A}}\underline{x} - \mu \underline{x}^H \underline{\underline{A}}^H \underline{x} - \bar{\mu}\underline{x}^H \underline{\underline{A}}\underline{x} + |\mu|^2 \underline{x}^H \underline{x} = \|\underline{\underline{A}}\underline{x}\|_2^2 + \|\underline{x}\|_2^2 (|\mu|^2 - \mu\bar{\lambda}^q - \bar{\mu}\lambda^q)$$

$$= \|\underline{\underline{A}}\underline{x}\|_2^2 + |\underline{x}\|_2^2 \left((\mu - \lambda^q)(\bar{\mu} - \bar{\lambda}^q) - |\lambda^q|^2 \right) \geq \|\underline{\underline{A}}\underline{x}\|_2^2 - |\lambda^q|^2 \|\underline{x}\|_2^2 = \|\underline{\underline{A}}\underline{x} - \lambda^q \underline{x}\|_2^2$$

($|\mu - \lambda^q|^2 \geq 0$)

- The Rayleigh quotient $R(\underline{\underline{A}}, \underline{x})$ is often defined for Hermitian matrices only, where it is real-valued.

It also has other properties, inc. second-order accuracy for estimate of eigenvalue (eigenvector error $O(\varepsilon)$, eigenvalue error $O(\varepsilon^2)$)

Stationary methods for $\underline{\underline{A}}\underline{x} = \underline{b}$

- Consider decomposing the matrix operator s.t. $\boxed{\underline{\underline{A}} = \underline{\underline{N}} - \underline{\underline{P}}}$. The matrix eqn. $\underline{\underline{A}}\underline{x} = \underline{b}$ becomes,

$$(\underline{\underline{N}} - \underline{\underline{P}})\underline{x} = \underline{b} \quad \rightarrow \quad \underline{\underline{N}}\underline{x} = \underline{b} + \underline{\underline{P}}\underline{x}$$

We can compute an approx soln $\underline{x}_{k+1}$

$$\boxed{\underline{\underline{N}}\underline{x}_{k+1} = \underline{b} + \underline{\underline{P}}\underline{x}_k}$$

where $\underline{x}_k$ is the most recent known estimate of the soln

- We keep iterating for the new estimate $\underline{x}_{k+1}$ and hopefully this converges to the exact soln.

- The key is to split $\underline{\underline{A}}$ s.t. eqns of the form $\underline{\underline{N}}\underline{x} = \underline{f}$ are easy (cheap) to solve, examples inc.

↳ Richardson: $\underline{\underline{N}} = \underline{\underline{I}}$ ↳ Jacobi: $\underline{\underline{N}} = \operatorname{diag}(\underline{\underline{A}})$.

↳ Gauss-Seidel: $\underline{\underline{N}} = L(\underline{\underline{A}})$ (lower triangular part of $\underline{\underline{A}}$, inc. the diagonal).

- Defining the error at the kth iterate as $\boxed{\underline{e}_k = \underline{x}_{exact} - \underline{x}_k}$, we have,

$$\underline{\underline{N}}(\underline{x}_{exact} - \underline{e}_{k+1}) = (\underline{\underline{N}} - \underline{\underline{P}})\underline{x}_{exact} + \underline{\underline{P}}(\underline{x}_{exact} - \underline{e}_k) \quad \rightarrow \quad \underline{\underline{N}}\underline{e}_{k+1} = \underline{\underline{P}}\underline{e}_k$$

$$\therefore \boxed{\underline{e}_{k+1} = \underline{\underline{N}}^{-1}\underline{\underline{P}}\underline{e}_k}$$

Expressing the initial error vector $\underline{e}_0$ as a linear combination of the eigenvectors of $\underline{\underline{N}}^{-1}\underline{\underline{P}}$,

$$\underline{e}_0 = c_1 \underline{u}_1 + \ldots + c_n \underline{u}_n$$

The error $\underline{e}_k$ is therefore

$$\underline{e}_k = (\underline{\underline{N}}^{-1}\underline{\underline{P}})^k \underline{e}_0 = c_1 \lambda^k \underline{u}_1 + \ldots + c_n \lambda^k \underline{u}_n$$

- For convergence, we req. $\underline{e}_k \to 0$ as k increases, which is true if the absolute value of every eigenvalue of $\underline{\underline{N}}^{-1}\underline{\underline{P}}$ is less than one — i.e. the spectral radius of $\underline{\underline{N}}^{-1}\underline{\underline{P}}$ is less than 1, $\boxed{\rho(\underline{\underline{N}}^{-1}\underline{\underline{P}}) < 1}$.

* The smaller $\rho < 1$, the faster the convergence.

Conjugate gradient (CG) method

- The CG method is a Krylov subspace method for solving $\underline{\underline{A}}\underline{x}=\underline{b}$, where $\underline{\underline{A}}$ is Hermitian, positive definite, and is more powerful than the stationary methods outlined above.
- The CG method is often presented as a direct method, but is applied in practice usually as an iterative method.

① CG method as a direct method.

- consider $\underline{\underline{A}}\underline{x}=\underline{b}$, where $\underline{\underline{A}}$ is a $n\times n$ Hermitian positive-definite matrix.

  consider that we have a set P of n non-zero vectors $P=\{\underline{p}_0 \dots \underline{p}_{n-1}\}$ that are A-conjugate, i.e.

$$\boxed{\underline{p}_i^H \underline{\underline{A}}\, \underline{p}_j = 0 \quad \text{if } i\neq j}.$$

  Note that since $\underline{\underline{A}}$ is positive definite, $\underline{p}_i^H \underline{\underline{A}}\, \underline{p}_i > 0$

- Using P as a basis for the soln, $\boxed{\underline{x}=\sum_{i=0}^{n-1}\alpha_i \underline{p}_i}$, so we have

$$\underline{b}=\underline{\underline{A}}\,\underline{x}=\sum_{i=0}^{n-1}\alpha_i \underline{\underline{A}}\,\underline{p}_i$$

  Pre-multiplying by $\underline{p}_j^H$

$$\underline{p}_j^H \underline{b}=\underline{p}_j^H \underline{\underline{A}}\,\underline{x}=\sum_{i=0}^{n-1}\alpha_i \underline{p}_j^H \underline{\underline{A}}\,\underline{p}_i=\alpha_j \underline{p}_j^H \underline{\underline{A}}\,\underline{p}_j$$

  We can therefore solve for all α_i and get $\underline{x}=\sum_{i=0}^{n-1}\alpha_i \underline{p}_i$, where α_j is given by

$$\boxed{\alpha_j=\frac{\underline{p}_j^H \underline{b}}{\underline{p}_j^H \underline{\underline{A}}\,\underline{p}_j}}$$

- To generate the A-conjugate set P, we can simply pick n linearly independent vectors and apply the Gram-Schmidt process to build P, but this is not practical for large problems ← build + store P too expensive

② CG method as an iterative method.

- The CG algorithm involves computing elements of P using a short difference eqn. Moreover, we may not even need all of P – we can get an accurate soln w/ a few elements of P.
- The CG algorithm only involves matrix-vector products and dot products, and only req. storing a few work vectors (low space complexity)
- When solving $\underline{\underline{A}}\underline{x}=\underline{b}$ where $\underline{\underline{A}}$ is a $n\times n$ matrix:
  - ↳ The error $\underline{e}_k=\underline{x}_{\text{exact}}-\underline{x}_k$ is monotone in the A-norm ($\|\underline{e}_k\|_A=\sqrt{\underline{e}_k^H \underline{\underline{A}}\,\underline{e}_k}$), i.e. $\boxed{\|\underline{e}_{k+1}\|_A \leq \|\underline{e}_k\|_A}$ and converges to 0, i.e. $\|\underline{e}_k\|_A$ for some $k\leq n$. (in exact arithmetic)
  - ↳ The no. of iterations req. to solve $\underline{x}$ exactly is equal to the no. of distinct eigenvalues of $\underline{\underline{A}}$ (so $k\leq n$)
  - ↳ The rate of convergence is affected by the condition no. $\kappa_2(\underline{\underline{A}})$,

$$\boxed{\frac{\|\underline{e}_k\|_A}{\|\underline{e}_0\|_A} \leq 2\left(\frac{\sqrt{\kappa_2(\underline{\underline{A}})}-1}{\sqrt{\kappa_2(\underline{\underline{A}})}+1}\right)^k}$$

  Note when $\kappa_2(\underline{\underline{A}})$ is close to 1, $\underline{e}_k$ converges to 0 quickly.
- If the condition no $\kappa(\underline{\underline{A}})$ is too large, the CG method may converge too slowly for practical use

  → we can pre-condition the problem, where we consider the transformed system

  left preconditioning → $\boxed{\underline{\underline{P}}^{-1}\underline{\underline{A}}\,\underline{x}=\underline{\underline{P}}^{-1}\underline{b}}$

  If $\underline{\underline{P}}^{-1}\approx\underline{\underline{A}}^{-1}$, the condition no. of $\underline{\underline{P}}^{-1}\underline{\underline{A}}$ is much better than $\underline{\underline{A}}$ → CG method converges faster
  + In practice, it can be a balancing act – $\underline{\underline{P}}^{-1}$ must be cheap to apply but close enough to $\underline{\underline{A}}^{-1}$ to be effective.

Singular value decomposition (SVD)

Matrix diagonalisation.

- We can diagonalise any Hermitian matrix $\underline{\underline{M}} \in \mathbb{C}^{n \times n}$

$$\underline{\underline{M}} = \underline{\underline{Q}}\,\underline{\underline{\Delta}}\,\underline{\underline{Q}}^H$$

where the columns of $\underline{\underline{Q}}$ are the l_2 normalised eigenvectors of $\underline{\underline{M}}$ (which are orthogonal $\rightarrow$ $\underline{\underline{Q}}$ is unitary).

and $\underline{\underline{\Delta}}$ is the diagonal matrix of the eigenvalues of $\underline{\underline{M}}$ (which are real).

- We can interpret the diagonalisation as a sum of rank 1 matrices.

$$\underline{\underline{M}} = \sum_i \lambda_i \underline{u}_i \underline{u}_i^H$$

where $(\lambda_i, \underline{u}_i)$ is the ith eigenpair of $\underline{\underline{M}}$.

- Note that the eigenvectors are not unique $\rightarrow$ sign ambiguity $\underline{\underline{M}} = \underline{\underline{Q}}\,\underline{\underline{\Delta}}\,\underline{\underline{Q}}^H = (-\underline{\underline{Q}})\,\underline{\underline{\Delta}}\,(-\underline{\underline{Q}})^H$

- Matrix diagonalisation is v. helpful as we see that in a particular basis, the action of a matrix on a vector is easy to interpret, and matrix-matrix operations become trivial

- However, matrix diagonalisation has a few major drawbacks

  ↳ only valid for square matrices ↳ defective matrices cannot be diagonalised

  ↳ The matrix $\underline{\underline{Q}}$ is guaranteed unitary only for Hermitian matrices $\underline{\underline{M}}$.

Singular value decomposition (SVD)

- The SVD of a $m \times n$ matrix $\underline{\underline{A}}$ is

$$\underline{\underline{A}} = \underline{\underline{U}}\,\underline{\underline{\Sigma}}\,\underline{\underline{V}}^H$$

where $\underline{\underline{U}} \in \mathbb{C}^{m \times m}$ is a unitary matrix. $\underline{\underline{V}} \in \mathbb{C}^{n \times n}$ is a unitary matrix

$p = \min(m, n)$

$\underline{\underline{\Sigma}} \in \mathbb{R}^{m \times n}$ is a diagonal matrix, w/ diagonal entries σ_i (singular values) sorted st. $\sigma_1 \geq \ldots \geq \sigma_p \geq 0$

- Consider $\underline{\underline{S}}_R = \underline{\underline{A}}^H \underline{\underline{A}}$ (which is Hermitian)

$$\underline{\underline{S}}_R = \underline{\underline{A}}^H \underline{\underline{A}} = (\underline{\underline{U}}\,\underline{\underline{\Sigma}}\,\underline{\underline{V}}^H)^H (\underline{\underline{U}}\,\underline{\underline{\Sigma}}\,\underline{\underline{V}}^H) = \underline{\underline{V}}\,\underline{\underline{\Sigma}}^H \underline{\underline{U}}^H \underline{\underline{U}}\,\underline{\underline{\Sigma}}\,\underline{\underline{V}}^H = \underline{\underline{V}}\,\underline{\underline{\Sigma}}^H \underline{\underline{\Sigma}}\,\underline{\underline{V}}^H$$

→ The columns of $\underline{\underline{V}}$ are the normalised eigenvectors of $\underline{\underline{A}}^H \underline{\underline{A}}$

The diagonal entries of $\underline{\underline{\Sigma}}^H \underline{\underline{\Sigma}}$ $\sigma_1^2, \sigma_2^2 \ldots \sigma_p^2$ are the eigenvalues of $\underline{\underline{A}}^H \underline{\underline{A}}$

- Consider $\underline{\underline{S}}_L = \underline{\underline{A}}\,\underline{\underline{A}}^H$ (which is Hermitian)

$$\underline{\underline{S}}_L = \underline{\underline{A}}\,\underline{\underline{A}}^H = (\underline{\underline{U}}\,\underline{\underline{\Sigma}}\,\underline{\underline{V}}^H)(\underline{\underline{U}}\,\underline{\underline{\Sigma}}\,\underline{\underline{V}}^H)^H = \underline{\underline{U}}\,\underline{\underline{\Sigma}}\,\underline{\underline{V}}^H \underline{\underline{V}}\,\underline{\underline{\Sigma}}^H \underline{\underline{U}}^H = \underline{\underline{U}}\,\underline{\underline{\Sigma}}\,\underline{\underline{\Sigma}}^H \underline{\underline{U}}^H$$

same eigenvalues

→ The columns of $\underline{\underline{U}}$ are the normalised eigenvectors of $\underline{\underline{A}}\,\underline{\underline{A}}^H$

The diagonal entries of $\underline{\underline{\Sigma}}\,\underline{\underline{\Sigma}}^H$ $\sigma_1^2, \sigma_2^2 \ldots \sigma_p^2$ are the eigenvalues of $\underline{\underline{A}}\,\underline{\underline{A}}^H$.

- To determine the sign of the eigenvectors, postmultiply both sides of $\underline{\underline{A}} = \underline{\underline{U}}\,\underline{\underline{\Sigma}}\,\underline{\underline{V}}^H$ by $\underline{\underline{V}}$, so

$$\underline{\underline{A}}\,\underline{\underline{V}} = \underline{\underline{U}}\,\underline{\underline{\Sigma}} \quad \rightsquigarrow \quad \underline{\underline{A}}\,\underline{v}_i = \sigma_i \underline{u}_i$$

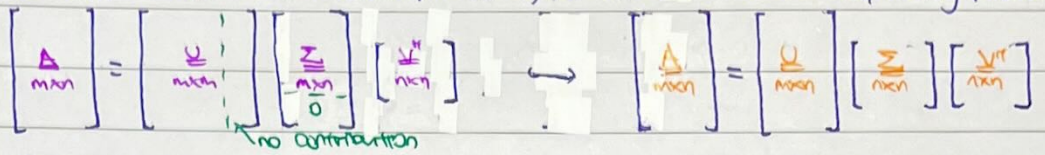
where $\underline{v}_i$ is the ith column of $\underline{\underline{V}}$ and $\underline{u}_i$ is the ith column of $\underline{\underline{U}}$.

* Every matrix has a SVD, which is unique up to the signs of the eigenvectors in $\underline{\underline{U}}$ and $\underline{\underline{V}}$.

* The SVD of a Hermitian matrix is the usual matrix diagonalisation.

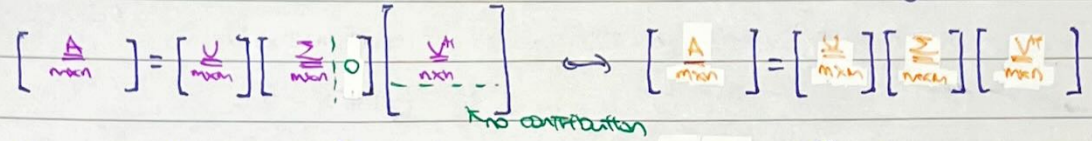
The reduced SVD.

- For a $m \times n$ matrix w/ $m > n$ (skinny matrix), the SVD can be reduced by removing redundant entries



The terms below row n in $\underline{\underline{\Sigma}}$ are always zero → last $m-n$ columns of $\underline{\underline{U}}$ make no contribution.

- For a $m \times n$ matrix w/ $m < n$ (fat matrix), the SVD can be reduced by removing redundant entries



The terms right of column m in $\underline{\underline{\Sigma}}$ are always zero → last $n-m$ rows of $\underline{\underline{V}}^T$ make no contribution

Low rank approximations

- If we expand the SVD $\underline{\underline{A}} = \underline{\underline{U}}\underline{\underline{\Sigma}}\underline{\underline{V}}^T$, we get a sum of rank-1 matrices.

$$\underline{\underline{A}} = \sum_{i=1}^{r} \sigma_i \underline{u}_i \underline{v}_i^T$$

where r is the rank of $\underline{\underline{A}}$ (no. of non-zero singular values),

and $\underline{u}_i$ is the ith column of $\underline{\underline{U}}$, $\underline{v}_i$ is the ith column of $\underline{\underline{V}}$.

- Noting that σ_i are sorted in descending order, a rank k approx of $\underline{\underline{A}}$ is given by

$$\underline{\underline{A}}_k = \sum_{i=1}^{k} \sigma_i \underline{u}_i \underline{v}_i^T \quad , \quad k < r$$

$\underline{\underline{A}}_k$ is a better approx of $\underline{\underline{A}}$ than any other matrix of rank k or less, $\underline{\underline{B}}_k$.

(i) $\|\underline{\underline{A}} - \underline{\underline{A}}_k\|_F = \sqrt{\sigma_{k+1}^2 + \dots + \sigma_r^2} \leq \|\underline{\underline{A}} - \underline{\underline{B}}_k\|_F$ or (ii) $\|\underline{\underline{A}} - \underline{\underline{A}}_k\|_2 = \sigma_{k+1} \leq \|\underline{\underline{A}} - \underline{\underline{B}}_k\|_2$

↳ proof of (ii): $\underline{\underline{A}} - \underline{\underline{A}}_k = \sum_{i=1}^{r} \sigma_i \underline{u}_i \underline{v}_i^T - \sum_{i=1}^{k} \sigma_i \underline{u}_i \underline{v}_i^T = \sum_{i=k+1}^{r} \sigma_i \underline{u}_i \underline{v}_i^T$

$\therefore \|\underline{\underline{A}} - \underline{\underline{A}}_k\|_2 = \left\| \sum_{i=k+1}^{r} \sigma_i \underline{u}_i \underline{v}_i^T \right\|_2 = \sigma_{k+1}$ — 2-2 norm of a matrix is equal to the largest singular value

As $\underline{\underline{B}}_k$ has rank k, there exists a vector $\underline{w} = \alpha_1 \underline{v}_1 + \dots + \alpha_{k+1} \underline{v}_{k+1}$ that lies in the null space of $\underline{\underline{B}}_k$.

Denoting $\underline{\underline{\Sigma}}_{k+1}$ as the first $k+1$ entries of $\underline{\underline{\Sigma}}$ and $\underline{w}$ as $\underline{w} = \begin{bmatrix} \uparrow & & \uparrow \\ \underline{v}_1 & \cdots & \underline{v}_{k+1} \\ \downarrow & & \downarrow \end{bmatrix} \begin{bmatrix} \alpha_1 \\ \vdots \\ \alpha_{k+1} \end{bmatrix} = \underline{\underline{V}}_{k+1} \underline{\alpha}$,

For convenience, normalize $\underline{w}$ s.t. $\|\underline{w}\|_2 = 1$ (so $\|\underline{\alpha}\| < 1$)

$$\|\underline{\underline{A}} - \underline{\underline{B}}_k\|_2^2 \geq \|(\underline{\underline{A}} - \underline{\underline{B}}_k)\underline{w}\|_2^2 = \|\underline{\underline{A}}\underline{w} - \underline{\underline{B}}_k\underline{w}\|_2^2 = \|\underline{\underline{A}}\underline{w}\|_2^2 = \|\underline{\underline{U}}\underline{\underline{\Sigma}}\underline{\underline{V}}^T\underline{\underline{V}}_{k+1}\underline{\alpha}\|_2^2 = \|\underline{\underline{\Sigma}}_{k+1}\underline{\alpha}\|_2^2$$

(norm invariant under rotation; $\underline{w} \perp$ rows of $\underline{\underline{B}}_k$ by construction of $\underline{w}$)

$$= \alpha_1^2\sigma_1^2 + \dots + \alpha_{k+1}^2\sigma_{k+1}^2 \geq \sigma_{k+1}^2 = \|\underline{\underline{A}} - \underline{\underline{A}}_k\|_2^2 \rightarrow \|\underline{\underline{A}} - \underline{\underline{B}}_k\|_2 \geq \|\underline{\underline{A}} - \underline{\underline{A}}_k\|_2$$

Effective rank

- When taking measurements, noise is often unavoidable → hard to detect linear dependencies.

↳ eg. $\underline{\underline{A}} = \begin{bmatrix} 1 & 1 & 1 \\ 2 & 2 & 2 \\ 1 & 0 & 1 \end{bmatrix}$ (rank 2), $\underline{\underline{A}}_{noisy} = \begin{bmatrix} 1.0000022 & 1.0000006 & 1.0000014 \\ 2.0000018 & 2.0000015 & 2.0000011 \\ 1.0000008 & 2.1924\times10^{-6} & 1.0000012 \end{bmatrix}$ (rank 3) $(\text{noise} \in [0, 10^{-6}])$

SVD on $\underline{\underline{A}}_{noisy}$ gives $\sigma_1 = 4.048$, $\sigma_2 = 0.7811$, $\sigma_3 = 3.9\times10^{-7}$ → effective rank = 2

- The effective rank is the no. of singular values that are greater than the noise level.

Least squares solution — full rank case.

- If $\underline{\underline{A}}$ is a $m \times n$ matrix w/ $m > n$, we can partition the SVD as

$$\underline{\underline{A}} = [\underline{\underline{U}}_1 \; \underline{\underline{U}}_2] \begin{bmatrix} \underline{\underline{\Sigma}}_1 \\ \underline{\underline{0}} \end{bmatrix} \underline{\underline{V}}^H = \underline{\underline{U}}_1 \underline{\underline{\Sigma}}_1 \underline{\underline{V}}^H$$

since $\underline{\underline{A}}$ is full rank, $\underline{\underline{\Sigma}}_1$ will be $n \times n$, and $\underline{\underline{U}}_1$ is $m \times n$.

* Note the columns of $\underline{\underline{U}}$ are orthonormal $\rightarrow \underline{\underline{U}}_1^H \underline{\underline{U}}_1 = \underline{\underline{I}}_{n \times n}$ and $\underline{\underline{U}}_2^H \underline{\underline{U}}_1 = \underline{\underline{0}}_{(m-n) \times n}$

- A least squares soln minimizes $r = \|\underline{\underline{A}}\hat{\underline{x}} - \underline{b}\|_2^2$

$$r = \|\underline{\underline{A}}\hat{\underline{x}} - \underline{b}\|_2^2 = \left\| \begin{bmatrix} \underline{\underline{U}}_1^H \\ \underline{\underline{U}}_2^H \end{bmatrix} \right\|_2^2 \left\| \begin{bmatrix} \underline{\underline{U}}_1 \underline{\underline{\Sigma}}_1 \underline{\underline{V}}^H \hat{\underline{x}} - \underline{b} \end{bmatrix} \right\|_2^2 = \left\| \begin{bmatrix} \underline{\underline{\Sigma}}_1 \underline{\underline{V}}^H \hat{\underline{x}} - \underline{\underline{U}}_1^H \underline{b} \\ -\underline{\underline{U}}_2^H \underline{b} \end{bmatrix} \right\|_2^2 = \left\| \begin{bmatrix} \underline{\underline{\Sigma}}_1 \underline{\underline{V}}^H \hat{\underline{x}} - \underline{\underline{U}}_1^H \underline{b} \\ \underline{\underline{U}}_2^H \underline{b} \end{bmatrix} \right\|_2^2 = \left\| \underline{\underline{U}}_1^H \underline{b} - \underline{\underline{\Sigma}}_1 \underline{\underline{V}}^H \hat{\underline{x}} \right\|_2^2 + \left\| \underline{\underline{U}}_2^H \underline{b} \right\|_2^2$$

(the second term is independent of $\hat{\underline{x}}$)

The residual r is minimized when $\underline{\underline{\Sigma}}_1 \underline{\underline{V}}^H \hat{\underline{x}} = \underline{\underline{U}}_1^H \underline{b}$, i.e.

$$\hat{\underline{x}} = \underline{\underline{V}} \underline{\underline{\Sigma}}_1^{-1} \underline{\underline{U}}_1^H \underline{b}$$

Both $\underline{\underline{\Sigma}}_1$ and $\underline{\underline{V}}$ are full rank, so the least squares soln is unique. We could add "zero-padding" to $\underline{\underline{V}}, \underline{\underline{\Sigma}}_1$,

$$\hat{\underline{x}} = \underline{\underline{V}} \underline{\underline{\Sigma}}^+ \underline{\underline{U}}^H \underline{b}, \quad \text{where } \underline{\underline{\Sigma}}^+ = \begin{bmatrix} 1/\sigma_1 & & \\ & \ddots & \\ & & 1/\sigma_n \\ 0 & \cdots & 0 \end{bmatrix} \text{ is the pseudo inverse of } \underline{\underline{\Sigma}}$$

- We can redefine the least squares expression by expanding $\hat{\underline{x}} = \underline{\underline{A}}^+ \underline{b}$, $\underline{\underline{A}}^+ = (\underline{\underline{A}}^H \underline{\underline{A}})^{-1} \underline{\underline{A}}^H$, $\underline{\underline{A}} = \underline{\underline{U}} \underline{\underline{\Sigma}} \underline{\underline{V}}^H$

$$\underline{\underline{A}}^+ = (\underline{\underline{A}}^H \underline{\underline{A}})^{-1} \underline{\underline{A}}^H = (\underline{\underline{V}} \underline{\underline{\Sigma}}^H \underline{\underline{U}}^H \underline{\underline{U}} \underline{\underline{\Sigma}} \underline{\underline{V}}^H)^{-1} \underline{\underline{V}} \underline{\underline{\Sigma}}^H \underline{\underline{U}}^H = \underline{\underline{V}} (\underline{\underline{\Sigma}}^H \underline{\underline{\Sigma}})^{-1} \underline{\underline{V}}^H \underline{\underline{V}} \underline{\underline{\Sigma}}^H \underline{\underline{U}}^H$$

$$= \underline{\underline{V}} \underline{\underline{\Sigma}}^{-1} \underline{\underline{\Sigma}}^{-H} \underline{\underline{\Sigma}}^H \underline{\underline{U}}^H = \underline{\underline{V}} \underline{\underline{\Sigma}}^{-1} \underline{\underline{U}}^H$$

$$\underline{\underline{U}}_1 \underline{\underline{\Sigma}}_1 \underline{\underline{V}}^H = \underline{\underline{A}} \rightarrow \hat{\underline{x}} = \underline{\underline{A}}^+ \underline{b} = \underline{\underline{V}} \underline{\underline{\Sigma}}^{-1} \underline{\underline{U}}^H \underline{b}$$

- Consider the ℓ-2 norm of the least squares soln, squared, $\|\hat{\underline{x}}\|_2^2$

$$\|\hat{\underline{x}}\|_2^2 = \|\underline{\underline{V}} \underline{\underline{\Sigma}}^{-1} \underline{\underline{U}}^H \underline{b}\|_2^2 \leq \|\underline{\underline{V}}\|_2^2 \|\underline{\underline{\Sigma}}^{-1}\|_2^2 \|\underline{\underline{U}}^H \underline{b}\|_2^2 = \|\underline{\underline{\Sigma}}^{-1}\|_2^2 \|\underline{b}\|_2^2 = \left(\max_i \left| \frac{1}{\sigma_i} \right| \right)^2 \|\underline{b}\|_2^2$$

(largest value of the diagonal = absolute value of the last entry)

i.e. if the smallest singular value σ_{min} is small, then the least squares soln will be large and sensitive to changes in $\underline{b}$ $\rightarrow$ poor conditioning // if $\sigma_{min} = 0$, we cannot compute a unique least squares soln.

Least squares solution — rank deficient case.

- If $\underline{\underline{A}}$ is a matrix w/ zero singular values, we can partition the SVD as

$$\underline{\underline{A}} = [\underline{\underline{U}}_1 \; \underline{\underline{U}}_2] \begin{bmatrix} \underline{\underline{\Sigma}}_1 & \underline{\underline{0}} \\ \underline{\underline{0}} & \underline{\underline{0}} \end{bmatrix} \underline{\underline{V}}^H = \underline{\underline{U}}_1 \underline{\underline{\Sigma}}_1 \underline{\underline{V}}^H$$

(non-zero singular values.)

where $\underline{\underline{\Sigma}}_1$ has size $r \times r$, r is the rank of the matrix $\underline{\underline{A}}$.

* Note the columns of $\underline{\underline{U}}$ are orthonormal $\rightarrow \underline{\underline{U}}_1^H \underline{\underline{U}}_1 = \underline{\underline{I}}_{r \times r}$ and $\underline{\underline{U}}_2^H \underline{\underline{U}}_1 = \underline{\underline{0}}_{(m-r) \times r}$

- A least squares soln minimizes $r = \|\underline{b} - \underline{\underline{A}}\hat{\underline{x}}\|_2^2$

$$r = \|\underline{b} - \underline{\underline{A}}\hat{\underline{x}}\|_2^2 = \left\| \begin{bmatrix} \underline{\underline{U}}_1^H \\ \underline{\underline{U}}_2^H \end{bmatrix} \right\|_2^2 \left\| \begin{bmatrix} \underline{\underline{U}}_1 \underline{\underline{\Sigma}}_1 \underline{\underline{V}}^H \hat{\underline{x}} - \underline{b} \end{bmatrix} \right\|_2^2 = \left\| \begin{bmatrix} \underline{\underline{\Sigma}}_1 \underline{\underline{V}}^H \hat{\underline{x}} - \underline{\underline{U}}_1^H \underline{b} \\ -\underline{\underline{U}}_2^H \underline{b} \end{bmatrix} \right\|_2^2 = \left\| \begin{bmatrix} \underline{\underline{\Sigma}}_1 \underline{\underline{V}}^H \hat{\underline{x}} - \underline{\underline{U}}_1^H \underline{b} \\ \underline{\underline{U}}_2^H \underline{b} \end{bmatrix} \right\|_2^2 = \left\| \underline{\underline{U}}_1^H \underline{b} - \underline{\underline{\Sigma}}_1 \underline{\underline{V}}^H \hat{\underline{x}} \right\|_2^2 + \left\| \underline{\underline{U}}_2^H \underline{b} \right\|_2^2$$

(the second term is independent of $\hat{\underline{x}}$)

The residual r is minimized when $\underline{\underline{\Sigma}}_1 \underline{\underline{V}}^H \hat{\underline{x}} = \underline{\underline{U}}_1^H \underline{b}$, i.e.

$$\hat{\underline{x}} = \underline{\underline{V}} \underline{\underline{\Sigma}}_1^{-1} \underline{\underline{U}}_1^H \underline{b}$$

- Note $\underline{\underline{V}}_1$ is only a part of $\underline{\underline{V}}$ $\rightarrow$ has a non-trivial null-space, specifically $\underline{\underline{V}}_1^H (\underline{\underline{V}}_2 \underline{z}) = \underline{0}$ for all $\underline{z} \in \mathbb{C}^{n-r}$

since the columns of $\underline{\underline{V}}_1$ and $\underline{\underline{V}}_2$ are orthogonal, we also have infinitely many soln satisfying $\underline{\underline{U}}_1^H \underline{b} = \underline{\underline{V}}_1^H \underline{\underline{V}} \underline{\underline{\Sigma}}$

$$\hat{\underline{x}} = \underline{\underline{V}}_2 \underline{z} + \underline{\underline{U}} \underline{\underline{\Sigma}}_1^{-1} \underline{\underline{U}}_1^H \underline{b}$$

Taking the norm and noting that $\underline{\underline{V}}_1$ and $\underline{\underline{V}}_2$ are orthogonal wrt each other,

$$\|\hat{\underline{x}}\|_2^2 = \|\underline{\underline{V}}_2 \underline{z} + \underline{\underline{V}}_1 \underline{\underline{\Sigma}}_1^{-1} \underline{\underline{U}}_1^H \underline{b}\|_2^2 = \|\underline{\underline{V}}_2 \underline{z}\|_2^2 + \|\underline{\underline{V}}_1 \underline{\underline{\Sigma}}_1^{-1} \underline{\underline{U}}_1^H \underline{b}\|_2^2$$

($\|\underline{\underline{V}}_2 \underline{z}\|_2^2 \geq 0$)

$\therefore \hat{\underline{x}} = \underline{\underline{V}}_1 \underline{\underline{\Sigma}}_1^{-1} \underline{\underline{U}}_1^H \underline{b}$ is the minimizer to the least squares problem w/ minimal ℓ-2 norm

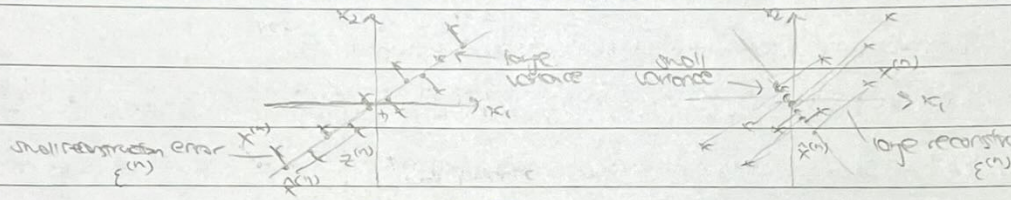
Fitting points and singular systems.

- To solve for the coefficients of the interpolating polynomial, we need to solve the matrix eqn. $\underline{\underline{A}}\underline{c} = \underline{f}$.
  If the data pts happen to lie on a line, then $\underline{\underline{A}}$ will be singular.
- When $\underline{\underline{A}}$ is singular, $\underline{\underline{A}}^T\underline{\underline{A}}$ is also singular, i.e. $(\underline{\underline{A}}^T\underline{\underline{A}})^{-1}$ doesn't exist → cannot find the least squares soln by solving $\underline{\underline{A}}^T\underline{\underline{A}}\underline{c} = \underline{\underline{A}}^T\underline{f}$.
- To find the least squares soln, we perform SVD of $\underline{\underline{A}}$ and compute $\underline{c} = \underline{\underline{V}}\,\underline{\underline{\Sigma}}^{+}\underline{\underline{U}}^T\underline{f}$
- If the matrix $\underline{\underline{A}}$ is near singular, we compute the SVD, ignore singular values less than the noise level, then compute $\underline{c} = \underline{\underline{V}}\,\underline{\underline{\Sigma}}^{+}\underline{\underline{U}}^T\underline{f}$ as usual. (unstable if we do not ignore those singular values).

Principal component analysis (PCA)

Principal component analysis (PCA).

- PCA is a dimension reduction technique. It finds new, uncorrelated variables from a dataset.
- The "new" variables are the dir. along which the variance of the sample is greatest / the reconstruction error is smallest. They can be defined as linear combination of the original variables. (both interpretations are equivalent)



- Consider projecting from $\underline{x}^{(n)} \in \mathbb{R}^D$ to $\underline{z}^{(n)} \in \mathbb{R}^M$, w/ $M < D$. Choose M projection vectors $\underline{w}_1 \dots \underline{w}_M \in \mathbb{R}^D$, s.t. they are unit length $\|\underline{w}_i\| = 1$ and orthogonal, $\underline{w}_i^T\underline{w}_j = 0 \ \forall i \neq j$. Also $\underline{x}^{(1)} \dots \underline{x}^{(N)}$ have zero mean.
  The projection of the nth item $\underline{x}^{(n)}$ onto the Mth dimension is $z_M^{(n)} = \underline{w}_M^T\underline{x}^{(n)}$, so in general, $\underline{x}^{(n)}$ is mapped to

$$\underline{z}^{(n)} = \begin{bmatrix} z_1^{(n)} \\ \vdots \\ z_M^{(n)} \end{bmatrix} = \begin{bmatrix} \underline{w}_1^T\underline{x}^{(n)} \\ \vdots \\ \underline{w}_M^T\underline{x}^{(n)} \end{bmatrix} = \begin{bmatrix} \leftarrow \underline{w}_1^T \rightarrow \\ \vdots \\ \leftarrow \underline{w}_M^T \rightarrow \end{bmatrix} \begin{bmatrix} x_1^{(n)} \\ \vdots \\ x_D^{(n)} \end{bmatrix} = \underline{\underline{W}}^T\underline{x}^{(n)}$$

  For construction, we have $\hat{\underline{x}}^{(n)} = \underline{\underline{W}}\,\underline{z}^{(n)}$ (normalizing the means makes derivation easier.)
- The (j,k)th entry of the sample covariance matrix is given by $s_{jk} = \frac{1}{N-1}\sum_{i=1}^{N} x_{ij}x_{ik}$, given $\bar{x}_i = 0 \ \forall i$
  In matrix form, the sample covariance matrix is given by $\underline{\underline{S}} = \frac{1}{N-1}\underline{\underline{X}}^T\underline{\underline{X}}$. for $\underline{\underline{X}} = \begin{bmatrix} \leftarrow \underline{x}^{(1)T} \rightarrow \\ \vdots \\ \leftarrow \underline{x}^{(N)T} \rightarrow \end{bmatrix}$
- The sample variance of direction w_1, $\hat{\sigma}_{z_1}^2$ is given by (note $\underline{\underline{S}} = \frac{1}{N-1}\underline{\underline{X}}^T\underline{\underline{X}} = \frac{1}{N-1}\sum_{n=1}^{N}\underline{x}^{(n)}\underline{x}^{(n)T}$)

$$\hat{\sigma}_{z_1}^2 = \frac{1}{N}\sum_{n=1}^{N}(z_1^{(n)} - \bar{z}_1^{\,0})^2 = \frac{1}{N}\sum_{n=1}^{N}(\underline{x}^{(n)T}\underline{w}_1)^2 = \frac{1}{N}\sum_{n=1}^{N}\underline{w}_1^T\underline{x}^{(n)}\underline{x}^{(n)T}\underline{w}_1 = \underline{w}_1^T\left(\frac{1}{N}\sum_{n=1}^{N}\underline{x}^{(n)}\underline{x}^{(n)T}\right)\underline{w}_1 = \underline{w}_1^T\underline{\underline{S}}\,\underline{w}_1$$

To max. this quantity, we choose $\underline{w}_1$ to be the eigenvector of $\underline{\underline{S}} = \frac{1}{N-1}\underline{\underline{X}}^T\underline{\underline{X}}$ that corresponds to the largest eigenvalue.
(we find $\underline{w}_2, \underline{w}_3$ by finding the eigenvector of $\underline{\underline{S}}$ that corresponds to the second/third largest eigenvalue)

- Note that the eigenvalues and eigenvectors of $\underline{\underline{X}}^T\underline{\underline{X}}$ can be computed using the SVD of $\underline{\underline{X}}$:
  ↳ eigenvalues of $\underline{\underline{X}}^T\underline{\underline{X}}$ are the squared singular values of $\underline{\underline{X}}$ (diagonals of $\underline{\underline{\Sigma}}$).
  ↳ eigenvectors of $\underline{\underline{X}}^T\underline{\underline{X}}$ are the columns of the unitary matrix $\underline{\underline{V}}$

Stochastic processes

Finite-space Markov chains

Markov chains

- A stochastic process $X_0, X_1 \dots$ is a Markov chain iff for all times $i \geq 1$

$$P(X_{i+1} \mid X_0 = j_0, X_1 = j_1 \dots X_i = j_i) = P(X_{i+1} \mid X_i = j_i)$$

where each time-indexed RV $X_0, X_1 \dots$ takes a value from the state space S ($j_i \in S \ \forall i$)

- A Markov chain can be described by the transition matrix $\underline{\underline{P}}$, where the (j,k)th element satisfies

$$P_{j,k} = P[\text{state } j \to \text{state } k] \qquad \sum_k P_{j,k} = 1$$

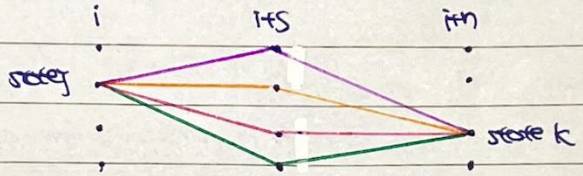
- Denoting the distribution of states at time n by $\underline{x}^{(n)}$ and the initial distribution by $\underline{x}^{(0)}$, we have

$$\underline{x}^{(n+1)} = \underline{x}^{(n)} \underline{\underline{P}} \quad \to \quad \underline{x}^{(n)} = \underline{\underline{P}}^n \underline{x}^{(0)}$$

i.e. we can get the n-step transition matrix by raising the 1-step transition matrix to the power n

- Using the notation $(\underline{\underline{P}}^n)_{j,k} = P^n_{jk}$, we can derive the Chapman-Kolmogorov eqn as follows:

$$P[X_{i+n} = k \mid X_i = j] = (\underline{\underline{P}}^n)_{j,k} = (\underline{\underline{P}}^s \underline{\underline{P}}^{n-s})_{j,k} = \sum_l P^s_{j,l} P^{n-s}_{l,k} = \sum_l P[X_{i+s} = l \mid X_i = j] \, P[X_{i+n} = k \mid X_{i+s} = l]$$



Stationary distributions and limiting distributions

- For a Markov process w/ transition matrix $\underline{\underline{P}}$, a stationary distribution is any distribution $\underline{x}^{(\infty)}$ s.t.

$$\underline{x}^{(\infty)} \underline{\underline{P}} = \underline{x}^{(\infty)}$$

- A limiting distribution is one which for any initial distribution $\underline{x}^{(0)}$ satisfies

$$\underline{x}^{(0)} \underline{\underline{P}}^n = \underline{x}^{(\infty)} \qquad \text{as} \quad n \to \infty$$

(A limiting distribution is the unique stationary position of a Markov process).

Transition matrix eigenvalues

- For a $n \times n$ transition matrix $\underline{\underline{P}}$, (i) $\lambda = 1$ is an eigenvalue, (ii) all eigenvalues satisfy $|\lambda| \leq 1$.

↳ (i) $\lambda = 1$ is an eigenvalue.

$$\underline{\underline{P}} - \underline{\underline{I}} = \begin{bmatrix} P_{1,1}-1 & P_{1,2} & \cdots & P_{1,N} \\ P_{2,1} & P_{2,2}-1 & \cdots & P_{2,N} \\ \vdots & \vdots & \ddots & \vdots \\ P_{N,1} & P_{N,2} & \cdots & P_{N,N}-1 \end{bmatrix}$$

Sum of each row is $1-1=0$

→ summing first $N-1$ col yields $-$ve last col

$\therefore \underline{\underline{P}} - \underline{\underline{I}}$ is not full rank → $\det(\underline{\underline{P}} - \underline{\underline{I}}) = 0$ → $\lambda = 1$ is an eigenvalue.

↳ (ii) All eigenvalues satisfy $|\lambda| \leq 1$

Select largest magnitude element of eigenvector v_k, i.e. $|v_k| \geq |v_i| \ \forall i$

$$|\lambda| |v_k| = |\lambda v_k| = |(\lambda \underline{v})_k| = |(\underline{\underline{P}} \underline{v})_k| = |P_{k,1} v_1 + \dots + P_{k,N} v_N| \leq P_{k,1} |v_1| + \dots + P_{k,N} |v_N|$$

$$\leq P_{k,1} |v_k| + \dots + P_{k,N} |v_k| = (P_{k,1} + \dots + P_{k,N}) |v_k| = |v_k| \quad \to \quad |\lambda| \leq 1$$

* We can find the stationary distribution by finding the left eigenvector corr. to $\lambda = 1$ $\ \underline{v} \underline{\underline{P}} = \underline{v}$.

Irreducible, recurrence and transience.

- Two states of a Markov chain j and k communicate iff there exist integers m, n s.t.

$$p^m_{j,k} > 0 \quad \text{and} \quad p^n_{k,j} > 0$$

i.e. there must be a route from state j to k and k to j in m / n steps

best determine by drawing the state diagram

- If all states in a Markov chain communicate w/ each other, then the Markov chain is irreducible.
- Let S denote the set of all possible states of a Markov chain. A subset $\tilde{S} \subseteq S$ is a recurrent set if (i) all pairs of states in $\tilde{S}$ communicate (ii) if $j \in \tilde{S}, k \notin \tilde{S}$, then $p^i_{jk} = 0$ for all i

i.e. it is possible to move btwn all states in a current set, but cannot move from a member of a current set to not a member.

- If a state belongs to a recurrent set, then it is recurrent, o/w it is transient.
  - ↳ If state k is recurrent, then $P[\text{return to state } k \mid X_0 = k] = 1$
  - ↳ If we start in a transient state, then we will leave that state and never return to it.

Periodicity

- The period of a state k of a Markov chain is the greatest common divisor of the set

$$\{ i > 0, \; p^i_{k,k} > 0 \}$$

(eg. $\{18, 24\} \rightarrow 18 = 2 \times 3^2$, $24 = 2^3 \times 3$ so $\text{GCD}(18, 24) = 2 \times 3 = 6$)

- A state k is aperiodic if it has period 1. This is true if for some time i

$$p^i_{k,k} > 0 \quad \text{and} \quad p^{i+1}_{k,k} > 0$$

- A Markov chain is aperiodic if all states are aperiodic.
- If the Markov chain is irreducible, all the states have the same period.

(All states in a recurrent set have the same period)

- For an irreducible Markov chain w/ period δ, there will be δ eigenvalues of P w/ magnitude 1 (δ roots of unity)

Regular ergodicity

Every regular MC is ergodic but an ergodic MC may not be regular

eg. $P = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$

- A Markov chain is regular if for some integer n, all entries of P^n are positive.

An ergodic Markov chain is equivalent to an irreducible Markov chain.

- If a Markov chain is irreducible and aperiodic, then it is regular ergodic.
  - ↳ A regular ergodic Markov chain has a limiting distribution which puts the mass on every element of the state space S
  - ↳ If $X_0, X_1 \ldots$ is a regular ergodic Markov chain w/ limiting distribution Π, then for any function $f(x)$, as $N \to \infty$, $\frac{1}{N} \sum_{i=1}^{N} f(X_i) \to \sum_{k \in S} \pi_k f(k)$

Detailed balance

- A transition matrix $\underline{P}$ and a distribution $\underline{\pi}$ are in detailed balance if

$$\pi_j P_{jk} = \pi_k P_{kj} \quad \forall j,k$$

Consider this distribution after one step, $\underline{\tilde{\pi}} = \underline{\pi}\underline{P}$.

$$\tilde{\pi}_k = \sum_j \pi_j P_{jk} = \sum_j \pi_k P_{kj} = \pi_k \left(\sum_j P_{kj}\right) = \pi_k \rightarrow \underline{\tilde{\pi}} = \underline{\pi}$$

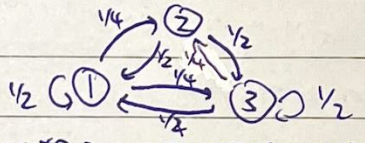
$\therefore \underline{\pi}$ is a stationary distribution of $\underline{P}$.

* If $\underline{\pi}$ satisfies detailed balance wrt $\underline{P}$, $\underline{\pi}$ is a stationary distribution for the transition matrix $\underline{P}$, but a stationary distribution of $\underline{P}$ does not necessarily satisfy detailed balance.

Waiting time

- To calculate the expected waiting time to reach a certain state k from state j, q_j, we set up an expression q_j for each of the N states and solve.
- e.g. Consider a Markov chain w/ transition matrix $\underline{P}$, $(N=3)$

$$\underline{P} = \begin{bmatrix} 1/2 & 1/4 & 1/4 \\ 1/2 & 0 & 1/2 \\ 1/4 & 1/4 & 1/2 \end{bmatrix}$$



Find the average wait time to transition to state 2, given we start at state 2.

$$\begin{cases} q_1 = \frac{1}{2}(1+q_1) + \frac{1}{4}(1) + \frac{1}{4}(1+q_3) \\ q_2 = \frac{1}{2}(1+q_1) + \frac{1}{2}(1+q_3) \\ q_3 = \frac{1}{4}(1+q_1) + \frac{1}{4}(1) + \frac{1}{2}(1+q_3) \end{cases} \rightarrow q_1 = 4,\ q_2 = 5,\ q_3 = 4$$

$\therefore$ average wait time req. is 5 steps.

Continuous state space systems

Birth process

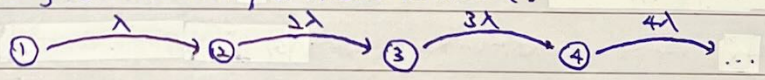
- Let $n(t)$ be the no. of cells at time t, $n(0) = n_0$ and λ be the birth rate for a cell per unit time.

$n(t)$ describes the state of the system $\rightarrow$ Markovian. For a small time step Δt,

$$n(t+\Delta t) = n(t) + n(t)\lambda\Delta t \xrightarrow{\Delta t \to 0} \frac{dn(t)}{dt} = \lambda n(t) \longrightarrow n(t) = n_0 \exp(\lambda t)$$

* $n(t)$ gives the expected value of the no. of cells $\rightarrow$ consider probabilistic approach for distribution of n.

- Let $P_n(t) = P[n(t) = n]$ be the probability of n cells at time t.



For a small time step Δt, $\quad P_n(t+\Delta t) = P_n(t)(1 - n\lambda\Delta t) + P_{n-1}(t)((n-1)\lambda\Delta t)$

As $\Delta t \to 0$, $\quad \frac{dP_n(t)}{dt} = -n\lambda P_n(t) + (n-1)\lambda P_{n-1}(t)$

- We can represent the system in terms of the transition rate matrix $\underline{Q}$

$$\frac{d\underline{x}(t)}{dt} = \underline{x}(t)\underline{Q}$$

where $\underline{x}(t) = [P_1(t) \dots P_n(t) \dots]$, $\sum_{n=1}^{\infty} P_n(t) = 1$ and $\underline{Q} = \begin{bmatrix} -\lambda & \lambda & 0 & 0 & \cdots \\ 0 & -2\lambda & 2\lambda & 0 & \cdots \\ 0 & 0 & -3\lambda & 3\lambda & \cdots \\ \vdots & \vdots & \vdots & \vdots & \ddots \end{bmatrix}$ rows sum to zero

- If can be shown that the solution to the birth process is

$$P_n(t) = \binom{n-1}{n_0-1} \exp(-\lambda n_0 t)(1-\exp(-\lambda t))^{n-n_0}$$

- If the birth process did reach a steady state, then

$$\frac{d\underline{x}(\infty)}{dt} = \underline{x}(\infty)\underline{\underline{Q}} = \underline{0}$$

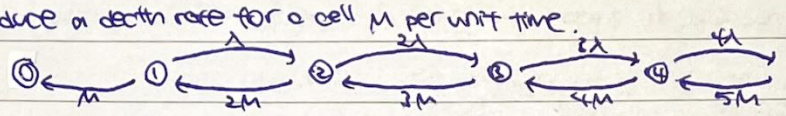
$$\rightarrow \begin{cases} 0 = -\lambda x_1(\infty) \\ 0 = \lambda x_1(\infty) - 2\lambda x_2(\infty) \\ \vdots \qquad\qquad \vdots \\ 0 = (n-1)x_{n-1}(\infty) - n\lambda x_n(\infty) \end{cases}$$

the set of eqns are only satisfied when $x_i(\infty) = 0$ for all i → not a valid PDF

∴ there is no steady state distribution (the no. of cells keep growing)

Birth-death process

- Now we introduce a death rate for a cell μ per unit time.



For a small time step Δt, $P_n(t+\Delta t) = P_n(t)(1-n\lambda\Delta t - n\mu\Delta t) + P_{n-1}(t)((n-1)\lambda\Delta t) + P_{n+1}(t)((n+1)\mu\Delta t)$

As $\Delta t \to 0$, $\frac{dP_n(t)}{dt} = -n(\lambda+\mu)P_n(t) + (n-1)\lambda P_{n-1}(t) + (n+1)\mu P_{n+1}(t)$

- We can represent the system in terms of the transition rate matrix $\underline{\underline{Q}}$.

$$\frac{d\underline{x}(t)}{dt} = \underline{x}(t)\underline{\underline{Q}}$$

where $\underline{x}(t) = [P_1(t) \dots P_n(t) \dots]$, $\sum_{n=1}^{\infty} P_n(t) = 1$ and $\underline{\underline{Q}} = \begin{bmatrix} 0 & 0 & 0 & 0 & \cdots \\ \mu & -(\mu+\lambda) & \lambda & 0 & \cdots \\ 0 & 2\mu & -2(\mu+\lambda) & 2\lambda & \cdots \\ \vdots & \vdots & \vdots & \vdots & \ddots \end{bmatrix}$

- For steady state, we req.

$$\frac{d\underline{x}(\infty)}{dt} = \underline{x}(\infty)\underline{\underline{Q}} = \underline{0} \quad \rightarrow \quad -n(\lambda+\mu)P_n + (n-1)\lambda P_{n-1} + (n+1)\mu P_{n+1} = 0$$

It can be shown that $\underline{x}(\infty) = [1\ 0\ 0 \cdots]$ → all states except 0 are transient.

* If we λ_i, μ_i be the birth/death rate per unit time for state i respectively, then it can be shown using recursion that for $i > 0$, $P_i(\infty) = \left(\frac{\prod_{j=0}^{i-1}\lambda_j}{\prod_{j=1}^{i}\mu_j}\right)P_0(\infty)$

Random walk

- consider a random walk — at each time, we take step ± 1, where the probability of dir. of time k $S_k \in \{-1, 1\}$ is uniform, i.e. $P(S_k = 1) = P(S_k = -1) = \frac{1}{2}$.

- After n steps, the position X_n is given by

$$X_n = \sum_{k=1}^{n} S_k$$

- At each instance, the average step is 0, and step variance is 1 → by CLT, the distribution of X_N as $N \to \infty$ is a Gaussian $N(0, N)$.

- If we now take a small step δ in dir. S_k every δ seconds, the distribution of the position W_t of time $t = N\delta$ as $N \to \infty$ is a Gaussian $N(0, t) = N(0, N\delta)$
(As $\delta \to 0$, we have Brownian motion)

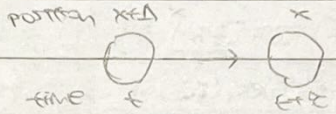
Brownian motion

- Brownian motion is the random motion of particles in a fluid resulting from collisions. since there are too many particles, we consider the particle density.
- consider a simplified system where each dir. is independent, where the overall form is obtained by simply multiplying dimensions together. → only consider x dimension
- Let the particle density at time t and position x be $f(x,t)$. We can define the state of the system since probability of the next state only depends on the current state at time t → Markov process.
- Assuming smoothness in the particle density $f(x)$ over position x and time t, we can write

$$f(x+\Delta, t) = f(x,t) + \Delta \frac{\partial f(x,t)}{\partial x} + \frac{\Delta^2}{2}\frac{\partial^2 f(x,t)}{\partial x^2} + O(\Delta^3) \qquad [1]$$

$$f(x, t+\tau) = f(x,t) + \tau \frac{\partial f(x,t)}{\partial t} + O(\tau^2) \qquad [2]$$

- The change in density at time $t+\tau$ results from particles moving to position x from time t to $t+\tau$.



Considering all initial positions $x+\Delta$, the density at time $t+\tau$ is given by

$$f(x, t+\tau) = \int_{-\infty}^{\infty} f(x+\Delta, t)\, p(-\Delta)\, d\Delta \qquad [3]$$

continuous form of Chapman-Kolmogorov

where $p(-\Delta)$ is the probability of a particle moving $-\Delta$ in time τ.

- Sub [1] into [3], and by symmetry $p(-\Delta) = p(\Delta)$,

$$f(x,t+\tau) \approx \int_{-\infty}^{\infty} \left[f(x,t) + \Delta \frac{\partial f(x,t)}{\partial x} + \frac{\Delta^2}{2}\frac{\partial^2 f(x,t)}{\partial x^2} \right] p(\Delta)\, d\Delta$$

$$= f(x,t)\int_{-\infty}^{\infty} p(\Delta)\, d\Delta + \frac{\partial f(x,t)}{\partial x}\int_{-\infty}^{\infty} \Delta p(\Delta)\, d\Delta + \frac{1}{2}\frac{\partial^2 f(x,t)}{\partial x^2}\int_{-\infty}^{\infty} \Delta^2 p(\Delta)\, d\Delta.$$

($\int p(\Delta)\,d\Delta = 1$, $\int \Delta p(\Delta)\,d\Delta = 0$)

$$= f(x,t) + \frac{1}{2}\frac{\partial^2 f(x,t)}{\partial x^2}\int_{-\infty}^{\infty} \Delta^2 p(\Delta)\, d\Delta$$

Comparing this w/ [2], we have

$$f(x,t+\tau) = f(x,t) + \tau\frac{\partial f(x,t)}{\partial t} = f(x,t) + \frac{1}{2}\frac{\partial^2 f(x,t)}{\partial x^2}\int_{-\infty}^{\infty} \Delta^2 p(\Delta)\, d\Delta$$

Fokker-Planck equation

$$\therefore \tau\frac{\partial f(x,t)}{\partial t} = \frac{1}{2}\frac{\partial^2 f(x,t)}{\partial x^2}\int_{-\infty}^{\infty} \Delta^2 p(\Delta)\, d\Delta \rightarrow \boxed{\frac{\partial f(x,t)}{\partial t} = D\frac{\partial^2 f(x,t)}{\partial x^2}} \text{ where } \boxed{D = \frac{1}{2\tau}\int_{-\infty}^{\infty} \Delta^2 p(\Delta)\, d\Delta}.$$

- The value of D can be obtained by measurement of physical properties → no need to consider τ or $p(\Delta)$.
- The final soln depends on the initial condition → particular solns.

solution for Brownian motion differential equation.

- Consider the Brownian motion DE. For a const. α,

$$\frac{\partial p(x,t)}{\partial t} = \alpha \frac{\partial^2 p(x,t)}{\partial x^2}$$

Assuming the soln can be written in the form, $p(x,t) = X(x)T(t)$,

$$X\frac{dT}{dt} = \alpha T\frac{d^2X}{dx^2} \rightarrow \frac{1}{\alpha T}\frac{dT}{dt} = \frac{1}{X}\frac{d^2X}{dx^2} = -k^2 \leftarrow \text{separation const.}$$

$\therefore X(x) = A\sin(kx) + B\cos(kx)$, $T(t) = C\exp(-\alpha k^2 t) \rightarrow p(x,t) = A(k)\exp(-\alpha k^2 t)\exp(ikx)$

The above soln is satisfied by any k, so the general soln is

$$p(x,t) = \int_{-\infty}^{\infty} A(k)\exp(-\alpha k^2 t)\exp(ikx)\,dk$$

- Consider the initial condition as $p(x,0) = \delta(x)$,

$$\delta(x) = p(x,0) = \int_{-\infty}^{\infty} A(k)\exp(ikx)\,dk \overset{\tilde{k}=-k}{=} \int_{-\infty}^{\infty} A(-\tilde{k})\exp(-i\tilde{k}x)\,d\tilde{k}$$

Noting that this is the FT of $A(-\tilde{k})$, i.e. $FT[A(-\tilde{k})] = \delta(x)$, so

$$A(-\tilde{k}) = IFT[\delta(x)] = \frac{1}{2\pi}$$

$$\therefore \boxed{p(x,t) = \frac{1}{2\pi}\int_{-\infty}^{\infty}\exp(-\alpha k^2 t)\exp(ikx)\,dk = \frac{1}{2\pi}\int_{-\infty}^{\infty}\exp(-\alpha k^2 t + ikx)\,dk}$$

complete the square in k.

The term in the exponential can be rewritten as $-\alpha k^2 t + ikx = -\left(\sqrt{\alpha t}\,k - \frac{ix}{2\sqrt{\alpha t}}\right)^2 - \frac{x^2}{4\alpha t}$

$$p(x,t) = \frac{1}{2\pi}\exp\left(-\frac{x^2}{4\alpha t}\right)\int_{-\infty}^{\infty}\exp\left[-\left(\sqrt{\alpha t}\,k - \frac{ix}{2\sqrt{\alpha t}}\right)^2\right]dk = \frac{1}{2\pi}\exp\left(-\frac{x^2}{4\alpha t}\right)\sqrt{\frac{\pi}{\alpha t}}$$

$$\boxed{p(x,t) = \frac{1}{\sqrt{4\pi\alpha t}}\exp\left(-\frac{x^2}{4\alpha t}\right) = N(x\,|\,0, 2\alpha t)}$$

Properties of a 1D Wiener process.

- The simplified Brownian motion is a 1D Wiener process. At time t, examine the Wiener process W_t.

We can consider an instance of a "path" at time t $W_t^{(i)}$ (the support of the RV x_t)

- The Wiener process $\{W_t\}_{t\geq 0}$ has the following properties:

(i) Stationarity: $\boxed{W_{t+s} - W_s \text{ is independent of } s}$, e.g. $W_t = W_{t+k} - W_k$

(ii) Independence: $\boxed{W_t - W_s \text{ is independent of } \{W_\tau\}_{\tau \leq s}}$, e.g. $W_{10} - W_5 \perp W_2$

(iii) Continuity: W_t is a continuous function w.r.t t

(iv) Gaussianity: W_t is a Gaussian w/ $\boxed{E[W_t] = 0,\ E[W_t W_s] = 2\alpha\min(t,s)}$

General continuous state-space systems

- A range of assumptions have been made in the previous derivation — some may not be valid.

- A simple extension is to include a drift term: Wiener process w/ drift.

$$\frac{\partial p(x,t)}{\partial t} = M\frac{\partial p(x,t)}{\partial x} + D\frac{\partial^2 p(x,t)}{\partial x^2}$$

- More complex models include the Ornstein-Uhlenbeck process.

$$\frac{\partial p(x,t)}{\partial t} = \frac{\partial(\beta x p(x,t))}{\partial x} + \frac{\partial^2(\beta p(x,t))}{\partial x^2}$$

where βx controls the drift and β controls the diffusion

- When $\beta = 0$, we have the Wiener process. When $\beta \neq 0$, the diffusion term can be balanced by the drift term so the particle density tends to a limiting distribution (finite variance)

Monte Carlo methods

Monte Carlo integration.

- consider a highly complicated, multi-dimensional ($\underline{x} \in \mathbb{R}^d$) integral $\int h(\underline{x}) d\underline{x}$.

  using a histogram approach (in d dimensions), we have

$$\int h(\underline{x}) d\underline{x} \approx \sum_{i=1}^{N} h(\underline{x}^{(i)}) \delta x_1 \delta x_2 \ldots \delta x_d$$

  where we have uniformly spread N histogram centres.

- The histogram approach is simple but wasteful for regions w/ very small (no) contributions to the integral.

  It also does not scale well w/ the no. of dimensions. ← to not lose accuracy in the approx, we req. n bins per dimension → n^d bins for d dimensions!

- For Monte Carlo integration, we sample from the normalised weighting function $w(\underline{x})$, $p(\underline{x})$

$$p(\underline{x}) = \frac{w(\underline{x})}{\int w(\underline{x}) d\underline{x}}$$

  This is used to model the distribution of the pts.

- Initially consider uniform distribution over a "volume" V, $p(\underline{x}) = \frac{1}{V}$. The integral becomes.

$$\int h(\underline{x}) d\underline{x} = \int \frac{h(\underline{x})}{p(\underline{x})} p(\underline{x}) d\underline{x} = E_p\left[\frac{h(\underline{x})}{p(\underline{x})}\right] \approx \frac{1}{N} \sum_{i=1}^{N} \frac{h(\underline{x}^{(i)})}{p(\underline{x}^{(i)})} = \frac{V}{N} \sum_{i=1}^{N} h(\underline{x}^{(i)})$$

  where $\underline{x}^{(i)}$ are drawn from $p(\underline{x})$.

- Selecting an appropriate "volume" for $p(\underline{x})$ avoids wasted samples → smaller variance for a fixed no. of samples.

Monte Carlo for expected values.

- consider computing the expected value of a function $f(\underline{x})$, $E[f(\underline{x})]$. The integral has the form

$$\int f(\underline{x}) p(\underline{x}) d\underline{x} = E_p[f(\underline{x})] \approx \frac{1}{N} \sum_{i=1}^{N} f(\underline{x}^{(i)})$$

  where $\underline{x}^{(i)}$ are drawn from $p(\underline{x})$.

- Denote the mean of the estimate using 1 sample w/ $\hat{M}_1$. $\hat{M}_1 = \int f(\underline{x}) p(\underline{x}) d\underline{x}$.

  The mean of the estimate using N samples, $\hat{M}_N$ is given by

$$\hat{M}_N = \frac{1}{N} \sum_{i=1}^{N} \hat{M}_1 = \hat{M}_1 \quad \leftarrow \text{unbiased.}$$

- Denote the variance of the estimate using 1 sample w/ $\hat{\sigma}_1^2$. $\hat{\sigma}_1^2 = \int (f(\underline{x}) - M)^2 p(\underline{x}) d\underline{x}$.

$$\hat{\sigma}_N^2 = \frac{1}{N^2} \sum_{i=1}^{N} \hat{\sigma}_1^2 = \frac{1}{N} \hat{\sigma}_1^2$$

- In fact, for large N, the distribution of $\frac{1}{N} \sum_{i=1}^{N} f(\underline{x}^{(i)}) \to N(\hat{M}_N, \hat{\sigma}_N^2) = N(\hat{M}_1, \frac{\hat{\sigma}_1^2}{N})$ by CLT.

  → the approach yields correct answers. (as $N \to \infty$).

Importance sampling.

- If we want to sample from another pdf $q(\underline{x})$ rather than $p(\underline{x})$ (say it is not possible to sample from $p(\underline{x})$ or we want to sample from a non-uniform distribution), we use importance sampling.
- Importance sampling can be used for computing expected values / numerical integration:

① computing expected value $E_p[f(\underline{x})]$

$$E_p[f(\underline{x})] = \int f(\underline{x}) p(\underline{x}) d\underline{x} = \int f(\underline{x}) \frac{p(\underline{x})}{q(\underline{x})} q(\underline{x}) d\underline{x} = E_q\left[f(\underline{x}) \frac{p(\underline{x})}{q(\underline{x})}\right] \approx \frac{1}{N} \sum_{i=1}^{N} f(\underline{x}^{(i)}) \frac{p(\underline{x}^{(i)})}{q(\underline{x}^{(i)})}$$

↳ the ratio $\frac{p(\underline{x}^{(i)})}{q(\underline{x}^{(i)})}$ is the importance of the drawn sample ← how much to "boost" a sample $f(\underline{x}^{(i)})$

($q(\underline{x}) > p(\underline{x})$: over-represented region ; $p(\underline{x}) > q(\underline{x})$: under-represented region)

↳ we can perform importance sampling w/o the normalised distributions. For $\underline{x}^{(i)} \sim q(\underline{x})$.

Let $p(\underline{x}) = \frac{1}{Z_p} p^*(\underline{x})$, $q(\underline{x}) = \frac{1}{Z_q} q^*(\underline{x})$, $w^{(i)} = \frac{p^*(\underline{x}^{(i)})}{q^*(\underline{x}^{(i)})}$, where Z_p, Z_q cannot be computed.

The expected value can be shown to be $V = \int f(\underline{x}) \frac{p(\underline{x})}{q(\underline{x})} q(\underline{x}) d\underline{x} \approx \frac{\frac{1}{N}\sum_{i=1}^{N} f(\underline{x}^{(i)}) w^{(i)}}{\frac{1}{N}\sum_{i=1}^{N} w^{(i)}}$

(Numerator: $\frac{1}{N}\sum_{i=1}^{N} f(\underline{x}^{(i)}) w^{(i)} \approx E_q[f(\underline{x}) w^{(i)}] = \int f(\underline{x}) \frac{p^*(\underline{x})}{q^*(\underline{x})} q(\underline{x}) d\underline{x} = \frac{Z_p}{Z_q} \int f(\underline{x}) p(\underline{x}) d\underline{x} = \frac{Z_p}{Z_q} V$

Denominator: $\frac{1}{N}\sum_{i=1}^{N} w^{(i)} \approx E_q[w^{(i)}] = \int \frac{p^*(\underline{x})}{q^*(\underline{x})} q(\underline{x}) d\underline{x} = \frac{Z_p}{Z_q} \int p(\underline{x}) d\underline{x} = \frac{Z_p}{Z_q}$)

② Numerical integration $\int h(\underline{x}) d\underline{x}$.

$$\int h(\underline{x}) d\underline{x} = \int \frac{h(\underline{x})}{q(\underline{x})} q(\underline{x}) d\underline{x} = E_q\left[\frac{h(\underline{x})}{q(\underline{x})}\right] \approx \frac{1}{N} \sum_{i=1}^{N} \frac{h(\underline{x}^{(i)})}{q(\underline{x}^{(i)})}$$

↳ The ratio $\frac{h(\underline{x}^{(i)})}{q(\underline{x}^{(i)})}$ is similar to the "importance" of the drawn sample.

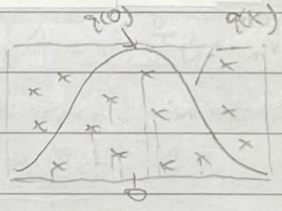
↳ we want $q(\underline{x})$ to be close to $h(\underline{x})$ → importance ≈ 1 everywhere

if $q(\underline{x}) \propto h(\underline{x})$, then $\frac{h(\underline{x}^{(i)})}{q(\underline{x}^{(i)})}$ is a const. → we can get the answer in 1 sample.

+ Using importance sampling may give a smaller variance in the estimate for a fixed no. of samples.

Generating random samples — rejection sampling.

- We want to generate random samples from the distribution $q(x)$, $x \in \mathbb{R}$
- In rejection sampling, we enclose (most of) the distribution $q(x)$ in a 2D box



assume we can generate uniform RVs

Drawing samples uniformly from the 2D box, we accept those under the curve (take x-values) and reject those above the line

- The fraction of samples accepted is given by

$$\text{fraction of samples accepted} = \frac{\text{area under curve}}{\text{area of box}}$$ ← slightly under 1.

For higher dimensions, $\underline{x} \in \mathbb{R}^d$, the fraction of samples accepted would be raised to the power of d → highly wasteful (curse of dimensionality).

Generating random samples — Metropolis-Hastings algorithm.

- We want to generate random samples from the distribution $p(\underline{x})$, $\underline{x} \in \mathbb{R}^d$.
- The Metropolis-Hastings algorithm is as follows:
  - Given the current sample $\underline{x}^{(i)}$, generate another sample $\underline{x}^{(*)}$ from $p(\underline{x}^{(*)} | \underline{x}^{(i)})$

$$\underline{x}^{(*)} = \underline{x}^{(i)} + \underline{z}$$

where $\underline{z}$ is the proposal distribution (e.g. $\underline{z} \sim N(\underline{0}, \underline{\underline{I}})$)

  - Accept the sample, i.e. $\underline{x}^{(i+1)} = \underline{x}^{(*)}$ w/ probability $\alpha = \min\left\{ \frac{p(\underline{x}^{(*)}) \, p(\underline{x}^{(i)} | \underline{x}^{(*)})}{p(\underline{x}^{(i)}) \, p(\underline{x}^{(*)} | \underline{x}^{(i)})}, 1 \right\}$. ($\underline{x}^{(*)} \to \underline{x}^{(i)}$, $\underline{x}^{(i)} \to \underline{x}^{(*)}$)

otherwise reject the sample, i.e. $\underline{x}^{(i+1)} = \underline{x}^{(i)}$

- It can be shown that the distribution of $\{\underline{x}^{(n)}\}$ as $n \to \infty$ has the same distribution as $p(\underline{x})$.
- Overall samples are drawn twice
  - initial samples depend on $\underline{x}^{(0)}$ → ignore the initial samples (burn-in phase).
  - samples correlated w/ neighbouring samples → take every nth sample (thinning)
- We need to decide on the proposal distribution $\underline{z}$. If $\underline{z}$ is symmetric, i.e. $p(\underline{x}^{(i)} | \underline{x}^{(*)}) = p(\underline{x}^{(*)} | \underline{x}^{(i)})$, then the probability of accepting the sample is $\alpha = \min\left\{ \frac{p(\underline{x}^{(*)})}{p(\underline{x}^{(i)})}, 1 \right\}$. ↑ true for $\underline{z} \sim N(\underline{0}, \underline{\underline{I}})$.

Markov chain Monte Carlo (MCMC).

- A MCMC approach is based on the Metropolis-Hastings algorithm to generate samples from a process w/ a limiting distribution $\underline{\pi}$.
- Consider a proposal function — a transition matrix $\underline{\underline{R}}$, where $r_{j,j} = 0$ and $r_{j,k} > 0$ $(j \neq k)$.
(The true transition matrix is not known).
- The MCMC algorithm is as follows:
  - Given the current state x_i, select $\hat{x}_{i+1}$ by sampling from the x_i-th row of $\underline{\underline{R}}$.
  - Accept the sample, i.e. $x_{i+1} = \hat{x}_{i+1}$ w/ probability $\alpha = \min\left\{ \frac{\pi_{\hat{x}_{i+1}} r_{\hat{x}_{i+1}, x_i}}{\pi_{x_i} r_{x_i, \hat{x}_{i+1}}}, 1 \right\}$. ($\hat{x}_{i+1} \to x_{i+1}$, $x_i \to \hat{x}_{i+1}$)

otherwise reject the sample, i.e. $x_{i+1} = x_i$

- The process is equivalent to a transition matrix $\underline{\underline{R}}'$ where,

$$r'_{j,k} = r_{j,k} \min\left\{ \frac{\pi_k r_{k,j}}{\pi_j r_{j,k}}, 1 \right\} \text{ if } j \neq k \qquad r'_{j,j} = 1 - \sum_{k \neq j} r_{j,k} \min\left\{ \frac{\pi_k r_{k,j}}{\pi_j r_{j,k}}, 1 \right\}$$

- We can show that $\underline{\underline{R}}'$ and $\underline{\pi}$ are in detailed balance

$$\pi_j r'_{j,k} = \pi_j r_{j,k} \min\left\{ \frac{\pi_k r_{k,j}}{\pi_j r_{j,k}}, 1 \right\} = \min\{ \pi_k r_{k,j}, \pi_j r_{j,k} \} = \pi_k r_{k,j} \min\left\{ 1, \frac{\pi_j r_{j,k}}{\pi_k r_{k,j}} \right\} = \pi_k r'_{k,j}$$

$\underline{\underline{R}}'$ and $\underline{\pi}$ are in detailed balance → $\underline{\pi}$ is a stationary distribution for the transition matrix $\underline{\underline{R}}'$.

By construction, $\underline{\underline{R}}$ is irreducible and aperiodic → regular ergodic → $\underline{\pi}$ is the limiting distribution.

Optimisation

The optimisation problem

- The goal of optimisation is to find the independent variables that min./max a given quantity, possibly subject to some restrictions on the allowed range of parameters.
- The quantity to be max/min is the objective / cost / utility / loss function, denoted by $f(\underline{x}): \mathbb{R}^N \to \mathbb{R}$.
- The parameters that can be changed are the control / decision variables
  The restrictions on the allowed parameter values are the constraints.
- Mathematically, the optimisation problem is

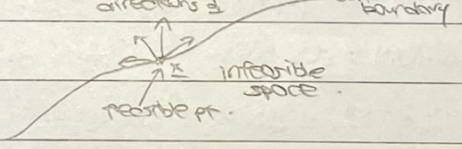
  minimise $f(\underline{x})$, $\underline{x} = [x_1, x_2 \dots x_N]^T$ subject to $h_i(\underline{x}) = 0$, $i = 1,2 \dots M$ and $g_j(\underline{x}) \leq 0$, $j = 1,2 \dots n$

  (inequality constraints on single variables are bounds, eg. $x_{i,min} \leq x_i \leq x_{i,max}$.)

  where $f(\underline{x})$ is the objective function, $\underline{x}$ is the col. vector of N control variables,
  $\{h_i(x), g_j(x)\}$ is the set of constraint functions.

+ A min. $\underline{x}$ of the function $-f(\underline{x})$ is a max. $\underline{x}$ of the function $f(\underline{x})$ → max. problems can be cast as min. problems

- Common forms of the objective function $f(\underline{x})$ include:
  - ↳ Linear: $f(\underline{x}) = \underline{b}^T \underline{x} + c$
  - ↳ Quadratic: $f(\underline{x}) = \underline{x}^T \underline{\underline{A}} \underline{x} + \underline{b}^T \underline{x} + c$
  - ↳ Nonlinear (not linear/quadratic): (eg) $f(\underline{x}) = \exp(x_1) + \underline{x}^T \underline{\underline{A}} \underline{x} + \underline{b}^T \underline{x} + c$. (more difficult to solve)
- When min. $f(\underline{x})$ subject to constraints, S is the feasible region, and any $\underline{x} \in S$ is a feasible soln.
  For an unconstrained problem, S is infinitely large.
- At a feasible pt, $\underline{x}$, a direction $\underline{d}$ is a feasible direction if an arbitrarily small move from $\underline{x}$ in direction $\underline{d}$ remains feasible.



Stationary points and types of minima.

- If $f(\underline{x})$ is smooth, s.t. $\nabla f(\underline{x})$ exists, then $\underline{x}^*$ is a stationary pt. of f if

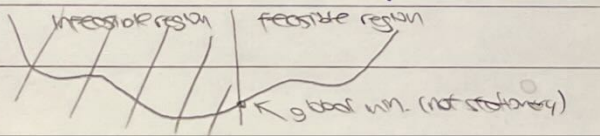
$$\nabla f(\underline{x}^*) = 0$$

+ minima, maxima and saddle pts are stationary pts.

- Types of minima includes the following:
  - ↳ Global min. $f(\underline{x}^*) \leq f(\underline{y}) \ \forall \underline{y} \in S$
  - ↳ Weak local min. $f(\underline{x}^*) \leq f(\underline{y}) \ \forall \underline{y} = \underline{x}^* + \varepsilon \underline{d} \in S, \ \underline{y} \neq \underline{x}^*$
  - ↳ Strong global min. $f(\underline{x}^*) < f(\underline{y}) \ \forall \underline{y} \in S, \ \underline{y} \neq \underline{x}^*$
  - ↳ Strong local min. $f(\underline{x}^*) < f(\underline{y}) \ \forall \underline{y} = \underline{x}^* + \varepsilon \underline{d} \in S$

+ Local max/min. are local extrema, and refer to interior local max./min, unless o/w specified.

- When subject to constraints, a global min. might not be a stationary pt.



Necessary and sufficient conditions for a local minimum

- A necessary condition for $\underline{x}^*$ to be a local min. of $f(\underline{x})$ in S is

$$\nabla f(\underline{x}^*) \cdot \underline{d} \geq 0 \quad \text{for all feasible directions } \underline{d}$$

- If $\underline{x}^*$ is an interior pt., i.e. it is not at the boundary of the feasible region, then all dir. are feasible → $\underline{x}^*$ must be a stationary pt. when $\nabla f(\underline{x}^*) = 0$

(the condition $\nabla f(\underline{x}^*) = 0$ is necessary but not sufficient for $\underline{x}^*$ to be a min. as the above condition holds for max / saddle pts)

- For the univariate case ($x \in \mathbb{R}$), the Taylor expansion about x^* is

$$f(x) = f(x^*) + (x-x^*)f'(x^*) + \tfrac{1}{2}(x-x^*)^2 f''(x^*) + \text{H.O.T.}$$

$$f(x) - f(x^*) = (x-x^*)f'(x^*) + \tfrac{1}{2}(x-x^*)^2 f''(x^*) + \text{H.O.T.}$$

If x^* is an interior stationary pt, $f'(x^*) = 0$, then we have a strong local min. if

$$f(x) - f(x^*) \approx \tfrac{1}{2}(x-x^*)^2 f''(x^*) > 0 \quad \rightarrow \quad f''(x^*) > 0$$

- At a stationary pt x^*, $f''(x^*) > 0$ is a sufficient condition of strong local min.
  $f''(x^*) \geq 0$ is a necessary condition of strong local min.

(If $f''(x^*) = 0$, we need to analyse the higher order terms)

- For the multivariate case ($\underline{x} \in \mathbb{R}^N$), the Taylor expansion about $\underline{x}^*$ is

$$f(\underline{x}) = f(\underline{x}^*) + (\underline{x}-\underline{x}^*)^T \nabla f(\underline{x}^*) + \tfrac{1}{2}(\underline{x}-\underline{x}^*)^T \underline{\underline{H}}(\underline{x}^*)(\underline{x}-\underline{x}^*) + \text{H.O.T.}$$

$$f(\underline{x}) - f(\underline{x}^*) = (\underline{x}-\underline{x}^*)^T \nabla f(\underline{x}^*) + \tfrac{1}{2}(\underline{x}-\underline{x}^*)^T \underline{\underline{H}}(\underline{x}^*)(\underline{x}-\underline{x}^*) + \text{H.O.T.}$$

the Hessian $\underline{\underline{H}}(\underline{x})$ is symmetric by def'n.

where $\underline{\underline{H}}(\underline{x}) = \nabla(\nabla f(\underline{x})) = \nabla\left(\left[\frac{\partial f}{\partial x_1} \dots \frac{\partial f}{\partial x_N}\right]^T\right) = \begin{bmatrix} \frac{\partial^2 f}{\partial x_1^2} & \cdots & \frac{\partial^2 f}{\partial x_1 \partial x_N} \\ \vdots & & \vdots \\ \frac{\partial^2 f}{\partial x_N \partial x_1} & \cdots & \frac{\partial^2 f}{\partial x_N^2} \end{bmatrix}$ is the Hessian

If $\underline{x}^*$ is an interior stationary pt, $\nabla f(\underline{x}^*) = 0$ then we have a strong local min. if

$$f(\underline{x}) - f(\underline{x}^*) \approx \tfrac{1}{2}\underline{d}^T \underline{\underline{H}}(\underline{x}^*)\underline{d} > 0 \quad \rightarrow \quad \underline{d}^T \underline{\underline{H}}(\underline{x}^*)\underline{d} > 0, \quad \text{where } \underline{d} = \underline{x} - \underline{x}^*$$

- At a stationary pt $\underline{x}^*$, $\underline{d}^T \underline{\underline{H}}(\underline{x}^*)\underline{d} > 0$ (PD Hessian) is a sufficient condition of strong local min.
  $\underline{d}^T \underline{\underline{H}}(\underline{x}^*)\underline{d} \geq 0$ (PSD Hessian) is a necessary condition of strong local min.

- If $\underline{\underline{H}}(\underline{x})$ is PD everywhere, then $f(\underline{x})$ is strictly convex → the min. is unique and a global min.

- A matrix is PD iff all its eigenvalues are +ve. For a 2×2 matrix, it is PD if

$$H_{11} > 0 \quad \text{and} \quad \det(\underline{\underline{H}}) > 0$$

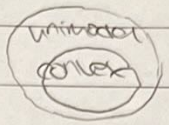
(If $\underline{\underline{H}}(\underline{x}^*) = 0$, we need to analyse the higher order terms)

Unimodal and convex functions

- A function is unimodal if it has a single extremum. It is strictly unimodal if the function strictly increases/decreases.
- A function is convex if its graph at any pt y is never below the tangent of any other pt x.

$$f(y) \geq f(x) + \nabla f(x)^T (y-x)$$

(If the function is strictly convex, we replace $\geq$ w/ $>$).



- convex functions are unimodal but not all unimodal functions are convex
- The local min. of a unimodal/convex function is unique → local min. is the global min.

Unconstrained optimization — search methods for univariate functions

General search algorithm

- The general search algorithm is as follows:
  - ↳ 1) start w/ an initial guess x_0 for the min. of $f(x)$. (random or informed)
  - ↳ 2) Propose a search dir. d_k
  - ↳ 3) Propose a step size α_k along d_k (typically by an inner line search loop)
  - ↳ 4) Update the estimate of the min location $x_{k+1} = x_k + \alpha_k d_k$.
  - ↳ 5) Repeat steps 2-5 until convergence

Convergence criteria and rate of convergence

- Convergence can be assessed in diff. ways. For a user-defined tolerance ε, we stop the search when
  - ↳ Norm of the residual $\|f(x_{k+1}) - f(x_k)\| < \varepsilon_f$
  - ↳ Norm of the error $\|x_{k+1} - x_k\| < \varepsilon_x$
  - ↳ Norm of the gradient $\|\nabla f(x_k)\| < \varepsilon_g$ ← used when the curvature is small, and combined w/ test on norm of the error.
- The rate of convergence of an algorithm quantifies how fast the approximate soln approaches the true min. x^*, close to the min, iteration after iteration. Mathematically,

$$\lim_{k \to \infty} \frac{\|x_{k+1} - x^*\|}{\|x_k - x^*\|^p} = \beta$$

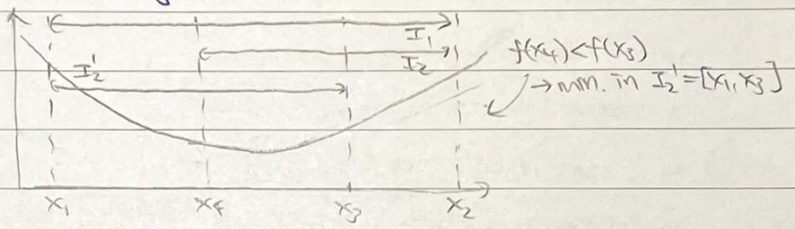
where β is the convergence ratio (smaller is better)

and p is the order of convergence (larger is better) — $p=1$: linear convergence; $p=2$: quadratic convergence

* The rate of convergence is an asymptotic quantity — it is the speed at which a sequence converges to its limit, if it converges.

Line search

- Line search is the search of a min. for a univariate function. Interval reduction is the best we can do if the gradient $f'(x)$ is not available.
- The line search algorithm is as follows:
  - 1) Start w/ three pts x_1, x_2, x_3 s.t. $f(x_1) > f(x_3)$ and $f(x_3) < f(x_2)$
    x_1 and x_2 are the bounds and x_3 is the interior pt.
  - 2) By Bolzano thm, there exists a min. in the interval $I_1 = [x_1, x_2]$ (bracketing)
  - 3) Compute $f(x_4)$ at a fourth pt $x_4 \in I_1$ (for the case $x_4 < x_3$)
  - 4) If $f(x_4) > f(x_3)$, the min. must be in $I_2 = [x_4, x_2]$, o/w, the min. is in $I_2' = [x_1, x_3]$
  - 5) Repeat steps 2-5 until convergence (interval smaller than tolerance)



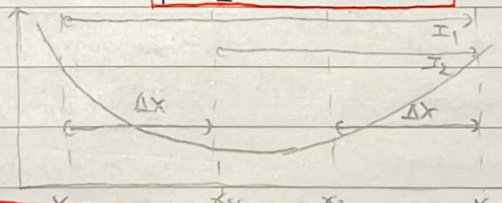
Golden section search

- To find the optimal location of the interior pt x_4, we impose a const. reduction factor β for the interval length, w/ linear convergence,

$$\frac{I_2}{I_1} = \frac{I_1 - I_2}{I_2} = \frac{\Delta x}{I_2}$$

Setting $\beta = \frac{I_2}{I_1}$ and solving the quadratic yields (inverse of the golden ratio)

$$\beta = \frac{\sqrt{5}-1}{2} \approx 0.618 = \frac{1}{\phi}$$



Note that $\Delta x = I_1 - I_2 = I_1(1-\beta)$. Find $\Delta x \rightarrow x_4 = x_1 + \Delta x$, $x_3 = x_2 - \Delta x$

- ✓: No need to compute the first or second derivatives
  ✓: Faster convergence than other "non-gradient" methods
  X: Slower convergence than gradient methods
  X: Not robust — goes wrong if the initial interval does not contain a min.

Fitting with a quadratic polynomial of three points

- we can approx. the objective function $f(x)$ w/ a quadratic function $q(x)$ by matching them at three pts.

$$q(x) = a_0 + a_1 x + a_2 x^2$$

we can solve the min. of $q(x)$ analytically $x^* = -\frac{a_1}{2a_2}$, which approx the min. of $f(x)$.

- To obtain the coefficients, we impose.

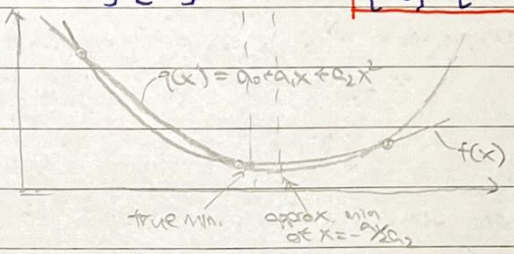
$$f(x_1) = a_0 + a_1 x_1 + a_2 x_1^2$$

$$f(x_2) = a_0 + a_1 x_2 + a_2 x_2^2$$

$$f(x_3) = a_0 + a_1 x_3 + a_2 x_3^2$$

using matrix form and solving, we have

$$\begin{bmatrix} f(x_1) \\ f(x_2) \\ f(x_3) \end{bmatrix} = \begin{bmatrix} 1 & x_1 & x_1^2 \\ 1 & x_2 & x_2^2 \\ 1 & x_3 & x_3^2 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \end{bmatrix} \rightarrow \boxed{\begin{bmatrix} a_0 \\ a_1 \\ a_2 \end{bmatrix} = \begin{bmatrix} 1 & x_1 & x_1^2 \\ 1 & x_2 & x_2^2 \\ 1 & x_3 & x_3^2 \end{bmatrix}^{-1} \begin{bmatrix} f(x_1) \\ f(x_2) \\ f(x_3) \end{bmatrix}}$$



Fitting with a quadratic polynomial of one point – Newton's method

- Instead of fitting the function at three pts, we can fit the function value, first and second derivatives evaluated at a single pt.

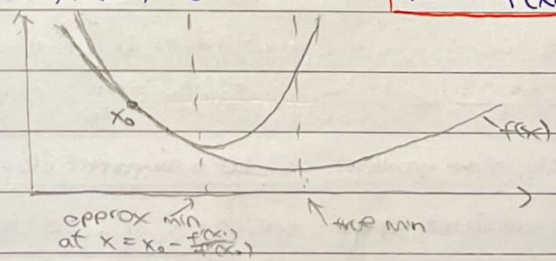
$$f(x) = f(x_0) + (x - x_0) f'(x_0) + \frac{1}{2}(x - x_0)^2 f''(x_0) + \text{H.O.T.}$$

We get a quadratic approx. $q(x)$ of $f(x)$ by neglecting the higher order terms,

$$q(x) = f(x_0) + (x - x_0) f'(x_0) + \frac{1}{2}(x - x_0)^2 f''(x_0)$$

- To calculate the min. of $q(x)$, we set its derivative to zero,

$$q'(x) = f'(x_0) + (x - x_0) f''(x_0) = 0 \rightarrow \boxed{x = x_0 - \frac{f'(x_0)}{f''(x_0)}}$$



- ✓ : When it works, has quadratic convergence ($p=2$)

✓ : could be generalized to multidimensional searches (but could be computationally expensive)

✗ : sensitive to the choice of the initial pt x_0 → could make the estimate worse or even converge to the wrong extremum (usually when $f''(x_0)$ is zero or negative)

Quasi-Newton methods

- The computation of the second derivatives can be expensive (especially for higher dim.)

A large no. of methods are designed to approx. the second derivative.

- The simplest approx is the secant method, which takes the current and previous pts to estimate the second derivative of the current pt.

$$f''(x_1) \approx \frac{f'(x_1) - f'(x_0)}{x_1 - x_0}$$

- The Taylor expansion abt. x_1 is approx. as

$$s(x) = f(x_1) + (x - x_1) f'(x_1) + \tfrac{1}{2}(x - x_1)^2 \frac{f'(x_1) - f'(x_0)}{x_1 - x_0}$$

Setting $s'(x) = 0$ yields the estimate for the min.

$$s'(x) = f'(x_1) + (x - x_1)\frac{f'(x_1) - f'(x_0)}{x_1 - x_0} = 0 \quad \rightarrow \quad x = x_1 - f'(x_1)\frac{x_1 - x_0}{f'(x_1) - f'(x_0)}$$

- Convergence for the secant method is superlinear ($p = \frac{1+\sqrt{5}}{2} \approx 1.618 > 1$)

- Quasi-Newton methods have higher dim. generalisations: Broyden's method, Barzilai-Borwein method.

Unconstrained optimisation — search methods for multivariate functions

Steepest-descent method

- Access to the gradient enables gradient search — more efficient than direct (gradient-free) search

- We can exploit the fact that the gradient is the dir. of max first-order change of a multivariate function $f(\underline{x})$. The steepest-descent algorithm is as follows:

↳ 1) Start w/ an initial guess for the min. $\underline{x}_0$.

↳ 2) Set the search dir. as the -ve gradient, $\underline{d}_k = -\nabla f(\underline{x}_k)$ ← main loop changes the direction

↳ 3) Determine by line search the step size α_k that min. $f(\underline{x})$ along $\underline{d}_k$.

↳ 4) Update the soln to $\underline{x}_{k+1} = \underline{x}_k + \alpha_k \underline{d}_k$ ← inner loop is optimisation in 1D

↳ 5) Repeat steps 2-4 until convergence.

- Ideally, the line search in step 3 should satisfy $\frac{df(\underline{x}_k + \alpha_k \underline{d}_k)}{d\alpha_k} = 0$, which implies that the gradient at $\underline{x}_{k+1} = \underline{x}_k + \alpha_k \underline{d}_k$ is orthogonal to $\underline{d}_k$, i.e. successive search dir. are orthogonal $\underline{d}_k^T \underline{d}_{k+1} = 0$

↳ $0 = \frac{df(\underline{x}_k + \alpha_k \underline{d}_k)}{d\alpha_k} = \frac{df(\underline{x}_{k+1})}{d\alpha_k} = \frac{\partial f(\underline{x}_{k+1})}{\partial \underline{x}_{k+1}} \cdot \frac{\partial \underline{x}_{k+1}}{\partial \alpha_k} = \nabla f(\underline{x}_{k+1}) \cdot \frac{\partial (\underline{x}_k + \alpha_k \underline{d}_k)}{\partial \alpha_k} = (-\underline{d}_{k+1}) \cdot \underline{d}_k$

- Practically, an approximate line search is performed, so $\frac{df(\underline{x}_k + \alpha_k \underline{d}_k)}{d\alpha_k} \approx 0 \rightarrow \underline{d}_k^T \underline{d}_{k+1} \approx 0$ ← approx too poor → may not converge; approx too good → waste resources

- If we approximate the objective function $f(\underline{x})$ w/ a 2nd order Taylor expansion, we can get a closed form optimum step size α_k, $\alpha_k = -\frac{\nabla f(\underline{x}_k)^T \underline{d}_k}{\underline{d}_k^T H(\underline{x}_k) \underline{d}_k}$

↳ $f(\underline{x}_k + \alpha_k \underline{d}_k) = f(\underline{x}_k) + \nabla f(\underline{x}_k)^T (\alpha_k \underline{d}_k) + \frac{1}{2}(\alpha_k \underline{d}_k)^T H(\underline{x}_k)(\alpha_k \underline{d}_k) + \text{H.O.T.}$

$\frac{df(\underline{x}_k + \alpha_k \underline{d}_k)}{d\alpha_k} = \nabla f(\underline{x}_k)^T \underline{d}_k + \alpha_k \underline{d}_k^T H(\underline{x}_k) \underline{d}_k = 0 \rightarrow \alpha_k = -\frac{\nabla f(\underline{x}_k)^T \underline{d}_k}{\underline{d}_k^T H(\underline{x}_k) \underline{d}_k}$

- However, if we have the Hessian $H(\underline{x}_k)$, better search methods (e.g. Newton-Raphson) are available.

convergence of steepest-descent method

- the steepest-descent algorithm converges linearly, i.e. $p=1$

$$\|\underline{x}_{k+1} - \underline{x}^*\| = \beta \|\underline{x}_k - \underline{x}^*\|$$

β is close to 1 → steepest-descent is slow.

- It can be shown using Kantorovich inequality that

$$\beta \leq \left(\frac{\kappa-1}{\kappa+1}\right)^2$$ ← $\kappa \to 1 \to \beta \to 0$ (good); $\kappa \to \infty \to \beta \to 1$ (bad)

where κ is the ℓ-2 condition no. of $H(\underline{x}^*)$, $\kappa = \frac{|\lambda_{max}(H(\underline{x}^*))|}{|\lambda_{min}(H(\underline{x}^*))|}$ ← note that the Hessian $H(\underline{x}^*)$ is Hermitian (symmetric).

- In typical engineering problems, $H(\underline{x}^*)$ is quite ill-conditioned → β is close to 1.

- Ill-conditioned quadratic functions have elongated valleys as isocontours → long valley of small gradients and a large distance to move → slow to converge.

Newton-Raphson method.

- Assume we can evaluate the gradient $\nabla f(\underline{x})$ and the Hessian $H(\underline{x})$.

- $f(\underline{x})$ is approx. by a quadratic function using the function value, gradient and Hessian at the current location.

$$f(\underline{x}) = f(\underline{x}_k) + \nabla f(\underline{x}_k)^T(\underline{x}-\underline{x}_k) + \frac{1}{2}(\underline{x}-\underline{x}_k)^T H(\underline{x}_k)(\underline{x}-\underline{x}_k) + \text{H.O.T.}$$

we get a quadratic approx $q(\underline{x})$ of $f(\underline{x})$ by neglecting the higher order terms,

$$q(\underline{x}) = f(\underline{x}_k) + \nabla f(\underline{x}_k)^T(\underline{x}-\underline{x}_k) + \frac{1}{2}(\underline{x}-\underline{x}_k)^T H(\underline{x}_k)(\underline{x}-\underline{x}_k).$$

- To calculate the min. of $q(\underline{x})$, we set its gradient to zero,

$$\nabla q(\underline{x}) = \nabla f(\underline{x}_k) + H(\underline{x}_k)(\underline{x}-\underline{x}_k) = 0 \quad \to \quad \underline{x} = \underline{x}_k - [H(\underline{x}_k)]^{-1}\nabla f(\underline{x}_k).$$

- We use the min. of $q(\underline{x})$ as our next guess for the min. of $f(\underline{x})$.

$$\underline{x}_{k+1} = \underline{x}_k + \alpha_k \underline{d}_k, \quad \alpha_k = 1 \text{ and } \underline{d}_k = -[H(\underline{x}_k)]^{-1}\nabla f(\underline{x}_k)$$

- We can extend the Newton-Raphson method by finding the step size α_k using line search at each iteration → speeds up convergence, will find a min. (not max/saddle).

convergence of Newton-Raphson method.

- The Newton-Raphson algorithm converges quadratically, i.e. $p=2$

$$\|\underline{x}_{k+1} - \underline{x}^*\| \leq \beta \|\underline{x}_k - \underline{x}^*\|^2$$

It converges to the min. in 1 iteration in a quadratic function $f(\underline{x})$.

- Newton-Raphson is not very robust — it may take the iterations away from a min. / not converge if the initial guess is not appropriate.

- W/O line search for step size α_k, if $H(\underline{x})$ is not PD, the method may converge to max./saddle.

- Newton-Raphson requires computing + inverting the Hessian. Computation $\sim O(N^2)$, Inversion $\sim O(N^3)$

Barzilai-Borwein method

- We can generalise the secant method for higher dimensions. The Hessian $H(x_k)$ in Newton-Raphson is approx as follows in the Barzilai-Borwein method.

Quasi-Newton method

$$\underline{x} = \underline{x}_1 - \nabla f(\underline{x}) \frac{(\underline{x}_1 - \underline{x}_0)^T (\nabla f(\underline{x}_1) - \nabla f(\underline{x}_0))}{|\nabla f(\underline{x}_1) - \nabla f(\underline{x}_0)|^2}$$

- Barzilai-Borwein method only requires computing the gradient $\nabla f(x_k)$, and is usually more stable than Newton's method or other Quasi-Newton methods.
- Not much is known about its convergence.

Conjugate gradient method.

- If we could step along the eigendirections, the convergence would be faster than steepest descent. (Eigendirections are good search dir. as they represent the principal axes of ellipsoids) — even for ill-conditioned quadratic functions.
- However, computing the eigenvectors $\sim O(N^3)$ — as bad as Hessian-based methods → don't use eigenvectors.
- Consider a set of A-conjugate vectors $\{\underline{d}_1 \dots \underline{d}_N\}$, i.e.

$$\underline{d}_i^T \underline{\underline{A}} \underline{d}_j = 0 \quad \text{for } i \neq j$$

(true for symmetric $\underline{\underline{A}}$.)

(Orthogonal eigenvectors fulfill the conjugacy condition, but there are other vectors that satisfy this)

- Consider transforming the coords $x \to y$, where $\underline{x} = \underline{\underline{S}}\,\underline{y}$, $\underline{\underline{S}} = [\underline{d}_1\ \underline{d}_2 \dots \underline{d}_N]$.

For a quadratic function $f(\underline{x}) = \underline{x}^T \underline{c} - \frac{1}{2}\underline{x}^T \underline{\underline{A}} \underline{x}$, we have

$$f(\underline{x}) = f(\underline{\underline{S}}\,\underline{y}) = (\underline{\underline{S}}\,\underline{y})^T \underline{c} - \frac{1}{2}(\underline{\underline{S}}\,\underline{y})^T \underline{\underline{A}} (\underline{\underline{S}}\,\underline{y}) = \underline{y}^T (\underline{\underline{S}}^T \underline{c}) - \frac{1}{2}\underline{y}^T (\underline{\underline{S}}^T \underline{\underline{A}} \underline{\underline{S}}) \underline{y}.$$

By construction, $\underline{\underline{S}}^T \underline{\underline{A}} \underline{\underline{S}} = \underline{\underline{\Delta}}$.

$$\underline{\underline{S}}^T \underline{\underline{A}} \underline{\underline{S}} = \begin{bmatrix} \uparrow & \uparrow & & \uparrow \\ \underline{d}_1 & \underline{d}_2 & \cdots & \underline{d}_N \\ \downarrow & \downarrow & & \downarrow \end{bmatrix}^T \underline{\underline{A}} \begin{bmatrix} \uparrow & \uparrow & & \uparrow \\ \underline{d}_1 & \underline{d}_2 & \cdots & \underline{d}_N \\ \downarrow & \downarrow & & \downarrow \end{bmatrix} = \begin{bmatrix} \underline{d}_1^T \underline{\underline{A}} \underline{d}_1 & \underline{d}_1^T \underline{\underline{A}} \underline{d}_2 & \cdots & \underline{d}_1^T \underline{\underline{A}} \underline{d}_N \\ \underline{d}_2^T \underline{\underline{A}} \underline{d}_1 & \underline{d}_2^T \underline{\underline{A}} \underline{d}_2 & \cdots & \underline{d}_2^T \underline{\underline{A}} \underline{d}_N \\ \vdots & \vdots & \ddots & \vdots \\ \underline{d}_N^T \underline{\underline{A}} \underline{d}_1 & \underline{d}_N^T \underline{\underline{A}} \underline{d}_2 & \cdots & \underline{d}_N^T \underline{\underline{A}} \underline{d}_N \end{bmatrix} = \begin{bmatrix} \underline{d}_1^T \underline{\underline{A}} \underline{d}_1 & 0 & \cdots & 0 \\ 0 & \underline{d}_2^T \underline{\underline{A}} \underline{d}_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \underline{d}_N^T \underline{\underline{A}} \underline{d}_N \end{bmatrix} = \underline{\underline{\Delta}}$$

→ The problem in the y space is a quadratic form along the axes (search dir. $\{\underline{d}_1 \dots \underline{d}_N\}$).

- At iteration k, we have the gradient $\nabla f(x_k)$ and the previous search dir. $\{\underline{d}_1 \dots \underline{d}_{k-1}\}$.

We construct the new search dir. as

$$\underline{d}_k = -\nabla f(\underline{x}_k) + \beta_k \underline{d}_{k-1}$$

($\underline{d}_k$ has memory of the previous dir. $\underline{d}_{k-1}$.)

Imposing the conjugacy condition $\underline{d}_{k-1}^T \underline{\underline{A}} \underline{d}_k = 0$,

$$\underline{d}_{k-1}^T \underline{\underline{A}} (-\nabla f(\underline{x}_k) + \beta_k \underline{d}_{k-1}) = -\underline{d}_{k-1}^T \underline{\underline{A}} \nabla f(\underline{x}_k) + \beta_k \underline{d}_{k-1}^T \underline{\underline{A}} \underline{d}_{k-1} \to \boxed{\beta_k = \frac{\underline{d}_{k-1}^T \underline{\underline{A}} \nabla f(\underline{x}_k)}{\underline{d}_{k-1}^T \underline{\underline{A}} \underline{d}_{k-1}}}$$

It can be shown that w/ this choice of β_k, the search dir. $\underline{d}_k$ is A-conjugate to all previous search dir.

- The conjugate gradient algorithm is as follows:

1) Initialise w/ steepest descent $\underline{d}_0 = -\nabla f(\underline{x}_0)$
2) Update location and determine α_k w/ line search, $\underline{x}_{k+1} = \underline{x}_k + \alpha_k \underline{d}_k$
3) Find a new search dir $\underline{d}_{k+1}$, $\underline{d}_{k+1} = -\nabla f(\underline{x}_{k+1}) + \beta_{k+1} \underline{d}_k = -\nabla f(\underline{x}_{k+1}) + \left[\frac{|\nabla f(\underline{x}_{k+1})|}{|\nabla f(\underline{x}_k)|}\right]^2 \underline{d}_k$
4) Repeat steps 2-3 until convergence

Convergence of conjugate gradient method.

- The conjugate gradient algorithm converges linearly, i.e. $p=1$

$$\|\underline{x}_{k+1}-\underline{x}^*\| = \beta \|\underline{x}_k-\underline{x}^*\|$$

β is close to 0 → conjugate gradient is superlinear.

- For large ill-conditioned problems, it can be shown that

$$\beta \propto \left(\frac{\nu}{\sqrt{\kappa}}\right)$$

where κ is the l-2 condition no.

- The conjugate gradient method does not req. the Hessian $H(\underline{x}^*)$ and its inversion $[H(\underline{x}^*)]^{-1}$.

- The conjugate gradient method is a Krylov subspace method — for a N-dimensional objective function $f(\underline{x})$, it converges after N iterations.

- Numerical errors can lead to loss of conjugacy → periodically reset by setting $g_k = -\nabla f(\underline{x}_k)$.

- The conjugate gradient method is originally used to solve linear systems $\underline{A}\underline{x}=\underline{b}$.

For optimisation of quadratic functions, $f(\underline{x}) = \frac{1}{2}\underline{x}^T\underline{A}\underline{x} - \underline{b}^T\underline{x} \rightarrow \nabla f(\underline{x}) = \underline{A}\underline{x}-\underline{b} = 0$

Summary of multi-dimensional search-direction methods.

Method.	Advantages	Disadvantages
Steepest-descent	Only needs gradient	Linear convergence slow for ill-conditioned problems
Newton-Raphson	Quadratic convergence (More reliable w/ line search)	Needs Hessian can diverge
Barzilai-Borwein	Very stable Faster than steepest descent	convergence analysis difficult
Conjugate gradient	Faster than steepest descent	Linear convergence

Least squares fitting of a model to data.

- Let $y^{(j)}$ be the measured data, and $\phi_j(\underline{x})$ be the model's prediction for $y^{(j)}$. $j=1,2\ldots m$ (usually nonlinear)

$\underline{x} = [x_1 \ldots x_n]^T$ is the vector of independent parameters, which can be tuned to fit the model to data.

- The elements of the residual vector $r(\underline{x})$ are the errors in the model's prediction for each data pt.

$$r_j(\underline{x}) = \phi_j(\underline{x}) - y^{(j)}$$

The cost function we want to min. is the sum of squared errors.

$$f(\underline{x}) = \sum_{j=1}^{m} r_j(\underline{x})^2 = r(\underline{x})^T r(\underline{x})$$

The gradient $\nabla f(\underline{x})$ is given by

$$\frac{\partial f}{\partial x_i} = \frac{\partial}{\partial x_i}\left(\sum_{j=1}^{m} r_j(\underline{x})^2\right) = 2\sum_{j=1}^{m} \frac{\partial r_j(\underline{x})}{\partial x_i} r_j(\underline{x}) \rightarrow \nabla f(\underline{x}) = 2J(\underline{x})^T r(\underline{x})$$

where $J(\underline{x}) = \begin{bmatrix} \partial r_1/\partial x_1 & \cdots & \partial r_1/\partial x_n \\ \partial r_m/\partial x_1 & \cdots & \partial r_m/\partial x_n \end{bmatrix}$ is the Jacobian matrix of $r(\underline{x})$.

The Hessian $H(\underline{x})$ is given by

$$\frac{\partial^2 f}{\partial x_i^2} = \frac{\partial}{\partial x_i}\left(2\sum_{j=1}^{m} \frac{\partial r_j(\underline{x})}{\partial x_i} r_j(\underline{x})\right) = 2\sum_{j=1}^{m} \frac{\partial^2 r_j(\underline{x})}{\partial x_i^2} r_j(\underline{x}) + 2\sum_{j=1}^{m} \frac{\partial r_j(\underline{x})}{\partial x_i}\frac{\partial r_j(\underline{x})}{\partial x_i} \rightarrow H(\underline{x}) = 2\sum_{j=1}^{m} r_j(\underline{x}) R_j(\underline{x}) + 2J(\underline{x})^T J(\underline{x})$$

where $R_j(\underline{x}) = \nabla(\nabla r_j(\underline{x}))$ is the Hessian of the residual $r_j(\underline{x})$

Gauss-Newton method

- The Gauss-Newton method assumes that the residuals at the optimum $r_j(x^*)$ are small, therefore $R_j(x)$ can be neglected, so the Hessian $H(x)$ can be approximated as.

$$H(\underline{x}) \approx \tilde{H}(\underline{x}) = 2\underline{J}(\underline{x})^T \underline{J}(\underline{x})$$

for $\underline{x}$ near the minimum $\underline{x}^*$, and $\tilde{H}(\underline{x})$ is PD.

- Applying the approx. for the Hessian $H(x)$ to the Newton-Raphson method, the search dir. $\underline{d}_k$ is

$$\underline{d}_k = -[\tilde{H}(\underline{x}_k)]^{-1} \nabla f(\underline{x}_k) = -[2\underline{J}(\underline{x}_k)^T \underline{J}(\underline{x}_k)]^{-1} 2\underline{J}(\underline{x}_k)^T r(\underline{x}_k) = -\underline{J}(\underline{x}_k)^+ r(\underline{x}_k)$$

cannot distribute the -1 as $J(x)$ may not be square.

where $\underline{J}(\underline{x}_k)^+ = [\underline{J}(\underline{x}_k)^T \underline{J}(\underline{x}_k)]^{-1} \underline{J}(\underline{x}_k)^T$ is the pseudo inverse matrix

- The estimate is updated as for the Newton-Raphson method, $\underline{x}_{k+1} = \underline{x}_k + \underline{d}_k$.
- The Gauss-Newton method only requires the Jacobian (first derivatives) and avoids computing or inverting the Hessian $H(x)$.
- The Gauss-Newton method converges nearly quadratically if the initial residuals are small (o/w, it has poor performance)

Constrained optimisation — Linear programming.

Linear programming (LP)

- LP is an optimisation problem in which both the objective and constraints are linear
- The standard / canonical form of a linear program is as follows:

Minimise	$f(\underline{x}) = \underline{c}^T \underline{x}$, $\underline{x} \in \mathbb{R}^n$
subject to m equality constraints	$\underline{\underline{A}}\,\underline{x} = \underline{b}$, $\underline{\underline{A}} \in \mathbb{R}^{m \times n}$
and bounds	$x_i \geq 0$, $i = 1 \dots n$

* We assume that $\text{rank}(\underline{\underline{A}}) = m$

- If there are inequality constraints of the form $\underline{a}_j^T \underline{x} \geq b_j$, we can convert them into equality constraints by introducing a new slack variable s_j, $s_j = \underline{a}_j^T \underline{x} - b_j$, where $s_j \geq 0$.
- The m constraints $\underline{\underline{A}}\,\underline{x} = \underline{b}$ define a subspace of $\mathbb{R}^n$ (usually $m < n$ o/w the feasible set is empty/ $\underline{\underline{A}}^{-1}\underline{b}$)
- The bounds $x_i \geq 0$ cut out a portion of this subspace to define the feasible region (convex polyhedron)
- The feasible region has a no. of corners - extremal pts, which are the basic soln's. The basic soln's have m non-zero basic variables and $n-m$ zero free (non-basic) variables
- The isocontours of f are a series of parallel $(n-1)$-dim hyperplanes. As f increases, the first isocontour to intersect the feasible region will do so at the minimum.

* We assume the problem is non-degenerate → there are m (non-zero) basic variables (o/w, cycling may occur)

* For low dimensional problems, LP can be solved graphically.

Fundamental theorem of linear programming.

- Given a LP in standard form where $A \in \mathbb{R}^{m \times n}$ and $\text{rank}(A) = m$, if there is a feasible soln, there is a basic feasible soln; if there is an optimal feasible soln, there is an optimal basic feasible soln.

→ We can solve a linear program by inspecting basic feasible solns until we find the optimum.

The simplex algorithm.

- Large LPs cannot be easily solved by hand → simplex algorithm systematically solves LPs.
- The simplex algorithm involves two phases:
  - (i) Phase 1: start the algorithm.
    - ↳ Find an initial basic feasible soln if there is not an obvious one
    - ↳ start from the initial basic feasible soln (vertex of feasible region w/ m (non-zero) basic variables)
  - (ii) Phase 2: move from one vertex to another until the min. is found
    - ↳ Move along an edge of the feasible region to another vertex where f is smaller
    - ↳ Repeat until you find a vertex where edge leading away would increase f → this is the min.

Phase 2: Move along one vertex to another (Tableau method)

- Step 0: Formalise the problem
  - ↳ minimise $\underline{c}^T \underline{x} = 200x_0 + 400x_1 + 680x_2 + 800x_3$

    subject to $\underline{\underline{A}}\underline{x} = \underline{b} \leftrightarrow \begin{bmatrix} 3 & 7 & 5 & 3 \\ 0 & 3 & 5 & 7 \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 80 \\ 40 \end{bmatrix} \leftrightarrow \begin{bmatrix} 3x_0 + 7x_1 + 5x_2 + 3x_3 = 80 \\ 3x_1 + 5x_2 + 7x_3 = 40 \end{bmatrix}$

    and $x_i \geq 0$
  - ↳ We can collect the constraint eqns and the objective function coeff into a single tableau ($(m+1) \times (n+1)$ matrix)

$$\begin{bmatrix} \underline{\underline{A}} & \underline{b} \\ \underline{c}^T & 0 \end{bmatrix} = \begin{bmatrix} x_0 & x_1 & x_2 & x_3 & \text{RHS} & \\ 3 & 7 & 5 & 3 & 80 & [1] \\ 0 & 3 & 5 & 7 & 40 & [2] \\ 200 & 400 & 680 & 800 & 0 & [3] \end{bmatrix} \begin{matrix} \\ \\ \text{cost function} \end{matrix} \quad \begin{matrix} \\ \text{constraints} \\ \\ \end{matrix}$$

(variables)

- Step 1: Start from a basic soln.
  - ↳ From phase 1, we get an initial basic feasible soln $\begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 0 \\ 10 \\ 2 \\ 0 \end{bmatrix}$ → x_1, x_2 are the basic variables.
  - ↳ Manipulate the tableau to get canonical form. (under basic variables, we get $\begin{smallmatrix} 1 & 0 \\ 0 & 1 \end{smallmatrix}$)

    Eliminate the coeff. of the basic variables in the last line (get reduced cost. -ve of objective function)

$$[4] := \tfrac{1}{4} \times ([1] - [2]) \qquad \begin{matrix} x_0 & x_1 & x_2 & x_3 & \text{RHS} \end{matrix}$$
$$[5] := \tfrac{1}{5} \times ([2] - 3 \times [4]) \qquad \begin{bmatrix} 3/4 & 1 & 0 & -1 & 10 \\ -9/20 & 0 & 1 & 2 & 2 \\ 206 & 0 & 0 & -160 & -5360 \end{bmatrix} \begin{matrix} [4] \\ [5] \\ [6] \end{matrix}$$
$$[6] := [3] - 400[4] - 680[5]$$

- Step 2: Identify the edge to follow
  - ↳ Choose the entering variable to be the one w/ the most -ve reduced cost → x_3 is EV
  - ↳ To choose the leaving variable, consider the ratios $\frac{\text{RHS}}{x_3}$ for each row, i.e. $\frac{10}{-1}$ vs $\frac{2}{2}$

    We choose the variable corresponding to the smallest, +ve ratio → x_2 is LV
- Step 3: calculate the next vertex
  - ↳ Repeat steps 1, 2 until all reduced costs are +ve.

$$\begin{matrix} x_0 & x_1 & x_2 & x_3 & \text{RHS} \end{matrix}$$
$$\begin{bmatrix} 0.525 & 1 & 0.5 & 0 & 11 \\ -0.225 & 0 & 0.5 & 1 & 1 \\ 170 & 0 & 80 & 0 & 5200 \end{bmatrix} \rightarrow \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 0 \\ 11 \\ 0 \\ 1 \end{bmatrix}$$

Phase 1: start the algorithm

- To find an initial basic feasible soln, we construct an auxiliary optimisation problem, where the objective function is the "error" that measures by how much the constraints are violated.
- The optimal soln of this problem has zero error → basic feasible soln.
- We create n slack variables, add one to each of the constraint eqns. The objective function is the sum of the slack variables. (the slack variables are the initial basic variables).
- Consider the previous example. The auxiliary problem is

$$3x_0 + 7x_1 + 5x_2 + 3x_3 + s_1 + 0 = 80$$
$$0 + 3x_1 + 5x_2 + 7x_3 + 0 + s_2 = 40.$$

The auxiliary objective function is $s_1 + s_2$, so the auxiliary tableau is.

x_0	x_1	x_2	x_3	s_1	s_2	RHS	
3	7	5	3	1	0	80	[1]
0	3	5	7	0	1	40	[2]
0	0	0	0	1	1	0	[3]

Eliminating the coeff of the basic variables in the last line,

	x_0	x_1	x_2	x_3	s_1	s_2	RHS	
	3	7	5	3	1	0	80	[1]
	0	3	5	7	0	1	40	[2]
[4]: [3]-[1]-[2]	-3	-10	-10	-10	0	0	-120	[4]

Then solve the auxiliary tableau as in phase 2.

- After solving the auxiliary tableau, if $s_1 = s_2 = 0$ (slack variables are non-basic variables), and the cost function equals 0, then we have found the initial basic feasible soln.

If $s_1 > 0$ and $s_2 > 0$, there is no feasible soln to the original problem.

Constrained optimisation — non-linear constrained optimisation

Nonlinear constrained optimisation

- Consider the optimisation problem (generally non-linear)

minimise	$f(\underline{x})$	$\underline{x} \in \mathbb{R}^N$	
subject to	$h_i(\underline{x}) = 0$	$i = 1, 2 \ldots m$	[equality constraints]
	$g_j(\underline{x}) \leq 0$	$j = 1, 2 \ldots n$	[inequality constraints]

The set of equality and inequality constraints defines the feasible region.

- At a location $\underline{x}$, constraints are (equalities, active inequalities)
  - ↳ active if they do not allow $\underline{x}$ to infinitesimally change in at least one dir.
  - ↳ inactive if they allow $\underline{x}$ to infinitesimally change in any dir, ← inactive inequalities.
- In N dimensions, it is not possible to have a soln w/ more than N independent constraints active.

Optimality condition on active constraints

- A necessary condition for $\underline{x}^*$ to be a local min. is

$$\nabla f(\underline{x}^*) \cdot \underline{d} \geq 0 \quad \text{for all feasible directions } \underline{d}.$$

In constrained optimization, some directions $\underline{d}$ may not be feasible, so in general,

$$\nabla f(\underline{x}^*) \cdot \underline{d} \geq 0 \not\Rightarrow \nabla f(\underline{x}^*) = 0$$

- consider that $\underline{x}$ is subject to one equality constraint $h(\underline{x}) = 0$.

Let $\underline{d}$ be the tangential dir. to the constraint. Suppose $\pm\underline{d}$ is a feasible dir.

↳ If $\nabla f(\underline{x}^*) \cdot \underline{d} < 0 \rightarrow \underline{x}$ is not a min.

↳ If $\nabla f(\underline{x}^*) \cdot \underline{d} > 0 \rightarrow$ in opposite dir $\underline{d}' = -\underline{d}$, $\nabla f(\underline{x}^*) \cdot \underline{d}' < 0 \rightarrow \underline{x}$ is not a min.

$\Rightarrow \underline{x}$ is a min. if $\nabla f(\underline{x}^*) \cdot \underline{d} = 0$.

Geometrically, $\nabla f(\underline{x}^*) \perp \underline{d} \rightarrow \nabla f(\underline{x}^*) \parallel \nabla h(\underline{x}^*)$, i.e. $\boxed{\nabla f(\underline{x}^*) = -\lambda \nabla h(\underline{x}^*)}$ where λ is a scalar

- w/ multiple independent active constraints, at the min, ∇f must be perpendicular to the tangent plane of the constraint intersection, which has dimensionality N-m. The optimality condition is thus

$$\boxed{\nabla f(\underline{x}^*) = -\sum_{i=1}^{m} \lambda_i \nabla h_i(\underline{x}^*) = -\nabla h(\underline{x}^*)^T \underline{\lambda}}, \text{ where } \lambda_i, i = 1,2...m \text{ are scalars}$$

Lagrangian and optimality condition for equality constraints

- We can recover the optimality condition by defining an unconstrained min. problem of a new cost function L

$$\boxed{L(\underline{x}, \underline{\lambda}) = f(\underline{x}) + \sum_{i=1}^{m} \lambda_i h_i(\underline{x})} \quad \text{note } h_i(\underline{x}) = 0$$

where $L(\underline{x}, \lambda_i)$ is the Lagrangian and λ_i are the Lagrangian multipliers (scalars)

- $\underline{x}^*$ is a stationary pt. of the Lagrangian $L(\underline{x}, \lambda_i)$ if

$$\nabla L(\underline{x}^*, \underline{\lambda}) = 0 \Rightarrow \boxed{\nabla f(\underline{x}^*) + [\nabla h(\underline{x}^*)]^T \underline{\lambda} = 0} \quad \text{and} \quad \boxed{h_i(\underline{x}) = 0} \quad \text{for } i = 1,2...m$$

A sufficient condition for $\underline{x}^*$ to be a minimum is

$$\boxed{\underline{d}^T \nabla^2 L(\underline{x}^*, \lambda_i) \underline{d} > 0} \quad \text{where } d \in \text{space tangent to all constraints.}$$

i.e. the Hessian $\nabla^2 L(\underline{x}^*, \lambda_i)$ is PD in the tangent space.

* If the Lagrangian multiplier is zero, $\lambda_i = 0$, then $\nabla f(\underline{x}^*) = 0$ at that pt.

Lagrangian and optimality conditions for inequality constraints

- For inequality constraints, we define the Lagrangian

$$L(\underline{x}, \underline{\mu}) = f(\underline{x}) + \sum_{j=1}^{n} \mu_j g_j(\underline{x})$$ Note $g_j(\underline{x}) \leq 0$

- $\underline{x}^*$ is a stationary pt. of the Lagrangian $L(\underline{x}, \mu_j)$ if

$$\nabla L(\underline{x}^*, \mu_j) = 0 \Rightarrow \nabla f(\underline{x}^*) + [\nabla g(\underline{x}^*)]^T \underline{\mu} = 0 \quad \text{and} \quad \mu_j g_j(\underline{x}^*) = 0, \ \mu_j \geq 0 \text{ for } j = 1, 2 \ldots n$$

These are the Karush-Kuhn-Tucker (KKT) conditions, μ_j are the KKT multipliers.

- For active constraints, $g_j(\underline{x}) = 0$ [constraint becomes an equality constraint]

For inactive constraints, $g_j(\underline{x}) < 0$ and $\mu_j = 0$

- In the general case w/ both equalities and inequality constraints,

$$L(\underline{x}, \lambda_i, \mu_j) = f(\underline{x}) + \sum_{i=1}^{m} \lambda_i h_i(\underline{x}) + \sum_{j=1}^{n} \mu_j g_j(\underline{x})$$

Sensitivity.

- consider perturbing the soln $\underline{x}^*$ by an infinitesimally small amount, i.e. $\underline{x}^* \to \underline{x}^* + \delta\underline{x}$.

$$\nabla f(\underline{x}^*) + [\nabla h(\underline{x}^*)]^T \underline{\lambda} + [\nabla g(\underline{x}^*)]^T \underline{\mu} = 0 \rightarrow \delta\underline{x}^T \nabla f(\underline{x}^*) = -\delta\underline{x}^T \left([\nabla h(\underline{x}^*)]^T \underline{\lambda} + [\nabla g(\underline{x}^*)]^T \underline{\mu} \right)$$

$$\therefore \quad \delta f(\underline{x}^*) = -\delta h(\underline{x}^*)^T \underline{\lambda} - \delta g(\underline{x}^*)^T \underline{\mu}$$

→ change in cost function due to a change in the soln is prop. to the -ve Lagrangian / KKT multipliers.

i.e. $-\lambda_i$ and μ_j are the sensitivities of f to an infinitesimal change in the constraint h_i and g_j respectively.

Penalty functions and barrier functions

- w/ penalty functions, we do not enforce the constraints exactly, but only approx and controllably. (i.e. soft constraints), we define a new unconstrained problem

$$q(\underline{x}, K) = f(\underline{x}) + K \sum_{i=1}^{m} h_i(\underline{x})^2$$

where K is the penalty parameter and $K \sum_{i=1}^{m} h_i(\underline{x})^2$ is the penalty function.

- It can handle both equalities and inequalities (for inequalities, use $K \sum_{j=1}^{n} \max[0, g_j(\underline{x})]^2$)

- For large K, $h(\underline{x})$ needs to be very small → problem close to constrained problem. For small K, $h(\underline{x})$ does not need to be small → constraint violated more. usually start w/ small K and increase it.

- w/ barrier functions, the feasibility is enforced exactly by using a penalty that diverges as we approach the boundary of the constraint. we define a new unconstrained problem

inverse barrier function $$q(\underline{x}, K) = f(\underline{x}) - \frac{1}{K} \sum_{j=1}^{m} \frac{1}{g_j(\underline{x})}$$

logarithmic barrier function $$q(\underline{x}, K) = f(\underline{x}) - \frac{1}{K} \sum_{j=1}^{m} \ln[-g_j(\underline{x})]$$

- It can only handle inequalities and always yields a feasible answer.

- If req. a feasible starting pt and cannot iterate through infeasible space.

a It basically restricts the search to the feasible region by penalising soln's near the edge.

- The larger the K, the sharper the divergence of the singularity near the constraint boundary.