Implementation & Overview:-

Mottonen et al. suggested in their paper "Transformation of quantum states using uniformly controlled - rotations". {arxiv.org/pdf/quant-ph/0407010.pdf}

Given a state $|a\rangle$, how can be go to $|b\rangle$.

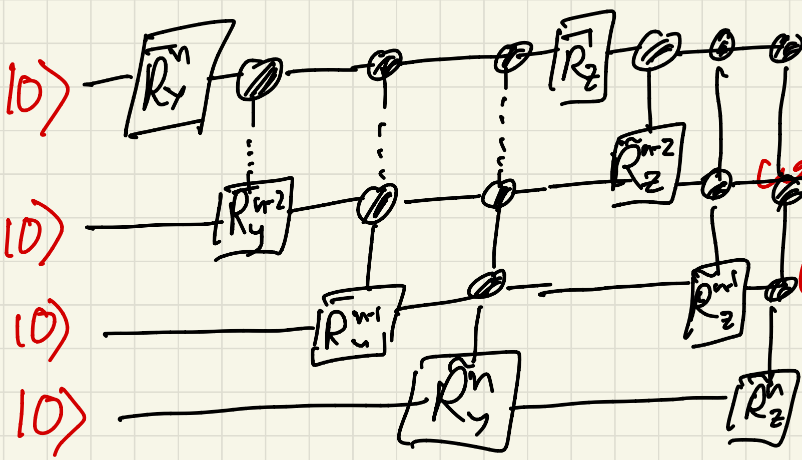
$$|a\rangle \longrightarrow |e\rangle \longrightarrow |b\rangle$$

$$\begin{pmatrix} a_1 \\ a_2 \\ a_3 \\ \vdots \\ a_n \end{pmatrix} \qquad \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \\ \vdots \end{pmatrix} \qquad \begin{pmatrix} b_1 \\ b_2 \\ b_3 \\ \vdots \\ b_n \end{pmatrix}$$

for this task, we require to go from

$$|e\rangle \longrightarrow |b\rangle$$

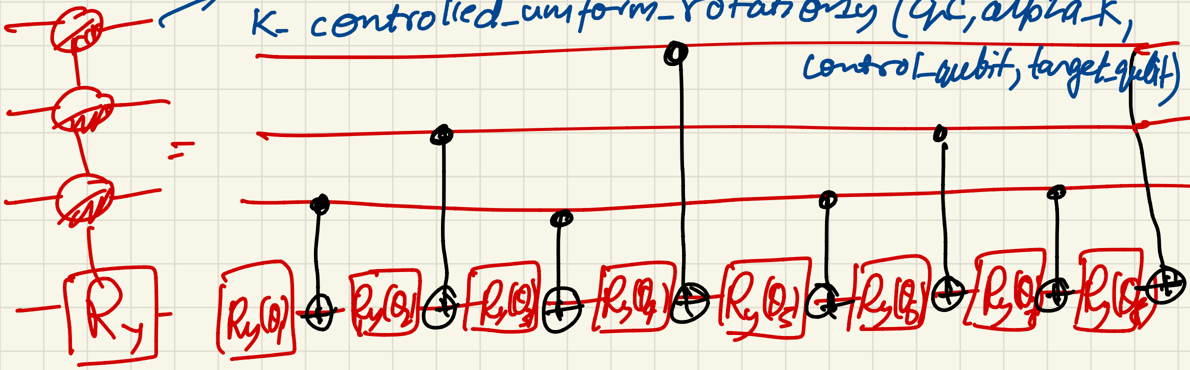
& the required targeted state whose amplitudes would be the required elements of the vector needed to be encoded.



here, R_y^n's, R_z^n's are K controlled rotational gates

= these gates need's to be further decomposed for their implementation

see func
K-controlled_uniform_rotationsy(qc, alpha_k, control_qubit, target_qubit)



we can see how a 3 uniformly controlled y-gate into native R_y & C-NOT gates. Similiarly, we can do this for R_z case.

In nutshell, we apply these K-controlled gates to encode the data point onto the qubit.

Suppose we have a vector,

$$\text{vec} = \begin{pmatrix} z_1 \\ z_2 \\ \vdots \\ z_n \end{pmatrix} \qquad z_i \in \mathbb{C}$$

$$\begin{pmatrix} z_1 \\ z_2 \\ z_3 \\ \vdots \\ z_n \end{pmatrix} = \begin{pmatrix} a_1 e^{i\omega_1} \\ a_2 e^{i\omega_2} \\ a_3 e^{i\omega_3} \\ \vdots \\ a_n e^{i\omega_n} \end{pmatrix}$$

Now, we want to encode this vector on to the qubits. This can done using k-controlled R_y gates which encodes the amplitude part & k-controlled R_z gates which encodes the phase part.

☆ angles required for encoding in k-controlled gates are

$$\alpha_z^{j,k} = \sum_{l=1}^{2^{k-1}} \left(\omega_{(2j-1)2^{k-1}+l} - \omega_{(2j-2)2^k+l} \right) / 2^{k-1}$$

$$\alpha_y^{j,k} = 2\arcsin\left\{ \frac{\sqrt{\sum_{l=1}^{2^{k-1}} |a_{(2j-1)2^{k-1}+l}|^2}}{\sqrt{\sum_{l=1}^{2^k} |a_{(j-1)2^k+l}|^2}} \right\}$$

see func^n
alpha_y(ar,k,j)

But, these rotational angles for k-controlled gates change when we decompose the main k-controlled gates into C-NOT's & R_y's & R_z's gates.

This transformation is given by

see func^n g(i)
m(k)

$$\{\alpha_i\} \longrightarrow \{\theta_i\}$$

See func^n theta(m, alpha)

$$\begin{pmatrix}\theta_1\\ \theta_2\\ \vdots\\ \theta_{2^k}\end{pmatrix} = M \begin{pmatrix}\alpha_1\\ \alpha_2\\ \vdots\\ \alpha_{2^k}\end{pmatrix};\quad \boxed{M_{ij} = 2^{-k}(-1)^{b_{j-1}\cdot g_{i-1}}}$$

Broad overview:

input_vector = $\begin{bmatrix} \\ \\ \end{bmatrix}$

$\downarrow$

extract the α's (α_z, α_y)

$\downarrow$

convert to θ's using $\{\theta_i\} = M\{\alpha_i\}$

$\downarrow$

apply the K-controlled gate in the decomposed form using C-NOT's & $R_y(\theta_i^y)$ & $R_z(\theta_i^z)$

$\downarrow$

After applying a cascade of controlled gates, the input vector gets encoded ~~onto~~ to the qubits, in the form of their amplitudes.

Task - 3

Mottonen - State preparation

the input vector is float values which means, k-uniformly controlled Z gates are not required.

In short, we have to just apply the cascade of k-uniformly controlled y-gates to get the vector encoded.

see funcn

state_prep(input_vector):

###