Complex numbers:

- **A complex number C is the number that can be represented in the form C = a + bi where $i^2 = -1$.**
- **i^n can take four values depending on the result of n%4**
  - n%4 = 0 ⇒ $i^n = 1$, n%4 = 1 ⇒ $i^n = i$, n%4 = 2 ⇒ $i^n = -1$, n%4 = 3 ⇒ $i^n = -i$
- **C = a + bi has Re(C) = a "the real part", Im(C) = b "the imaginary part".**
  - The conjugate of C = a + bi is $\overline{C}$ = a - bi
- **There are 3 main representations of complex numbers: algebraic/triangular/polar form, Transformations between them:**
  - $a + bi = L(\cos(\theta) + i.\sin(\theta)) = L.e^{i\theta}$, where $L = \sqrt{(a^2 + b^2)}$, $\theta = \arctan(b/a)$.
- **Complex plane is the coordinate system where we represent the complex number with the real part on x axis and the imaginary part on y axis.**

Notes about complex matrices:

- The superscript H meaning: $A^H = \overline{A}^T$ with each element in A conjugated.
  - It has the same properties of T superscript, $(AB)^H = B^H A^H$
- A **Hermitian** matrix is the matrix that is equal to its conjugate transpose $A = A^H$
  - The Hermitian matrix is square obviously.
  - If the Hermitian matrix is symmetric, then its elements are in fact real numbers.
- In general, the transpose of a complex matrix/vector A is not A^T but it's A^H.
  - The length of a complex vector $x(c_1, c_2)$ is not $\sqrt{(c_1^2 + c_2^2)}$ but it's $\sqrt{x.x}$
  - The inner product (dot product) of two complex vector x and y of the same dimension is not equal to $x^T y$, but it is $x^H y$.
  - A **Unitary** matrix U is the complex equivalent of the orthonormal matrix Q.
    - It has independent column vectors.
      - An example is the matrix S of eigenvectors.
    - We know that $Q^T Q = I$, and if Q is square (orthogonal), then $Q^{-1} = Q^T$
    - In the same sense, $U^H U = I$, if U is square, then $U^{-1} = U^H$

Discrete Fourier Transform (DFT) (Using MIT lecture notation):

- We have a vector **y** of n components, which we want to approximate using a Fourier series with coefficients from another vector **c** of n components.
- DFT says that F_n **c** = **y**, where F_n is the Fourier matrix. Then **c** = F_n^{-1} **y**

$$Fc = y \qquad \begin{bmatrix} 1 & 1 & 1 & \cdot & 1 \\ 1 & w & w^2 & \cdot & w^{n-1} \\ 1 & w^2 & w^4 & \cdot & w^{2(n-1)} \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ 1 & w^{n-1} & w^{2(n-1)} & \cdot & w^{(n-1)^2} \end{bmatrix} \begin{bmatrix} c_0 \\ c_1 \\ c_2 \\ \cdot \\ c_{n-1} \end{bmatrix} = \begin{bmatrix} y_0 \\ y_1 \\ y_2 \\ \cdot \\ y_{n-1} \end{bmatrix}$$

- Where $w = e^{2\pi i/n}$ is the n-th root of 1.
- Computing the Fourier matrix **inverse** is super easy, you just
  - Change any w to $w^{-1} = \omega = e^{-2\pi i/n}$
  - Multiply the whole matrix by 1/n.

- The transformation matrix for DFT matrix is called $W = F_n^{-1}$ and is given by c = Wy
    - For computational convenience, the factor outside W is 1/n
    - The factor $1/\sqrt{n}$ is used for unitary DFT which preserves energy.

$$W = \frac{1}{\sqrt{N}} \begin{bmatrix} 1 & 1 & 1 & 1 & \cdots & 1 \\ 1 & \omega & \omega^2 & \omega^3 & \cdots & \omega^{N-1} \\ 1 & \omega^2 & \omega^4 & \omega^6 & \cdots & \omega^{2(N-1)} \\ 1 & \omega^3 & \omega^6 & \omega^9 & \cdots & \omega^{3(N-1)} \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \omega^{N-1} & \omega^{2(N-1)} & \omega^{3(N-1)} & \cdots & \omega^{(N-1)(N-1)} \end{bmatrix},$$

Computing DFT is $O(n^2)$, A faster algorithm is required.

Fast Fourier Transform (FFT)

- A way to compute DFT in O(n * log(n)), it simplifies the multiplication Fc = y using the formula:
- $\boldsymbol{F_n = \begin{bmatrix} I_{\frac{n}{2}} & D_{\frac{n}{2}} \\ I_{\frac{n}{2}} & -D_{\frac{n}{2}} \end{bmatrix} \begin{bmatrix} F_{\frac{n}{2}} & 0 \\ 0 & F_{\frac{n}{2}} \end{bmatrix} P_n \rightarrow F_n c = \begin{bmatrix} I_{\frac{n}{2}} & D_{\frac{n}{2}} \\ I_{\frac{n}{2}} & -D_{\frac{n}{2}} \end{bmatrix} \begin{bmatrix} F_{\frac{n}{2}}. c_{even} \\ F_{\frac{n}{2}}. c_{odd} \end{bmatrix}}$
- D_n is the same as F_n but with all elements off the main diagonal = 0
- P_n is the n*n row permutation matrix that satisfy the equality:
- $\boldsymbol{P_n \begin{bmatrix} c_0 \\ c_1 \\ c_2 \\ \dots \\ c_{n-1} \end{bmatrix} = \begin{bmatrix} c_{even} \\ c_{odd} \end{bmatrix}, c_{even} = \begin{bmatrix} c_0 \\ c_2 \\ c_4 \\ \dots \end{bmatrix}, c_{odd} = \begin{bmatrix} c_1 \\ c_3 \\ c_5 \\ \dots \end{bmatrix}}.$
- **MIT notation: c' = $F_{n/2}$ * y_{even}, c'' = $F_{n/2}$ * y_{odd}**
- We can use the same rule to compute **$F_{n/2}$ * c_{even} and $F_{n/2}$ * c_{odd}** and so on till we reach F_2 which is very easy to compute.
- If n is not a power of two, we can complete the power by adding more zeros rows to y, and it'll still be a faster computation.

Inverse Fourier transform:

- **The inverse the transform F_n.c = y is c = 1/n * F_n.y**