1) Definitions & Properties

Linear Dependence – if some sum of our vectors with any coefficients = 0, they are linearly dependent. If we have pivot vars, we are linearly independent, else we're not.

Properties of Determinants: $A, B \in R^{n\times n}$

1) $\det(A^T) = \det(A)$
2) $\det(AB) = \det(BA) = \det(A)\det(B)$.
3) $\det(kA) = k^n\det(A)$

$\det(A) = 0 \Leftrightarrow$ Matrix Not Invertable. Else, $\det(A^{-1}) = 1/\det(A)$.

We can find the determinant of a square matrix only, and this is done by getting into RREF and multiplying by lead diag. The following operations have effects on det:

1) Swapping row multiplies it by -1
2) Adding/subbing rows does nothing
3) If any two rows are equal, or lead diag = 0, then det = 0.
4) Multiplying by scalar also increases det by a scalar.

Trace – sum of diag elems.

Rank of invertible matrices – If an n^2 matrix is invertible ⇔ rank is n ⇔ columns are linearly indep ⇔ rows linearly indep.

Singular – A square matrix is nonsingular if the columns are linearly indep, ie: if rk(A) = n, or det(A) =/= 0. Else its singular.

Vector Space – a set U =/= ∅ is a vector space if U is closed under addition and scalar multiplication ie:

1) For all u, v ∈ U, u + v ∈ U
2) For all u ∈ U and c ∈ R, cu ∈ U

Vector Subspace: subset of a Vector Space

Generating set: Our vector subspace X is a generating set of U it could express every vector U in the Vector Space as a linear combo of its vectors.

A **basis** is a minimal generating set. To find it we get the REF, and take original vector of each pivot column. Span of this = basis.

A **simple basis** is one with as many 0s, as possible, gotten by getting RREF and taking the pivot columns span as our simple basis.

Dimension = number of basis vectors.

M1) Finding Change of Basis Matrix

We just represent each basis vector in terms of the other basis, and each representation is one of our columns.

$$\mathcal{B} = \left\{\begin{bmatrix}1\\-2\end{bmatrix}, \begin{bmatrix}1\\1\end{bmatrix}\right\} \mathcal{B}' = \left\{\begin{bmatrix}2\\1\end{bmatrix}, \begin{bmatrix}-1\\1\end{bmatrix}\right\}$$

Here, $B_1 = 4B'_1 + 6B'_2$. $B_2 = -B'_2$.

$$I_{B'B} = \begin{bmatrix}4 & 0\\6 & -1\end{bmatrix}$$

Euclidean norm: magnitude of vector, square each item and then root total.

Parallelogram Law: $\forall u, v \in R^n$, $||u+v||^2+||u-v||^2=2||u||^2+2||v||^2$

Angle between vectors: $\cos(x) = u.v / ||u||\,||v||$

2) Orthogonality

Vectors u and v are **orthogonal** if u.v = 0. They're **orthonormal** if they're orthogonal and their magnitudes are 1.

A matrix $A \in R^{n\times n}$ is **orthogonal** if they are invertible and $A^{-1} = A^T$

Two subspaces are **orthogonal** if ∀u ∈ U, ∀v ∈ V, u · v = 0. We write U ⊥ V.

3) Cauchy Schwarz inequality: $u, v \in R^n$, $|u \cdot v| \le ||u||\,||v||$

4) Triangle inequality: $\forall u, v \in R^n$, $||u + v|| \le ||u|| + ||v||$
$u \perp v \Leftrightarrow ||u + v||^2 = ||u||^2 + ||v||^2$

5) Linear Maps

A map from one vector subspace to another is linear if:
For all u, v ∈ V , f(u + v) = f(u) + f(v)
For all u ∈ V and c ∈ R, f(cu) = cf(u)

A basis change matrix from u to v is a linear map. Here the matrix [0 1] ... is a linear map from the subspaces visible.

$$f(x) = \begin{bmatrix}y\\x\\y+x\end{bmatrix} = \begin{bmatrix}0&1\\1&0\\1&1\end{bmatrix} * \begin{bmatrix}x\\y\end{bmatrix}.$$

We can compose basis change and linear maps.

$$\begin{tikzcd}
\text{coord in } \mathcal{B} \arrow[r, "F_{\mathcal{DB}}"] & \text{coord in } \mathcal{D} \arrow[d, "I_{\mathcal{D}'\mathcal{D}}"] \\
\text{coord in } \mathcal{B}' \arrow[u, "I_{\mathcal{BB}'}"] \arrow[r, "F_{\mathcal{D}'\mathcal{B}'}"] & \text{coord in } \mathcal{D}'
\end{tikzcd}$$

In an f: $R^n \to R^m$ linear map, the **Image Space:** refers to the set of points mapped to in R^m. The **Kernel/Null** space is the set of points in the N space that is 0.

M2) Finding Image and Basis

We can find the basis of an image by just taking the span of the columns of the linear map. We find the basis of the kernel, we find the image, get it into RREF, and solve for the values and free vars as usual.

Rank Nullity Theorem: For a linear map $R^n \to R^m$ linear map with matrix $A \in R^{m\times n}$:
dim(im(f)) + dim(ker(f)) = n
rk(A) + dim(ker(A)) = n

M3) Intersection of Subspaces

Let U = span $\{[1\ 0\ 0]^T, [0\ 1\ 0]^T\}$
Let V = span $\{[1\ 1\ 1]^T, [-1\ 1\ 1]^T\}$
Then we solve $x = V_1 + V_2$ and $x = U_1 + U_2$ and get the line.

6) Eigenvalues and Eigenvectors

We find the eigenvalues of a Matrix A by solving Characteristic Polynomial (CP) of A.

Spectrum of a matrix: set of its eigenvalues. tr(A) = sum of eigenvalues. det(A) = product of eigenvalues.

The Algebraic Multiplicity (AM) of eigenvalue = number of times it is a root of the CP. The Geometric Multiplicity (GM) = dimension of the eigenspace of the eigenvalue. **1 <= AM <= GM.**

The Eigenspace of a matrix is the span of its eigenvectors, as eigvals can have multiple. To find the EVecs of an EVal, we solve. (A-kI)x = 0

M4) Diagonalization of a Matrix A:

1) Obtain the eigenvalues by solving CP, get their eigenspaces.
2) Write the matrix in the form $A = PDP^{-1}$, by writing D = matrix of eigenvalues, P = matrix of eigenvectors, preserving order.

Similar Matrices: Two matrices A and B are similar if there is a matrix P such that $A = PBP^{-1}$.

Cayley Hamilton Theorem: If we take the characteristic polynomial of A and sub in A itself, we get the 0 matrix. We can find inverses this way:
A = [1 -1, 2 1]. Char polyn = $x^2-2x+3 = 0$. Subbing A in:
$-A^2+2A=3I \Leftrightarrow A\,1/3(-A+2I) = I \Leftrightarrow$
$A^{-1}=1/3(2I-A)$

Computational Techniques

1) Vector Norms: Ways of computing size on a vector. They satisfy the properties:
For vector x, y in R^n, scalar k
1) ||x|| > 0. (for non-zero x)
2) ||kx|| = ||k|| ||x||
3) ||x+y|| <= ||x|| + ||y||

The l_p norm is $||v||_p = \left(\sum_{i=1}^n |v_i|^p\right)^{1/p}$

L_1 norm = sum of vector elems.
L_2 = Euclidean Norm
L_∞ = max vector entry.

Any two vector norms $||.||_a$ and $||.||_b$ in R^n are equivalent – there are r > 0 and s > 0 s.t
$\forall x \in R^n$, $r||x||_a \le ||x||_b \le s||x||_a$
L_1, L_2, L_∞ are **equivalent:**
1) $||x||_\infty \le ||x||_2 \le ||x||_1$
2) $||x||_2 \le \sqrt{n}||x||_\infty$
3) $||x||_1 \le \sqrt{n}||x||_2$

2) Matrix Norms (MN):
||A|| > 0 ||kA|| = ||k|| ||A||
||A + B|| <= ||A|| + ||B||, as with vector norms. We only consider Submultiplicative MNs, which also have ||AB|| <= ||A|| ||B||
L_1=max absolute column sum (e.g. for each column, add up the absolute values of each entry. Highest total = L_1).
L_2=Largest singular value of A.
L_3= Maximum absolute row sum.

A Matrix norm is **consistent** with the vector norms $||.||_a$ and $||.||_b$ if for all $A^{m\times n}$ <= $||A||\,||x||_a$ if a = b, they're compatible.

Subordinate Matrix Norm: $\forall A\in R^{m\times n}$
$||A|| = \max\{ ||Ax|| : x \in R^n, ||x||=1 \}$ (*)
AKA: max of our vec norm on Ax, taken over all vectors with norm 1.
(*) = $\max\{||Ax|| / ||x|| : x \in R^n, x \neq 0\}$
= $\max\{||Ax|| : x \in R^n, ||x|| \le 1\}$

A vector norm is compatible with its subordinate matrix norm: for all $A^{m\times n}$ and x^n, $||Ax|| <= ||A||\,||x||$.
For p = 1, 2 and infinity the MN is subordinate to the vector norm. Thus the matrix norm is also compatible with the vector norm.

3) Complex Linear Maps

Complex number + its conjugate > 0.
Complex Matrix/Scalar multiplication works normally. The complex conjugate of its matrix simply has us compute the conjugate of each element (if complex). Denote $\bar{u}$ as conjugate of u, lowercase = vector, uppercase = matrix. k, z are scalar.

$U = v \Leftrightarrow \bar{u} = \bar{v}$, $A = B \Leftrightarrow \bar{A} = \bar{B}$
$\overline{kz} = \bar{k}\,\bar{z}$, $\overline{(ku)} = \bar{k}\,\bar{u}$, $\overline{(Au)} = \bar{A}\,\bar{u}$, $(\bar{A})^T = \overline{(A^T)}$
If a vector, scalar or matrix = its conjugate, then its real.

M4) Standard Inner Product (SIP) – SIP of two vectors u, v ∈ C^n = $\bar{u}^Tv$
Standard Norm: $(\bar{u}^Tv)^{1/2}$. Also, the SIP is non negative – if it is 0 then u = 0.

Complex Eigenvalues and Eigenvectors: The eigenvalues of a real matrix can be complex. This means our eigenvectors are complex too.

4 Least Squares Method

Endomorphism: f of R^n is a linear map f: $R^n \to R^n$, aka domain and codomain are the same.

Automorphism: a bijective endomorphism.
f: $R^n \to R^n$ automorphism ⟺ f is injective (ker(f) = {0}) ⟺ f is surjective (im(f) = R^n)

Projection of a Subspace: Let U ⊂ R^m be an n dimensional subspace generated by an ordered basis $(u_1,\cdots,u_n)$. Let $U = [U_1,\dots,U_n]$
The **orthogonal projection** π_U on U is the following endomorphism:
π_U: $R^m \to R^m$
$v \to \pi_U(v) = U(U^TU)^{-1}U^Tv$
im(A) ⊥ ker(A^T)

We can uniquely decompose vectors.
Let A ∈ $R^{m\times n}$. For all vectors b ∈ R^m, there exists a unique b_i ∈ im(A), and a unique b_k ∈ ker(A^T) such that: $b = b_i + b_k$.

Let A ∈ $R^{m\times n}$ and b ∈ R^m. Suppose Ax = b has no solution for x ∈ R^n, i.e., b ∉ im(A).

LSM finds x ∈ R^n such that $||Ax - b||_2$, or equivalently, $||Ax - b||^2_2$ is minimised.
$||Ax-b||_2$ is minimised when $||Ax-b_i||_2 = 0 \Leftrightarrow Ax = b_i$

M5) Least Squares Method (LSM)

$A^TAx = A^Tb$, solving this gives the solution to the least square problem, as this is when Ax = bi.
Take A = [2 2, 1 2, 2 0], b = $[0, 5, 1]^T$.
Ax = b has **no solution**. So, compute A^TA = $[9\ 6, 6\ 8]^T$, A^Tb= [3, 10]. We proceed with Gaussian Elim and get $x_1 = -1$, $x_2 = 2$.

5) Linear Regression

We have a set of points (y_i, a_i), y a real number, and a is a real vector of dimension n. We want to find the model of best fit with parameters s_0 ∈ R and s ∈ R^n, so the sum of the errors squared is minimized:

$$\sum_{i=1}^m (s_0 + \boldsymbol{s}\cdot\boldsymbol{a_i} - y_i)^2$$

AKA: We require s_0+s . $a_i \approx y_i$
The sum of the squared errors is this:

$$\sum_{i=1}^m (s_0 + \boldsymbol{s}\cdot\boldsymbol{a_i} - y_i)^2 = \|\boldsymbol{Az} - \boldsymbol{y}\|_2^2$$

M6) Linear Regression Method

We construct a Matrix A, with 1 column, and record each of our as row by row, corresponding to our y (which had that data).

$$A = \begin{bmatrix}1 & a_{11} & a_{12} & \cdots & a_{1n}\\ 1 & a_{21} & a_{22} & \cdots & a_{2n}\\ \vdots & \vdots & \vdots & & \vdots\\ 1 & a_{i1} & a_{i2} & \cdots & a_{in}\\ \vdots & \vdots & \vdots & & \vdots\\ 1 & a_{m1} & a_{m2} & \cdots & a_{mn}\end{bmatrix}$$

$$Az-b = \begin{bmatrix}s_0 + s_1a_{11} + \ldots + s_na_{1n} - y_1\\ \vdots\\ s_0 + s_1a_{m1} + \ldots + s_na_{mn} - y_m\end{bmatrix}$$

(b is our vector of ys). Minimizing Az-b gives us our solution. We do this by solving the normal equation $A^TAz = A^Ty$
To be clear:
z = vector of parameters we want to estimate.
A = matrix of data point estimates we're minimizing
b = y = vector of measurements

6) Spectral Decomposition of Symmetric Matrices

Orthogonality of Matrix A = A is invertible and $A^T = A^{-1}$
The transformations performed by an orthogonal matrix can be interpreted as a change of basis or a series of rotations and reflections.

Properties of Orthogonal Matrices:
1) Orthogonal Matrix transformations preserve the Euclidean length of vectors: ∀u ∈ R^n , $||Qu||_2 = ||u||_2$
2) Orthogonal transformations preserve the magnitude of the angle between vectors: ∀u, v ∈ R^n , $\underline{Qu\,Qv} = \underline{u\,v}$
3) det = 1 or -1.
4) All eigenvalues have a modulus of 1

Properties of Symmetric Matrices:
1) A^T = A.
2) If A is a real symmetric matrix, then all its Eigenvalues are real.
3) If A is a real symmetric matrix, then for each Eigenvalue the algebraic multiplicity and geometric multiplicity are equal.
4) If A is an n by n real symmetric matrix, eigenvectors for distinct eigenvalues are orthogonal.

Spectral Theorem:
If A is a real, symmetric matrix then it can be diagonalised like so: **$A = QDQ^T = QDQ^{-1}$**
where: Q is an **orthogonal** matrix and D is a **diagonal** matrix of eigenvalues.

M7) Gram-Schmidt (GS) Process:

If we have a linearly independent set of vectors that are a basis for V, we can use the GS Process to convert this set into an orthonormal basis for V (a set of unit vectors that form a basis for V which are all orthogonal to each other).
1) From left to right, considering 1 to n vectors at a time: The first vector v_1 is orthogonal to everything so far as we haven't considered any other vectors yet. We need to make sure the magnitude is 1 though by dividing our first vector by its magnitude to get $\mathbf{u_1}$.
2) For the 2^{nd} vector v_2, we need to find an orthogonal version to $\mathbf{u_1}$. We do this by replacing v_2 with v_2 – $(v_2.u)u$, and then normalise to get $\mathbf{u_2}$. (aka we replace v_2 with v_2 but removing its projection on the u plane).
3) We do the same thing for 3^{rd} onwards, subtracting its projection onto all the planes from before: eg: $\mathbf{u_3 = v_3 - ((v_3.u_1)u_1) - ((v_3.u_2)u_2)}$
Formula: $u_j = v_j - \mathrm{proj}_{u1}(v_j) - \mathrm{proj}_{u2}(v_j) - \dots - \mathrm{proj}_{uj-1}(v_j)$

M8) Spectral Decomposition Method on Matrix A:

1) Solve the Char Polynomial to get the eigenvalues
2) For each eigenvalue k_i, find the corresponding eigenspace E_i – the dimensions of which are equal to the multiplicity of K_i.
3) For each eigenspace find an **orthonormal basis**
4) Combine these basis columnwise to form the matrix Q – the matrix Q is orthogonal as these basis are all orthogonal to each other.
5) The columns of Q are eigenvectors of A. We write P as the diagonal eigenvalues matrix, and now $A = QPQ^{-1}$

Symmetric matrices perform scaling operations in the direction of their eigenvectors.

7) Singular Value Decomposition

A is real symmetric matrix.
A is **positive definite** iff: ∀x ∈ R^n - {0}, $x^TAx > 0$
It's **positive semi-definite** iff: ∀x ∈ R^n - {0}, $x^TAx >= 0$

Theorem: Positive definiteness in terms of Eigenvalues
1) Positive definite ⇔ all eigenvalues are strictly positive
2) Positive semi definite ⇔ eigenvalues are non negative

Properties of A^TA and AA^T Take arbitrary A ∈ R^{mn}
$A^TA \in R^{nn}$ and $AA^T \in R^{mm}$ are both **symmetric** and **positive-semi definite.**
This gives rise to SVD – a more general decomposition than spectral, which takes more work to get into but still exhibits useful properties.

SVD Definition:
Take arbitrary A ∈ R^{mn}. The SVD of A is any decomposition of the form: **$A = USV^T$**
Where: U ∈ R^{mm} and V ∈ R^{nn} are orthogonal matrices.
S ∈ R^{mn} is a diagonal matrix, S = diag($\sigma_1, \sigma_2, \sigma_3, \dots, \sigma_p$) where p = min(m, n).
$\sigma_1 \ge \sigma_2 \ge \dots \ge \sigma_p \ge 0$ (**We write largest first**)
The values $\sigma_1 \ge \sigma_2 \ge \dots \ge \sigma_p \ge 0$, are known as the singular values of A.

Useful Properties of SVD:
1) SVD: $A = USV^T$
If $U = [u_1, \dots, u_m]$ and $V = [v_1, \dots, v_n]$, then **$A = \sigma_1u_1v_1^T + \sigma_2u_2v_2^T + \cdots + \sigma_ru_rv_r^T$.**
2) There is a link between Rank of a Matrix and SVD, **the rank of A = num singular positive values in S.**
3) Let A ∈ R^{mn}. $||A||_2$ = largest singular value of A.
4) Let A ∈ R^{mn}. **The positive singular values of A are the positive square roots of the eigenvalues of AA^T or A^TA.**
5) The span of the first r columns of U = im(A). The span of the last m-r columns is ker(A), where r is the rank.

M9) Singular Value Decomposition Method

When matrix has more rows than columns:
1) Get the Eigenvalues and Eigenvectors of A^TA. Eigenvalues are all positive. Each singular value is the positive square root of an eigenvalue. Write S - the diagonal matrix of singular values.
2) Construct orthogonal matrix V of Eigenvectors $[v_1,\cdots,v_n]$ corresponding to these singular values in order. 3) r = rk(A).
4)$u_i = 1/\sigma_i A v_i$ (aka, 1 over our singular value, multiplied by A, multiplied by our eigenspace span vector), for 1 <= i <= r.
5) Use the Gram Schmidt Process to turn the matrix U into an orthonormal basis.

When matrix has more columns than rows:
Let B = A^T. Compute SVD of B using the method above.

Sometimes we can't diagonalize some matrices as there aren't enough eigenvectors of A to form a basis for R^n.e.g dim(kernel) of (A – λI) for some EVec =/= the multiplicity. Remember, dim(ker) = m-rank

8) Generalised Eigenvectors

Take a square matrix A ∈ R^{nn}. A non zero vector V ∈ C^n is a generalised eigenvector **of rank m** associated with eigenvalue λ if $(A - \lambda I)^m v = 0$ and $(A - \lambda I)^{m-1} v$ =/= 0 Note that any eigenvector associated with its eigenvalue is a generalised eigenvalue of size 1.
If you take an eigenvector of A associated with λ and multiply it by the matrix A - λI, the result will be the zero vector. If A has a generalized eigenvector of rank 1 associated with λ, then when you multiply it by A - λI, the result will be another eigenvector associated with λ. If A has a generalized eigenvector of rank 2 associated with λ, then when you multiply it by A - λI, the result will be a linear combination of a rank 1 generalized eigenvector and another eigenvector associated with λ. This pattern continues for higher-rank generalized eigenvectors. Example:
Take A = [1 1 1, 0 1 0, 0 0 1]. CP: $(1-\lambda)^3$,
λ_1 = 1 (with AM = 3). We end up with two linearly indep EVecs: $(0, 1, -1)^T$, $(1, 0, 0)^T$. There is one more EVec for this value which can be computed by $(A-I)v_3$ = v_2, and so $v_3 = (0, 0, 1)^T$. **It is not always the case that we can do** $(A-I)v_3 = v_2$ or v_1. A generalised eigenvector u3 can always be found by using a suitable linear combination au1 +bu2 of u1 and u2, i.e., we need to find the generalised eigenvector from the eigenspace generated by the eigenvectors for the eigenvalue λ. For an Eigenvalue with AM = x, we have x Generalised Eigenvectors (remember normal eigenvectors count).

9) Jordan Normal Form

A matrix is in Jordan Normal Form if it is of this form:

$$\begin{bmatrix}J_{k_1}(\lambda_1) & & & 0\\ & J_{k_2}(\lambda_2) & & \\ & & \ddots & \\ 0 & & & J_{k_n}(\lambda_n)\end{bmatrix}$$

where each $J_{ki}(\lambda_i)$ is a Jordan block of size k_i with diagonal (not necessarily unique) coefficient λ_i:

$$J_{k_i}(\lambda_i) = \begin{bmatrix}\lambda_i & 1 & 0 & \dots & 0 & 0\\ 0 & \lambda_i & 1 & \ddots & 0 & 0\\ 0 & 0 & \lambda_i & \ddots & 0 & \vdots\\ \vdots & \ddots & \ddots & \ddots & 1 & 0\\ 0 & 0 & 0 & 0 & \lambda_i & 1\\ 0 & 0 & \dots & 0 & 0 & \lambda_i\end{bmatrix}$$

Even if A ∈ R^{nn}, its JNF might not be in R^{nn} but in C^{nn}.

Blocs in JNF and Multiplicities of Eigenvalues:
The algebraic multiplicity of an eigenvalue λ is the sum of the sizes of blocks with λ on the diagonal. The geometric multiplicity of λ is the number of blocks with λ on the diagonal.

M10) Finding the JNF of a Matrix

1) Find the eigenvalues of A, noting the AMs.
2) For each eigenvalue λ_i compute the eigenspace (ES) E_λ. Note the GMs of each ES and their associated EVecs.
3) If $g_i < a_i$ then we find the missing $a_i - g_i$ generalised eigenvectors $v^k_{i,j} \in C^n$ using gaussian elimination such that: $(A - \lambda_i I)v^k_{i,j} = v^{k-1}_{ij}$. We must do it with all Evecs as each of them might generate part of the Generalised Eigenvectors.

$$B = \left[v_{11}^1, \dots, v_{11}^{k_{11}}, \dots v_{1g_1}^1, \dots, v_{1g_1}^{k_{1g_1}}, \dots v_{m1}^1, \dots, v_{m1}^{k_{m1}}, \dots v_{mg_m}^1, \dots, v_{mg_m}^{k_{mg_m}}\right]$$

5) Write JNF as: J = $\begin{bmatrix}J_{k_{11}}(\lambda_1) & & 0\\ & \ddots & \\ 0 & & J_{k_{mg_m}}(\lambda_m)\end{bmatrix}$
Where each $Jk_{i,j}$ is a jordan Block of size $k_{i,j}$ with diagonal coeff λ_i. AKA, each block $Jk_{i,j}$ is associated with eigenvalue λi, and corresponds to the transformation of each generalised eigenvector v^k_{ij} obtained from the eigenvector $v^{1\,ij}$. We have $B^{-1}AB = J$

M10.1) Example of finding the JNF

1) Take A = [-2 2 1, -7 4 2, 5 0 0]. det(A-λI) = $-\lambda(1-\lambda)^2$. We have $\lambda_1 = 0$, $\lambda_2 = 1$, with AM 1 and 2 respectively.
2) The eigenspace of λ_1 = span $\{[0\ {-1}\ 2]^T\}$. AM = GM, so we're done.
The eigenspace of λ_2 = span $\{[1\ {-1}\ 5]^T\}$. AM < GM, so we must find a generalised eigenvector.
3) We find it by:
$(A - \lambda_i I)v^k_{i,j} = v^{k-1}_{ij}$. $\lambda_2 = 1$, $v_0 = [1\ {-1}\ 5]^T$
so: $(A-I)v^k_{i,j} = v^{k-1}_{ij}$ = [-3 2 1, -7 3 2, 5 0 -1]$v_1 = v_0 = [1\ {-1}\ 5]^T$ solving this yields $v_1 = [0\ 3\ {-5}]^T$.
4) Thus $\{[0\ {-1}\ 2]^T, [-7\ 3\ 2]^T, [5\ 0\ {-1}]^T\}$ is a basis of R^3.
B = [0 1 0, -1 -1 3, 2 5 -5]
5) computing $B^{-1}AB$ gives us = [0 0 0, 0 1 1, 0 0 1] = J.

10) Cholesky Decomposition

A matrix A^{nn} is **lower triangular** ∀i < j, $A_{ij} = 0$
A matrix A^{nn} is **upper triangular** ∀i < j, $A_{ij} = 0$
These matrices exhibit useful properties. The equation Ax = b can easily be solved on them, by first getting x_1, and then x_2 with direct substitution and so on, on lower triangular matrices. For upper triangular we get x_n first, and go backwards.

Additional Properties of Symmetric Matrices:
Let A ∈ R^{nn} be a symmetric matrix.
1) If A is **positive definite**, all its diagonal elements are **strictly positive**.
2) If A is **positive semi-definite**, all its diagonal elements are **non-negative**.
3) If A is positive definite then max $(A_{ii}, A_{jj}) > |A_{ij}|$. If A is positive semi definite then max $(A_{ii}, A_{jj}) >= |A_{ij}|$.
Thus, the largest coefficient of A is on its diagonal.
4) If A is positive definite then the 1x1, 2x2, … mxm… matrices in the upper left corner of A are also positive definite. Same holds for semi-definite.

We can use these to quickly notice non **positive-semi definite** matrices – if we have a symmetric matrix with a negative element, it cannot be PSD. Also: if we see a matrix element $|A_{ij}| >$ max (A_{ii}, A_{jj}) then its not PSD (e.g if the 3^{rd} element in the first row = 3, 1^{st} in 1^{st} row = 2, 3^{rd} in 3^{rd} row = 1, then we violate rule 3 as above).

Cholesky Decomposition: any decomposition of a real square matrix A^{nn} of the form $A = LL^T$, where L is a lower triangular matrix.
If A is not positive semi-definite then it doesn't have a Cholesky Decomposition. If A is positive semi definite then it always has a Cholesky Decomposition. Also there exists a version of L with strictly positive diagonal elements.

M11) Computing the Cholesky Decomposition

We note the dimensions of A. We then choose L to have the same dimensions, set LL^T =, label each of L's indexes with i and j for row and column num. Then, LL^T

$$\boldsymbol{LL^T} = \begin{bmatrix}l_{11}^2 & l_{11}l_{21} & l_{11}l_{31}\\ l_{11}l_{21} & l_{21}^2 + l_{22}^2 & l_{21}l_{31} + l_{22}l_{32}\\ l_{11}l_{31} & l_{21}l_{31} + l_{22}l_{32} & l_{31}^2 + l_{32}^2 + l_{33}^2\end{bmatrix}$$

We can now compute the value of each index by pattern matching to A's elements and solving our simultaneous equations, to yield L.

M12) Using Cholesky Decomposition to solve an equation

Given an equation Ax = b, where A has a Cholesky Decomposition $A = LL^T$, we can solve the equation $LL^Tx = b$ in 2 steps:
1. Let $y = L^Tx$. Solve Ly = b by forward substitution.
2. Solve $L^Tx = y$ by backward substitution to find x

11) QR Decomposition

QR decomposition is a matrix factorisation method, decomposing our matrix A into an orthogonal matrix Q and an upper triangular matrix such that A = QR.

M13) QR Decomposition using the Gram-Schmidt Process

$a_1, \dots, a_n$ are the vector columns of A: $[a_1 \dots a_n]$. Assume they're linearly independent.
1) We apply GS process constructing an orthonormal basis $(e_1 \dots e_n)$ s.t span$\{e_1,\dots,e_n\}$ = span$\{a_1,\dots, a_n\}$.
From the GS formula (computing the orthonormal basis of u_j by subtracting from u_j its projection on all previous bases):
2) **Selecting Q and R:**
Set O = $[e_1, \dots, e_n]$. It is semi orthogonal (e.g $O^TO = I$).

$$\boldsymbol{A} = \boldsymbol{Q}\underbrace{\begin{pmatrix}(\boldsymbol{e_1}\cdot\boldsymbol{a_1}) & (\boldsymbol{e_1}\cdot\boldsymbol{a_2}) & \cdots & (\boldsymbol{e_1}\cdot\boldsymbol{a_m})\\ 0 & (\boldsymbol{e_2}\cdot\boldsymbol{a_2}) & \cdots & (\boldsymbol{e_2}\cdot\boldsymbol{a_m})\\ \vdots & 0 & \ddots & \vdots\\ 0 & \cdots & 0 & (\boldsymbol{e_m}\cdot\boldsymbol{a_m})\end{pmatrix}}_{\boldsymbol{R}}$$

M14) QR Decomposition

Let A = [1 1 0, 1 0 1, 0 1 1]. $u_1 := a_1 = (1, 1, 0)^T$, $a_2 = (1, 0, 1)^T$, $a_3 = (0, 1, 1)^T$.
$e_1 = (1, 1, 0)^T / \sqrt{2}$ (i.e. a but normalized). $e_1 \cdot a_1 = \sqrt{2}$, $e_1 \cdot a_2 = 1/\sqrt{2}$, $e_1 \cdot a_3 = 1/\sqrt{2}$
$u_2 = a_2 - (e_1 \cdot a_2)e_1 = (1/2, -1/2, 1)^T$, $e_2 = (1, -1, 2)^T/\sqrt{6}$. $e_2 \cdot a_2 = 3/\sqrt{6}$, $e_2 \cdot a_3 = 1/\sqrt{6}$
$u_3 = a_3 - (e_1 \cdot a_2)e_1 - (e_2 \cdot a_3)e_2 = (-1, 1, 1)^T/\sqrt{3}$ with $e_3 = (-1, 1, 1)^T/\sqrt{3}$. $e_3 \cdot a_3 = 1/\sqrt{6}$.

$$\boldsymbol{Q} = [\boldsymbol{e_1}, \boldsymbol{e_2}, \boldsymbol{e_3}] = \begin{bmatrix}\frac{1}{\sqrt{2}} & \frac{1}{\sqrt{6}} & \frac{-1}{\sqrt{3}}\\ \frac{1}{\sqrt{2}} & \frac{-1}{\sqrt{6}} & \frac{1}{\sqrt{3}}\\ 0 & \frac{2}{\sqrt{6}} & \frac{1}{\sqrt{3}}\end{bmatrix},$$

$$\boldsymbol{R} = \begin{bmatrix}\boldsymbol{e_1}\cdot\boldsymbol{a_1} & \boldsymbol{e_1}\cdot\boldsymbol{a_2} & \boldsymbol{e_1}\cdot\boldsymbol{a_3}\\ 0 & \boldsymbol{e_2}\cdot\boldsymbol{a_2} & \boldsymbol{e_2}\cdot\boldsymbol{a_3}\\ 0 & 0 & \boldsymbol{e_3}\cdot\boldsymbol{a_3}\end{bmatrix} = \begin{bmatrix}\sqrt{2} & \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}}\\ 0 & \frac{3}{\sqrt{6}} & \frac{1}{\sqrt{6}}\\ 0 & 0 & \frac{2}{\sqrt{3}}\end{bmatrix}, \text{ with } \boldsymbol{A} = \boldsymbol{QR}$$

12) Householder Maps:

Suppose we have a hyper-plane P going through the origin with unit normal u ∈ R^m, i.e., P = {x ∈ R^m : u·x = 0}. The **Householder matrix** defined by $H_u = I - 2uu^T$ induces reflection wrt P.

Properties of a Householder Matrix:
H_u is **involutory**: $H_u = H_u^{-1}u$. H_u is **orthogonal**: $H^Tu = H^{-1}u$.
H_u **preserves the euclidian length of vectors**: $||H_u(x)|| = ||x||$.
The **eigenvalues** are only 1 or -1. The **eigenvectors** are vectors perpendicular to the hyperplane P reflects across – e.g any vectors in the hyperplane P.
H_u **preserves Euclidian length and angles** between vectors as it is orthogonal, because of this any rotations and reflections are orthogonal transformations (because they preserve the aforementioned things). The orthogonal projection Q on the hyperplane P is given by: $Q = I - uu^T$ with $Q^2 = Q$ and $Q = Q^T$.

13) QR Algorithm

Useful to find the eigenvalues of a matrix. Works for most matrices.
1) Set A_0 = A.
2) For k ∈ N, apply the QR decomposition to A_k: $A_k = Q_{k+1}R_{k+1}$ where Q_{k+1} is an orthogonal matrix and R_{k+1} is an upper triangular matrix
3) Set $A_{k+1} = R_{k+1}Q_{k+1}$. Stop after sufficient iterations

Properties of QR Decomposition:
1) For k ∈ N, A_k is similar to A. (similar means $A_k = P^{-1}AP$ ie, it can be obtained from performing a transformation matrix on A).
2) For k ∈ N, we have that $A_k = Q_k^TAQ_k$ from above. So **A_k and A have the same eigenvalues** and **v is an eigenvector of A_k if and only if Q_kv is an eigenvector of A**
3) The sequence (Ak) converges to an upper triangular matrix under certain conditions. This is important because of property 4. (if A is symmetric)
4) **The eigenvalues of an upper triangular matrix are simply its diagonal elements.**
So, the QR decomposition is easily findable, and from this, we can converge it to an upper triangle matrix, from which eigenvalues are easily findable.

Application to Symmetric Matrices
If A is symmetric, so are all the A_k.
If A is symmetric, the algorithm converges , under certain conditions, to a diagonal matrix, hence the eigenvectors of A are in effect the columns of Q˜ k for large enough k.

LU Decomposition:
A non singular matrix A ∈ R^{nn} can be factorised as A = LU
where L is a lower triangular matrix and U is an upper triangular matrix **if and only if A can be reduced to its row echelon form without swapping any two rows.**
If A is non singular and A = LU with the diagonal elements of L being all one, then decomposition is unique.
Also, $A^n = Q_1 \dots Q_nR_1 \dots R_n$
Let A ∈ R^{nn} be a symmetric positive definite matrix with distinct eigenvalues $\lambda_1 > \lambda_2 > \dots > \lambda n > 0$ with eigendecomposition $A = Q\Lambda Q^T$. Suppose Q^T = LU with unit lower triangular L and the diagonal elements of U are positive. Then $A_k \to \Lambda$

14) Fixed Points

Convergence of a Sequence of Real Numbers: Let $(a_n)_{n\in N} \in R^N$ be a sequence of real numbers and l ∈ R. The sequence (a_n) is said to converge to its limit l, $\lim_{n\to\infty} a_n$ = l if and only if: ∀e > 0, ∃N ∈ N such that ∀n > N, $|a_n - l| < e$
The steps we take to show a sequence of real numbers converges:
1) Find the limit l
2) Take e > 0
3) Find N ∈ N such that $|a_n - l| < e$ for n > N, the value of N will usually depend on and decrease with e

Cauchy Sequence: Let $(a_n)_{n\in N} \in R^N$ a sequence of real numbers. Then (a_n) is said to be a Cauchy sequence if and only if: ∀e > 0, ∃N ∈ N such that ∀n, m > N, $|a_n - a_m| < e$
This gives rise to the Cauchy Test: Let $(a_n)_{n\in N} \in R^N$ a sequence of real numbers. Then (a_n) is convergent if and only if it is a Cauchy sequence

Metric Spaces: A metric space is a tuple (S, d) where S is an non-empty set and d is a metric over S meaning: A function d : S × S → R such that:
∀x, y ∈ S, d(x, y) ≥ 0 (positivity)
∀x, y ∈ S, d(x, y) = 0 ⟺ x = y (reflexivity)
∀x, y ∈ S, d(x, y) = d(y, x) (symmetry)
∀x, y, z ∈ S, d(x, y) ≤ d(x, z) + d(z, y) (triangle inequality)

Metric in a normed Vector Space: V is a vector space equipped with the norm ||.||.
Let d be the function defined by $d : \begin{cases} V \times V & \to \mathbb{R}\\ (\mathbf{x}, \mathbf{y}) & \mapsto \|\mathbf{x} - \mathbf{y}\|\end{cases}$. d is a metric space.

From here we will consider the concepts seen in R but generalised to a metric space (S, d). The generalisations will be pretty straightforward, mostly consisting in swapping the absolute value of the difference with the distance.

Convergence in a Metric Space: Convergence in a metric space. Let (S, d) be a metric space and (a_n) a sequence in S. Then a_n is said to converge to a limit l ∈ S iff :
∀e > 0, ∃N ∈ N such that ∀n > N, $d(a_n, l) < e$

Cauchy Sequence in a Metric Space: Let (S, d) be a metric space and (a_n) a sequence in S. (a_n) is a Cauchy sequence iff: ∀e > 0, ∃N ∈ N such that ∀n, m > N, $d(a_n, a_m) < e$

Cauchy Test on a Metric Space: Let (be S, d) a metric space and (a_n) a sequence in S: if (a_n) is convergent, then it is a Cauchy sequence

Complete Space: Let (S, d) be a metric space. Then it is said to be a complete space if and only if every Cauchy sequence in S is also converging in S.

Completeness of the L1, L2 and L∞ Norms: For any k > 0, R^k equipped with any of the three metrics induced by L1, L2 or L∞ is complete.
The space (C[a, b], k · k∞) is complete.

Only Cauchy Sequences Converge in Complete Metric Spaces: Let (S, d) be a complete metric space and (a_n) a sequence in S. (a_n) is converges iff it's a Cauchy Seq.

15) Fixed Point Equations

Fixed Point: Let S be a non-empty set and f: S → S a function from S to itself. p ∈ S is a fixed point iff: f(p) = p

Contraction: Let (S, d) be a metric space and f: S → S. f is called a contraction of S (or a contracting map) **if there exists $0 \le \alpha < 1$** called the **contraction constant** s.t: $\forall x, y \in S$, $d(f(x), f(y)) \le \alpha d(x, y)$.

Fixed Point Theorem: Let (S, d) a complete metric space and f a contraction of S. Then f has a unique fixed point. This leads to the theorem:

Uniqueness of Solutions of Differential Equations: Differential Equations have unique solutions: there exists a unique function x: $(t_0 - d, t_0 + d)$ satisfying: dx/dt = f(x(t), (t)) with $x(t_0) = x_0$.

$$\begin{cases} \frac{dx}{dt} = \dot{x}(t) = f(x,t) \\ x(t_0) = x_0 \end{cases}$$

Where d = min(a,b/(B+bc)) with B = sup{|f(x, t)| : $|x-x_0|<b$, $|t-t_0|<a$}>0

16) Condition Numbers

Condition numbers is the measure of the sensitivity of a problem to small fluctuations in its input. **Definition:** Let P be the problem of interest, d an input, e some perturbation on the input and s(d), s(d + e) the corresponding outputs. Then the condition number of P, κ(P) is:

$$\kappa(P) = \max \frac{\|s(d+\epsilon)\|}{\|\epsilon\|}$$

If our system is g(x) = 0.1x, then k(P) = max ||g(d) – g(d+e)|| / ||e|| = ||0.1d – 0.1d + 0.1e|| / ||e|| = 0.1. So our condition num is constant.

We can use it in the context of linear equations too:

1) x + y = 1.
2) x + ay = 0, where a is some unknown constant. When a =/= 1, our eqs have a solution. When a is close to one, its value drastically changes our solutions: Take a = 0.9999, x = -9999 and y = 10000.

In this example, d = 0.9999, e = -0.0009, s(d) = (-9999, 10000), s(d+e) ≈ (-999, 1000).
So, k(P) = max||s(d) – s(d+e)|| / e = ||(-9999, 10000) - (-999, 1000)||/-0.0009 = ||(-9000, 9000)||/-0.0009.
We can choose different norms to evaluate the top part, with the l_1 norm we end with $2*10^7$, with the l2, we get root $2 * 10^7$
To compute **relative** condition number we do:

$$\frac{\|s(d)-s(d+\epsilon)\|}{\epsilon} \; \frac{\|d\|}{\|s(d)\|}$$

Condition Numbers on Square Matrices: A be a non-singular matrix, it's Condition Number is: κ(A) = $\|A^{-1}\| \|A\|$ = **Relative Condition Number.**

For a linear equation, Ax = b, the condition number of the problem = condition number of the matrix A. Big norm != ill conditioned. We might just be working with big numbers!!

Non square matrices:
Ax = b⇔$A^TAx = A^Tb$⇔x = $(A^TA)^{-1}A^Tb$. The matrix $(A^TA)^{-1}A^T$ is known as the **pseudo inverse** of A, and denoted A^+. We can use this to compute the condition number of a non square matrix A: k(A) = $\|A^+\| \|A\|$

Conditioning of a Problem:
To decide if a problem is well or ill conditioned, we can use this Rule of Thumb to guide us: For a condition number κ(A), you lose about **log10(κ(A)) significant figures** in accuracy. If we have an accurate implementation to 12 dp, if the condition number is 1,000,000 then we'll lose about 6 dp in precision.

17) Iterative Solutions of Linear Equations

$A \in R^{nn}$ and $b \in R^n$. We want to solve Ax = b. Via Gaussian Elimination O(n³),too expensive. Instead approximate the solution with iteration.

Theorem of Convergence of Sequences: Let $M \in R^{nn}$ and $c \in R^n$. Let ||.|| be a consistent norm on R^{nn}, if ||M|| < 1 then the sequence (x_k) defined by $x_{k+1} = Mx_k + c$ converges for any starting point x_0.
So, to solve Ax+b we solve the equation x = Mx+c. Note how this is in the form required for our Theorem. M = $-G^{-1}R$, c = $G^{-1}b$. A solution to the equation, thanks to fixed point theory would be when $x_{k+1} = x_k$

Choosing Efficient Choices of Splitting:
1. $G^{-1}R$ and $G^{-1}b$ are easy to compute
2. ||M|| is small (for fast convergence to a solution).
We'll assume A has no 0s on it's diagonal (if it does we can do a change of basis to achieve it):

M15) Change of Basis to solve Ax = b
We want to solve Ax = b with A = [0 1, 1 0]. It has 0s on its diagonal.
Consider C = 1/√2 (1 1, 1 -1). Then C^{-1} = C and: Ax = b ⇔ $C^{-1}Ax = C^{-1}b$ ⇔ $(C^{-1}AC)C^{-1}x = C^{-1}b$ ⇔ $(C^{-1}AC)C^{-1}x = C^{-1}b$. We select C such that it has a diagonal: here $C^{-1}AC$ = [1 0, 0 -1] = B. By denoting $C^{-1}x$ = y and $C^{-1}b$ = c it is equivalent to solve **By = c** where B can be split according to the given method. Solving this equation will give y and x can be retrieved as x = Cy

Methods of splitting:
Common Splitting:
Then we can write A = D + L + U, where D is the diagonal of A, L, U are the lower and upper triangle parts of A respectively.

1) The Jacobi Method using the ADLU splitting from earlier: we denote R = L + U. Suppose that for $b \in R^n$ we want to solve the equation Ax = b.
Then Ax = b ⇔ (D+R)x = b ⇔ x = $-D^{-1}Rx + D^{-1}b$ ⇔ x = Mx + c. (M = $-D^{-1}R$, c = $D^{-1}b$).
This splitting is good as x^{k+1} is easily computed as: D^{-1} = reciprocal of each diag element as diags, R = the combination of the upper and lower triangle matrices, so: Mx = $-D^{-1}Rx$ =

$$-D^{-1}\begin{bmatrix} \sum_{j\neq 1} a_{1j}x_j \\ \sum_{j\neq 2} a_{2j}x_j \\ \vdots \\ \sum_{j\neq n} a_{nj}x_j \end{bmatrix} = \begin{bmatrix} -\frac{1}{a_{11}}\sum_{j\neq 1} a_{1j}x_j \\ -\frac{1}{a_{22}}\sum_{j\neq 2} a_{2j}x_j \\ \vdots \\ -\frac{1}{a_{nn}}\sum_{j\neq n} a_{nj}x_j \end{bmatrix}$$

(This makes sense as $-D^{-1}$ is the negative of the reciprocal of each diagonal element, and we multiply this by L+U, which is a matrix of elements except the diagonal).
And since $x^{k+1} = Mx^k + c$, and C = $D^{-1}b$,

$$x_i^{(k+1)} = \frac{1}{a_{ii}}\left(b_i - \sum_{j\neq i} a_{ij}x_j^{(k)}\right)$$

Computationally to calculate the ith element of $x^{(k+1)}$ we only need the b^k and x^k and i^{th} row of A, so we can parallelize this – dispatching each row ot a different thread.

2) The Gauss Seidel Method: use the ADLU splitting, but do A = (D+L) + U.
Then Ax = b ⇔ (D+L)x = b – Ux ⇔ x = Mx + c, Where M = $-(D+L)^{-1}U$, c = $(D+L)^{-1}b$.
D+L is a lower triangle matrix with a diagonal D, U is an upper triangle matrix.
And then $x^{k+1} = Mx^k + c$ ⇔ $(D+L)x^{k+1} = -Ux^k+b$

$$\sum_{j\le i} a_{ij}x_j^{(k+1)} = -\sum_{j>i} a_{ij}x_j^{(k)} + b_i$$

So the i^{th} element of $x^{(n+1)}$ can be computed thanks to A, $x^{(n)}$, b and the k^{th} elements of $x^{(n+1)}$ for k < i. As the update is computed with more recent quantities, the convergence is faster.

Convergence of these methods:
A square matrix is **Strictly Row Diagonally Dominant** if $\forall i, |a_{ii}| > \sum_{j\neq i} |a_{ij}|$

If A is strictly row diagonally dominant then the Jacobi and Gauss-Seidel methods will converge., with the G-S method converging faster

M16) Iteratively Solving Linear Equations
A = [5 2, 3 4] ∈ R^{2x2} . We work in R² with the L1-norm. Take G = [5 0, 0 4]. R = [0 2, 3 0]. Since M = $-G^{-1}R$ = -[0 2/5, 3/4 0]. $\|M\|_1$ = ¾ < 1, we can find an iterative solution.

Other Results: 1) Influence of the Condition Number on Convergence: These techniques converge faster for low condition numbers. If big, it might even diverge.**2) Irreducible:** A matrix is irreducible if it can't take the form: $\begin{bmatrix} A_{11} & A_{12} \\ 0 & A_{22} \end{bmatrix}$ where A_{11} and A_{22} are block matrices.
3) If A is weakly row diagonally dominant and irreducible, Jacobi and G-S still converge.

18) Iterative Techniques for computing Eigenvectors and Eigenvalues

It's O(n⁴) to find eigenvectors, as we have to find roots of Char Polyn, and then use Gaussian Elim.
We need numerical methods:
A **dominant eigenvalue** of A is an eigenvalue with the largest modulus, a dominant eigenvector is an eigenvector of a dominant eigenvalue.

Power Iteration Theorem:
Let $A \in R^{nn}$ be a diagonalisable matrix with eigenvalues of distinct modulus. Let λ ∈ R be the dominant eigenvalue.
We consider the sequence (x_k) defined by $x_{k+1} = Ax_k / \|Ax_k\|$ and $x_0 \in R^n \setminus \{0\}$. Then, as k→infinity, $x_k \to v$, and $\|Ax_k\| \to |\lambda|$.
This convergence to v is not rigorous – if λ is negative then we oscillate between v and –v.

M17) Power Method:
A = [1 -3 2 4, 2 4 1 -3, 4 2 -3 1]
EigenVs are 2, 4, 6, -8, with eigenspaces: E_2=span{[-1 1 -1 1]} E_4=span{[1 1 1 1]} E_6=span{[-1 1 -1 1]}E_8=span{[-1 -1 1 1]}
Considering the sequence $X_{n+1} = Ax_n / \|Ax_n\|_\infty$, and taking x_0 = [1 0 0 0]. We keep iterating and eventually x_{15} = [-0.97 -1 0.97 1], the eigenvector corresponding to the eigenvalue of the largest modulus.

Power Iteration has **limitations:**
1. x_0 is chosen at random, it's possible for it to be such that α1 = … = αp = 0. In this case, the iteration yields second dominant eigenvalue and eigenvector. We need to make sure there's a non zero component in the corresponding eigenspace – usually happens in computers due to flop error.
2. Eigenvalues might not have distinct modulus, in which case the power iteration will converge to a linear combination of the corresponding eigenvectors.
3. Convergence might be slow if dominant evec isn't too dominant.

18.1) Inverse Power Iteration
Lets us find the smallest eigenvalue and its eigenvector. We do this by taking the inverse of A, and then performing power method. This yields the eigenvalue 1/λ, where λ = smallest eigenvalue of A.

Inverse Power Iteration Theorem:
Let $A \in R^{nn}$ be a diagonalisable non-singular matrix with eigenvalues of distinct modulus. Let λ ∈ R be the eigenvalue with the smallest modulus. We consider the sequence (x_k) defined by $x_{k+1} = A^{-1}x_k / \|A^{-1}x_k\|$ and $x_0 \in R^n \setminus \{0\}$. Then:

$$x_k \xrightarrow[k\to\infty]{} v \text{ and } \|A^{-1}x_k\| \xrightarrow[k\to\infty]{} \frac{1}{|\lambda|}$$

18.2) Shifts
Let $A \in R^{nn}$ and s ∈ R. The matrix A-sI is called the **shifted matrix** and with the following property:
Let $A \in R^{nn}$ and s ∈ R. λ ∈ R is an eigenvalue of A if and only if λ-s is an eigenvalue of A-sI with the same eigenvectors. Thanks to the shifted matrix, it is possible to focus on a particular eigenvalue of the matrix A.
Theorem of Shifts: $A \in R^{nn}$ is a diagonalisable matrix with eigenvalues of distinct modulus and s ∈ R. Suppose that the shifted matrix A – sI is non-singular (it is non-singular as long as s is not an eigenvalue of A), then by performing inverse power iterations, we can find the eigenvalue of A that it the closest to s.
Thus we can use the Inverse Power Method and Shifts to discover all eigenvalues.

18.3) Rayleigh Quotient
Let $A \in R^{nn}$ and $x \in R^n \setminus \{0\}$. Rayleigh quotient R(A, x) is given by: x^TAx / x^Tx.
When using an iterative technique to find the EVal/EVec of a matrix the Rayleigh quotient monitors the convergence to the eigenvalue as it gives access to an approximation of the eigenvalue (rather than the modulus). Applying the Rayleigh quotient to the sequence written before, we have: R(A, x0) = 1, with values in between R(A, x15) = −7.99.

18.4) Deflation
We can use the power method to get the max eigenvalue. But it's a problem to get the next eigenvalue, as we have to perform a shift – but we don't actually know what to shift by. Instead we can **deflate our matrix into an $A^{(n-1)(n-1)}$** matrix, removing the dominant eigenvector. Let $\lambda_1, \ldots, \lambda_n$ be the eigenvalues of A, ordered according to their magnitude with λ_1 being the dominant one. Define H as a non-singular matrix such that: $Hx_1 = \alpha e_1$ where α ∈ R \ {0} and e1=[1,0,..., 0]ᵀ is the first vector in the standard basis. We then have: $HAH^{-1}e_1 = HAx_1/\alpha = H\lambda_1x_1/\alpha = \lambda_1e_1$. The first column of HAH^{-1} is$[\lambda_1, 0, \ldots 0]^T$ and so $HAH^{-1} = [\lambda_1\ b^T, 0\ B]$, where B is an (n-1) by (n-1) matrix with eigenvalues $\lambda_2, \ldots, \lambda_n$.
They also have the property:
Let λ_2 be the second dominant eigenvalue of A with $\lambda_2 =/= \lambda_1$. An eigenvector x_2 of A corresponding to eigenvalue λ_2 is: $x_2 = H^{-1}[\beta\ z_2]$ With $\beta = b^Tz_2 / \lambda_2 - \lambda_1$ and z_2 is a dominant eigenvector of B

M18) Deflation Method
Let A = $\begin{bmatrix} 1 & -3 & 2 & 4 \\ -3 & 1 & 4 & 2 \\ 2 & 4 & 1 & -3 \\ 4 & 2 & -3 & 1 \end{bmatrix}$

1) We'll apply deflation to this using the **Householder matrix**.
First we need to construct it:
H = I = $2uu^T / u^Tu$
We take x_1 = [1 1 -1 -1]ᵀ, which is a vector in the eigenspace of our biggest eigenvalue.
u = $x_1 + \|x_1\|_2 e_1$
We take u = [3 1 -1 -1]ᵀ and so H = $\frac{1}{6}\begin{bmatrix} -3 & -3 & 3 & 3 \\ -3 & 5 & 1 & 1 \\ 3 & 1 & 5 & -1 \\ 3 & 1 & -1 & 5 \end{bmatrix}$

Since H is a householder matrix, HAH = HAH^{-1} = $\begin{bmatrix} -8 & 0 & 0 & 0 \\ 0 & 4 & 4/3 & -4/3 \\ 0 & 4/3 & 10/3 & 0 \\ 0 & -4/3 & 0 & 14/3 \end{bmatrix}$

And we receive a matrix B, and -8 is our largest eigenvalue (as we knew). bᵀ = [0 , 0 , 0]. Performing power iterations on B, the algorithm converges to z_2 = [1 1/2 -1]ᵀ, with corresponding eigenvalue 6. In this case, $\beta = b^Tz_2 / \lambda_2 - \lambda_1$ = 0 and

$$x_2 = H^{-1}\begin{bmatrix} 0 \\ 1 \\ 1/2 \\ -1 \end{bmatrix} = H\begin{bmatrix} 0 \\ 1 \\ 1/2 \\ -1 \end{bmatrix} = \frac{3}{4}\begin{bmatrix} -1 \\ 1 \\ 1 \\ -1 \end{bmatrix}$$

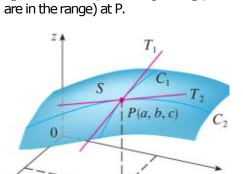
which is an eigenvector of A.

19) Functions of Several Variables
Functions of N Variables are rules which assign to each ordered tuple of real numbers $(x_1, x_2, \ldots x_n)$ a unique real number denoted by $f(x_1, x_2 \ldots x_n)$.
Range = set of values f takes, domain = set of values taken by params. The graph of f is the set of all points (x, y, z) in R^{n+1}.

19.1) Level Contours
The level curves of a function on two variables are the curves with equation ***f*(*x*, *y*) = *k***, where ***k*** is a constant (in the range of *f*) A level curve *f*(*x*, *y*) = *k* is the set of all points in the domain of *f* at which *f* takes on a given value *k*. These look like rings at a height *k*, around the parts of the graph which assume that value.
Like contour maps, the surface is steep where level curves are close.

19.2) Partial Derivatives
For a function of two variables, the partial derivatives are denoted by
$f_x(x, y) = \lim_{h\to0} f(x+h, y) - f(x,y)/h$
$f_y(x, y) = \lim_{h\to0} f(x, y + h) - f(x, y)/h$
(AKA just differentiate with respect to 1 and treat the other as a constant).

Geometrically, the partial derivatives of a 2 valued functions are tanget lines that are constant to the dimension which they "ignore" (as seen by T_1 and T_2 which are tangents to C_1 and C_2 (both curves are in the range) at P.



They represent the rate of change of z, with respect to x or y, when the other is fixed. The higher derivatives are computed as imagined
Clairaut's theorem: Suppose *f* is defined on a disk D that contains the point (*a*, *b*). If the functions *fxy* and *fyx* are both continuous on D then ***fxy*(*a*, *b*) = *fyx*(*a*, *b*)**

19.3) Directional Derivatives
For a function *z* = *f*(*x*, *y*), the partial derivatives *fx* and *fy*, represent the rates of change in *z* in the *x* – and *y* – directions, i.e., in the directions of unit vectors **i** and **j**. (calculated in the same way as before).
The rate of change of an arbitrary unit vector can be found with basic trig.
Our tangent line represents the rate of change of z at a specific point, the direction of which depends on the variable we diff with respect to. (e.g, how much do we move in x and y as we progress?)
Theorem: If *f* is a differentiable function of *x* and *y*, then *f* has a directional derivative in the direction of a unit vector **u** = *a*, *b*:
$D_uf(x, y) = f_x(x, y)a + f_y(x, y)b$
If the unit vector **u** makes an angle *θ* with the positive *x*-axis, then we can write **u** = <cos *θ*, sin *θ*> and the formula becomes $D_uf(x, y)$ = $f_x(x, y)\cos\theta + f_y(x, y)\sin\theta$

M19) Calculating the Directional Derivative
1. The formula is:
$\mathbf{D_uf(x, y) = f_x(x, y)cos\theta+f_y(x, y)sin\theta}$
2. So we partial differentiate wrt x, and y and then sub in, along with our theta.
3. We will get a value which represents how much we change in the direction u, per unit u.
Example:
For a function $f_{(x,y)} = x^3 - 3xy + 4y^2$, find the directional derivative
$D_uf(x, y)$ at (1,2) where **u** is the unit vector at an angle $\theta = \pi/6$:
1) $D_uf(x, y) = f_x(x, y)\cos\theta+f_y(x, y)\sin\theta$
$D_uf(x, y) = f_x(x, y)\cos\pi/6 +$
$f_y(x, y)\sin\pi/6 =$
$(3x^2 - 3y)\sqrt{3}/2 + (-3x + 8y)\,1/2 =$
$= ½[3\sqrt{3}x^2 - 3x + (8 - 3\sqrt{3})y]$
Subbing (1, 2) we get (13 - 3√3) / 2

20) The Gradient Vector
$D_uf(x,y) = \langle f_x(x,y), f_y(x, y)\rangle \cdot u$
The first vector in the dot product is the gradient vector, denoted as grad *f* or **∇*f***.
It is given by this formula:
$\nabla f(x,y) = \langle f_x(x, y), f_y(x, y)\rangle$
$= \partial f/\partial x\mathbf{i} + \partial f/\partial y\mathbf{j}$

M20) Calculating Gradient Vector
1. To calculate, just differentiate wrt x – this is our i component, and differentiate wrt y, this is our j component.
Example:
f(x, y) = x²y³ - 4y. Find the directional derivative at (2, -1) in the direction of v = 2i + 5j. Taking partial derivatives and subbing in (2,-1), we get ∇*f*(2,-1)=-4**i**+8**j**
This is our gradient vector, and we can continue to get the directional derivative by multiplying it by the directional unit vector, and multiplying u and grad f, then finding the magnitude

M21) Maximising the Directional Vector
We have a function *f* of two/three variables. To maximise it the directional vector (out of all possible directions), we can use this theorem:
Max($D_\mathbf{u}f$) = Max(∇*f*·**u**) = Max(|∇*f*||**u**|cos*θ*) = Max(|∇*f*| cos *θ*) = |∇*f*|.
The max value of cos is 1, at *θ* = 0. So the max value of $D_\mathbf{u}f$ is |∇*f*| when *θ* = 0. AKA it's just the magnitude of the gradient vector – this makes sense as the gradient vector and directional unit vector are going in the same path – the angle between them is 0.

Example: f(x, y) = xeʸ. The rate of change of f at the point P(2,0) in the direction from P to Q(1/2, 2).
∇*f*(*x*, *y*) = <*fx*, *fy*> = <eʸ, xeʸ>.
So ∇f(2, 0) = <1, 2>. (Gradient vector)
The unit vector in direction PQ = <-3/2, 2> is <-3/5, 4/5>. (Directional Unit vector)
So the rate of change of f in the direction P to Q is: $D_uf(2, 0)$ =∇*f* (2,0) · u = -3/5 * 4/5 (Directional Derivative).
To find the max rate of change now, we know f increases fastest in the direction of the gradient vector, ∇*f* (2,0) = <1,2>, and the magnitude of this is √5
Tangent Plane: The tangent plane to a surface at a given point is a flat plane that locally approximates the surface at that point, and is parallel to the plane spanned by the partial derivatives of the surface at that point. We have these for 3 valued functions.

The gradient vector $\nabla F(x_0, y_0, z_0)$ gives the direction of fastest increase of ***f***.
$\nabla F(x_0, y_0, z_0)$ is orthogonal to the level surface *S* of *f* through *P*.

If we consider a topographical map of a hill and let *f*(*x*, *y*) represent the height above sea level at a point with coordinates (*x*, *y*) then **a curve of steepest ascent** can be drawn as shown in the figure by making it perpendicular to all of the contour lines.
Maxima and Minima:
Local Maximum & Local Minimum I already know. Absolute Max and Absolute Min I already know.

M22) Determining whether a function has extreme value at a Critical Point:
We take partial derivatives, and see if they're equal to 0. To be conclusive about our result, we must take the second partial derivative:
Second Derivative Test:
Suppose the second partial derivatives of *f* are continuous on a disk with centre (*a*, *b*), and suppose that $f_x(a, b) = 0$, $f_y(a, b) = 0$ and [that is, (*a*, *b*) is a critical point of *f*].

$$D = \begin{vmatrix} f_{xx} & f_{xy} \\ f_{yx} & f_{yy} \end{vmatrix} = f_{xx}f_{yy} - (f_{xy})^2$$

where ***fxy* = *fy***x
With the matrix denoted as a Hessian Matrix, and D = det.
a. If *D* > 0 and $f_{xx}(a, b) > 0$, then *f*(*a*, *b*) is a local minimum
b. If *D* > 0 and $f_{xx}(a, b) < 0$, then *f*(*a*, *b*) is a local maximum
c. If *D* < 0, then *f*(*a*, *b*) is not an extrema but a saddle point
d. If *D* = 0, then the test is inconclusive and *f*(*a*, *b*) could be a local maxima, minima or a saddle point.

This test generalizes to a test based on the eigenvalues of the function's Hessian matrix at the critical point P. In particular, assuming that all second-order partial derivatives of *f* are continuous on a neighbourhood of a critical point P, then:
If the eigenvalues of the Hessian at P are all positive, then P is a local minimum
If the eigenvalues are all negative, then P is a local maximum
If some are positive and some negative, then the point is a saddle point.
If the Hessian matrix is singular, or for a case not listed above, the second derivative test is inconclusive.

23) Gradient Based Optimization
A quadratic n variable function looks like:
$Q(x_1, \ldots, x_n) = b_{11}x_1^2 + b_{12}x_1x_2 + \cdots + b_{ij}x_ix_j + \cdots + b_{nn}x_n^2$
We can write this more conveniently with matrices: Q(x) = xᵀAx, where x is a column vector with elements x_i, 1 <= i <= n, and A is a symm. matrix with $a_{ij} = b_{ij} + b_{ji} / 2$
Note $x_ix_j = x_jx_i$. Eg:
$Q(x_1,x_2) = x_1^2 - ax_1x_2 + bx_2x_1 - cx_1x_3 + 5x_2^2$
is expressed as
x = $[x_1, x_2, x_3]$ A = $\begin{bmatrix} 1 & \frac{a+b}{2} & -\frac{c}{2} \\ \frac{a+b}{2} & 5 & 0 \\ -\frac{c}{2} & 0 & 0 \end{bmatrix}$
And then $x^TAx = Q(x_1, x_2)$

23) Gradient Based Optimization
Iterative algorithms that find the local min or max using good initial guesses.
23.1) Gradient Descent:
$\nabla f(\underline{x}^{(k)})$ is a gradient of an n-dimensional function f($\underline{x}$) at iteration k.
The Gradient Descent Formula:
Gradient descent* at iteration *k* + 1:
$\underline{x}^{(k+1)} = \underline{x}^{(k)} - \alpha\nabla f(\underline{x}^k)$, where *α* > 0 is a fixed step size.
(AKA, to compute the next point, we take the current point and subtract the gradient and some step size).
Steepest Descent is a faster version – as we minimise what we have left.
$\underline{x}^{(k+1)} = \underline{x}^{(k)} - \alpha_k \nabla f(\underline{x}^k)$
$a_k = \text{argmin } f(\underline{x}^{(k)} - \alpha\nabla f(\underline{x}^k))$
Graphically, we compute the gradient at our point, then go the opposite direction by step size a, and repeat.
For quadratic functions of the form
f(x) = 1/2xᵀQx – bᵀx where
Q^{nn} is real symmetric positive definite matrix, b^n is a real vector, and x^n is a real vector the gradient of the function is:
∇f(x) = Qx – b, Hessian = ∇²f(x) = Q.
With this, the optimal step size a at iteration k is: $a_k = (g^k)^T(g^k) / ((g^k)^T)Hg^k$
g^k = ∇f(x) and H = Q = ∇²f(x).

M23) Steepest Descent on Quadratic
1) Compute the gradient function, g^k = $\boldsymbol{\nabla f(x)}$, and H which is computed in M22 by taking second derivatives wrt specific variables in specific orders.
2) Sub in the x^0 into the gradient function, then compute $a_k = (g^k)^T(g^k) / ((g^k)^T)Hg^k$.
3) Then generate x_1 as $x_0 - a_0\,\boldsymbol{\nabla f(x^0)}$
3) Repeat until a constraint is met.
Example: One step
We apply Steepest Descent to **minimize**
$f(x) = x_1^2+x_2^2+2x_2+4$. Choose x^0= [2 1]ᵀ
The gradient g^k= $\boldsymbol{\nabla f(x)}$ = $[2x_1, 2x_2+2]^T$
H = $\boldsymbol{\nabla^2 f(x)}$ = [2 0, 0 2]
The gradient at x^0=[2(2), 2(1)+2]=[4, 2]ᵀ
The step length a = $(g^0)^T(g^0) / ((g^0)^THg^0$ = [4 4][4 4]ᵀ/ [4 4][2 0, 0 2][4 4]ᵀ = 32/64.
X¹ = [2 1]ᵀ – ½[4 4]ᵀ = [0 -1].
We continue and find the next gradient is 0, so stop as we found a minimum point.
On a computer, because of flop, we wouldn't actually have 0. Common stopping criteria include:
1) ||∇*f*|| <e – gradient becomes close to 0
2) $\|f(x_{k+1}) - f(k)\| < e$ – terms don't change much
3) $\|f(x_{k+1}) - f(k)\| / \max(1, |f(x_k)|)$ – a relative measure, or we could take the norm of x_k instead of function val.
Rather than computing best step size, we could use a constant, a diminishing one, a small minimization.

24) Conjugate Gradient Method
Minimizes a quadratic function
F(x) = ½ xᵀAx – bx, where A^{nn} is a real symmetric positive definite matrix, and b^n is a real vector. The solution to the minimization problem is equivalent to finding x in ∇*F*(*x*) = 0, ie: solving Ax-b = 0.
A-conjugate direction: $\{d_0, \ldots d_{n-1}\}$, $d_i \in \mathbb{R}^n$ are search direction vectors orthogonal to each other wrt A if $d_i^TAd_j = 0, \forall i \neq j$.
$d_i^TAd_j$ is also written as $\langle d_i, d_j\rangle_A = \langle d_i, Ad_j\rangle = \langle Ad_i, d_j\rangle = \langle d_j, A^Td_j\rangle$
These directions $d_0 \ldots d_n$-1 are A-orthogonal or A-conjugate directions. If A = I these vectors are orthogonal in the normal way.

M24) Using the Conjugate Gradient Method
1) Use a residual vector r_0 as the initial search direction d_1. The method of selecting d_k changes afterwards.
$r_0 = b - Ax_0$
2) Calculate the scalar a_1 with $r_0^Tr_0 / d_1^TAd_1$
3) Now x_1 can be computed with the formula: $x_1 = x_0 + a_1d_1$
4) Compute the residual r_1 for the next iteration: $r_0 - a_1Ad_1$
5) We are now at iteration 2. Calculate the Scalar $\beta_2 = -r_1^Tr_1 / r_0^Tr_0$
6) Use β_2 to compute the next search direction $d_2 = r_1 - \beta_2d_1$
7) Calculate a_2 using $d_2 = r_1^Tr_1 / d_2^TAd_2$
8) Find $x_2 = x_1 + a_2d_2$
9) Compute the residual $r_2 = r_1 - a_2Ad_2$ and use a norm (usually L_2) to see if we have small enough value to terminate. If not, repeat steps 5-9.

Example:
1) Use residual vector r_0 as the initial search direction d_1.
A = [5 1, 1 8], x = $[x_1\ x_2]^T$, b = [3 2]T $r_0 = b - Ax_0$ = [3 2]ᵀ – [5 1, 1 8][2 1]ᵀ = [-8 -8]ᵀ. $d_1 = r_0$ = [-8 -8]ᵀ
2) Calculate scalar $a_1 = r_0^Tr_0 / d_1^TAd_1$ = [-8, -8][-8 -8]ᵀ [-8, -8] [5 1, 1 8][-8 -8]ᵀ= 2/15
3) Now x_1 (improved approximation) can be reached x_1 = [2 1]ᵀ + 2/15[-8 -8]ᵀ = [0.933, -0.0667]
4) Compute the residual for the next iteration $r_1 = r_0 - a_1Ad_1$ =[-8 -8]ᵀ – 2/15[5 1, 1 8][-8 -8]ᵀ = [-1.6 1.6]ᵀ
5) Calculate the scalar B_2:
[-1.6, 1.6] [-1.6 1.6]ᵀ / [-8, -8][-8 -8]ᵀ = -0.04
6) $d_2 = r_1 - B_2d_1$ = [-1.6 1.6] + 0.04[-8 -8] = [-1.92 1.28]
7) $a_2 = r_1^Tr_1 / d_2^TAd_2$ =
[-1.6, -1.6][-1.6 1.6]ᵀ / [-1.92, 1.28][5 1, 1 8][-1.92 1.28] = 0.1923
8) Find x_2 and see if we have reached our solution:
$x_2 = x_1 + a_2d_2$=[0.9333,0.0667]+0.192[-1.92 1.28]=[0.5641, 0.1794]ᵀ
9) $r_2 = r_1 - a_2Ad_2$ = [-1.6 1.6] – 0.1923[5 1, 1 8][-1.92 1.28]ᵀ
= 1.0e – 0.4[-0.64, 0.64]. $\|r_2\|$ = 9.051e-0.5

25) Linear Programming
The core is **the Simplex Method** which is an error free procedure to maximise a problem that can solve a constraints maximization / minimization problem quickly. These problems are easily graph solvable.

M25) The Simplex Method
1) Write out the list of constraints from our problem
2) Use slack variables; tighten inequality constraints to equality constraints
3) Put the problem into tableau form
4) Choose the most negative entry in the z row and mark that column, evaluate the ratios off the solution column and the positive entries in the our chosen column. Choose the smallest of these and mark that row.
5) Divide the row by our chosen value (denoted by our row and column). Then use Gaussian Elimination to clear every other entry in the y column.
6) Repeat step 4 and 5, until all the entries in the z row are non negative. Then our optimum is achieved.

A manufacturing company has two circuit boards, R1 and R2, with different components. R1 has 3 resistors, 1 capacitor, 2 transistors, and 2 inductors, while R2 has 4 resistors, 2 capacitors, and 3 transistors. The company has 2400 resistors, 900 capacitors, 1600 transistors, and 1200 inductors for a day's production. We make a profit of 5p on R1 and 9p on R2. Calculate how many of each circuit board the company should produce daily to maximize its overall profits.
1) Constraints: We know x >=0, y >= 0. Maximise z(profit) = 5x + 9y
3x+4y<=2400(a), x+2y<=900 (b), 2x+3y<=1600 (c), 2x<=1200 (d)
2) Constraints are now: 3x+4y+r = 2400(a_s), x+2y+s = 900 (b_s),
2x+3y+t = 1600 (c_s), 2x + t = 1200 (d_s)

3)

		Non-basic variables		Basic variables				
		x	*y*	*r*	*s*	*t*	*u*	Solution
Objective function *z*		−5	−9	0	0	0	0	0
Basic variables	*r*	3	4	1	0	0	0	2400
	s	1	2	0	1	0	0	900
	t	2	3	0	0	1	0	1600
	u	2	0	0	0	0	1	1200

4) We choose column y, and value 2 (row s), because the ratios are 2400/4, 900/2, 1600/3, of which 900/2 is the smallest.
5) (*z* row) + 9 ∗ (*y* row) • (*r* row) − 4 ∗ (*y* row) • (*t* row) − 3 ∗ (*y* row),

	x	*y*	*r*	*s*	*t*	*u*	Solution
z	− 1/2	0	0	9/2	0	0	4050
r	1	0	1	−2	0	0	600
y	1/2	1	0	1/2	0	0	450
t	1/2	0	0	−3/2	1	0	250
u	2	0	0	0	0	1	1200

6) elided, but we get *z* = 4300, *r* = 100, *y* = 200, *x* = 500 and *u* = 200

The Dual LP Problem
Every linear programming problem is associated with another problem, known as it's **Dual Problem**.
The Duality Principle: The objective function of the minimisation problem reaches its minimum if and only if the objective function of its dual reaches its maximum. And when they do, they are equal.
We can find the dual maximization problem by taking the transpose, or a maximization or minimization problem. It might be easier to solve, and ideally we want to solve minimization problems as they're doable with the above.

Primal Problem: ➔
Minimise: $Z = 12x_1 + 16x_2$
Subject to the constraints:
$x_1 + 2x_2 \ge 40$
$x_1 + x_2 \ge 30$
$x_1 \ge 0; x_2 \ge 0$

Dual Problem:
Maximise: $Z = 40y_1 + 30y_2$
Subject to the constraints:
$y_1 + y_2 <= 12$
$2y_1 + y_2 <= 16$
$y_1 >= 0; y_2 >= 0$

Original matrix (minimisation)

1	2	40
1	1	30
12	16	0

Transpose matrix (maximisation)

1	1	12
2	1	16
40	30	0