

ΠΟΛΥΤΕΧΝΕΙΟ ΚΡΗΤΗΣ
ΤΜΗΜΑ
ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ
ΜΗΧΑΝΙΚΩΝ Η/Υ



*ΚΒΑΝΤΙΚΗ ΜΗΧΑΝΙΚΗ ΜΑΘΗΣΗ,
ΕΦΑΡΜΟΓΕΣ ΚΑΙ ΥΛΟΠΟΙΗΣΗ ΣΕ
ΚΒΑΝΤΙΚΟ HARDWARE*

ΣΚΟΡΔΙΑΣ ΘΕΜΙΣΤΟΚΛΗΣ - ΙΩΑΝΝΗΣ

Εξεταστική Επιτροπή

Αναπληρωτής Καθηγητής Δημήτριος Αγγελάκης (Επιβλέπων) (HMMY)

Καθηγητής Δημοσθένης Έλληνας(HMMY)

Αναπληρωτής Καθηγητής Γεώργιος Χαλκιαδάκης(HMMY)

ΠΕΡΙΛΗΨΗ

Η διπλωματική αυτή εργασία ασχολείται με θέματα κλασσικής και κβαντικής μηχανικής μάθησης. Αρχικά, ξεκινάμε παρουσιάζοντας τα βασικά στοιχεία του κβαντικού υπολογισμού, την έννοια του qubit, τις κβαντικές πύλες ενός και δύο qubits, τον κβαντικό εναγκαλισμό, καθώς και πως λειτουργούν μερικοί από τους βασικούς κβαντικούς αλγορίθμους, όπως ο αλγόριθμος του Deutsch. Έπειτα αναλύουμε λεπτομερώς τα μαθηματικά δύο σημαντικών κβαντικών προχωρημένων αλγορίθμων. Του κβαντικού αλγορίθμου του μετασχηματισμού Fourier και αυτού της εκτίμησης φάσης. Στη συνέχεια, παρουσιάζουμε τις κατηγορίες της κλασσικής μηχανικής μάθησης και πιο συγκεκριμένα τον κλασσικό αλγόριθμο της Ανάλυσης Κύριων Συνιστωσών(PCA), ο οποίος χρησιμοποιείται για τη μείωση συνιστωσών σε πολύπλοκα προβλήματα ανάλυσης δεδομένων.

Στο κεντρικό κομμάτι της διπλωματικής αναλύουμε τον κβαντικό αλγόριθμο της Ανάλυσης Κύριων Συνιστωσών(QPCA), παρουσιάζοντας λεπτομερώς τις απαιτούμενες κβαντικές πύλες, τα βήματα και τα κβαντικά κυκλώματα που χρειάστηκαν. Επίσης, συζητάμε τις αναμενόμενες επιταχύνσεις σε σύγκριση με την κλασσική περίπτωση. Ακόμα, στο κεντρικό κομμάτι της διπλωματικής παρουσιάζουμε τις υλοποιήσεις και την σύγκριση και των δύο αλγορίθμων, χρησιμοποιώντας τους διαθέσιμους κβαντικούς υπολογιστές της IBMQ και την κβαντική γλώσσα προγραμματισμού QSkIt, καθώς και δικούς μας προσομοιωτές σε γλώσσα Python. Ένα παράδειγμα με πραγματικά δεδομένα χρησιμοποιείται για να συγκρίνει την επίδοση σε κάθε περίπτωση και για να αναλύσει τις διαφορές και τα πλεονεκτήματα της κβαντικής περίπτωσης σε σύγκριση με την κλασσική.

ABSTRACT

This thesis deals with the interface of classical and quantum machine learning.

We start by introducing the basic principles of quantum computation, the notion of a qubit, single and two qubit gates, entanglement, as well as the workings of some of the basic quantum algorithms such as the Deutsch algorithm. As a next step we discuss in detail the mathematics of the two building blocks of advanced quantum algorithms, the quantum phase estimation and quantum fourier transform. We then proceed by reviewing the classical machine learning methods and more specifically the Principal Component Analysis algorithm used in the reducing the number of features in complex data analytics problems.

In the main part of the thesis, we analyze the quantum Principal Component Analysis algorithm, present in detail the required quantum gates, steps, and circuits involved and also discuss the expected speed ups compared to the classical case. In this main part, we also present implementation and comparison of both algorithms using online prototype available quantum computers by IBM Q using the QSket quantum programming language, as well our own simulators in Python. An example using real data is used to compare the performance in each case, and to illustrate the differences and advantages of the quantum case compared to the classical one.

ΕΥΧΑΡΙΣΤΙΕΣ

Αρχικά, θα ήθελα να εκφράσω θερμές ευχαριστίες στον επιβλέποντα καθηγητή μου κ. Δημήτριο Αγγελάκη, για την εμπιστοσύνη που μου έδειξε κατά την ανάθεση της παρούσας διπλωματικής εργασίας, καθώς και για την άριστη συνεργασία. Επίσης, θα ήθελα να ευχαριστήσω τον υποψήφιο Διδάκτορα, Καλογεράκη Μιχάλη, για την πολύτιμη και ανιδιοτελή βοήθειά του. Ακόμα, θα ήθελα να ευχαριστήσω τους φίλους μου, για την συμπαράσταση και ενθάρρυνση όλους αυτούς τους μήνες, συμβάλλοντας με το δικό τους ξεχωριστό τρόπο στην ολοκλήρωση της διπλωματικής μου εργασίας. Κλείνοντας, ευχαριστώ ειλικρινά τους γονείς μου για την ηθική και αμέριστη στήριξή τους, τη συμπαράστασή και την αγάπη τους.

ΠΕΡΙΕΧΟΜΕΝΑ

1 ΕΙΣΑΓΩΓΗ.....	1
1.1 ΑΠΟ ΤΗΝ ΚΒΑΝΤΙΚΗ ΦΥΣΙΚΗ ΣΤΗΝ ΚΒΑΝΤΙΚΗ ΠΛΗΡΟΦΟΡΙΑ.....	2
2 ΒΑΣΙΚΑ ΣΤΟΙΧΕΙΑ ΚΒΑΝΤΙΚΟΥ ΥΠΟΛΟΓΙΣΜΟΥ.....	6
2.1 ΒΑΣΙΚΕΣ ΑΡΧΕΣ ΚΒΑΝΤΙΚΗΣ ΘΕΩΡΙΑΣ.....	6
2.2 QUBIT ΚΑΙ ΚΒΑΝΤΙΚΗ ΚΑΤΑΣΤΑΣΗ.....	7
2.3 ΚΒΑΝΤΙΚΟΣ ΚΑΤΑΧΩΡΗΤΗΣ.....	8
2.4 ΚΒΑΝΤΙΚΟΣ ΕΝΑΓΚΑΛΙΣΜΟΣ.....	9
2.5 ΚΒΑΝΤΙΚΕΣ ΠΥΛΕΣ.....	10
2.6 ΤΕΛΕΣΤΗΣ ΠΥΚΝΟΤΗΤΑΣ.....	21
2.7 ΚΒΑΝΤΙΚΑ ΚΥΚΛΩΜΑΤΑ, ΥΠΟΛΟΓΙΣΜΟΣ ΚΑΙ ΑΛΓΟΡΙΘΜΟΙ.....	23
2.8 ΚΒΑΝΤΙΚΟΣ ΑΛΓΟΡΙΘΜΟΣ ΜΕΤΑΣΧΗΜΑΤΙΣΜΟΥ FOURIER.....	25
2.9 ΚΒΑΝΤΙΚΟΣ ΑΛΓΟΡΙΘΜΟΣ ΕΚΤΙΜΗΣΗΣ ΦΑΣΗΣ.....	31
3 ΜΗΧΑΝΙΚΗ ΜΑΘΗΣΗ ΚΑΙ ΚΛΑΣΣΙΚΟΣ ΑΛΓΟΡΙΘΜΟΣ ΑΝΑΛΥΣΗΣ ΚΥΡΙΩΝ ΣΥΝΙΣΤΩΣΩΝ.....	35
3.1 ΒΑΣΙΚΑ ΣΤΟΙΧΕΙΑ ΜΗΧΑΝΙΚΗΣ ΜΑΘΗΣΗΣ ΚΑΙ ΚΑΤΗΓΟΡΙΕΣ.....	35
3.2 ΚΛΑΣΣΙΚΟΣ ΑΛΓΟΡΙΘΜΟΣ ΑΝΑΛΥΣΗΣ ΚΥΡΙΩΝ ΣΥΝΙΣΤΩΣΩΝ(PCA).....	38
3.3 ΒΗΜΑΤΑ ΑΛΓΟΡΙΘΜΟΥ PCA.....	40
4 ΚΒΑΝΤΙΚΟΣ ΑΛΓΟΡΙΘΜΟΣ ΑΝΑΛΥΣΗΣ ΚΥΡΙΩΝ ΣΥΝΙΣΤΩΣΩΝ.....	48
4.1 ΒΗΜΑΤΑ ΑΛΓΟΡΙΘΜΟΥ QPCA.....	49
4.2 ΚΒΑΝΤΙΚΟ ΚΥΚΛΩΜΑ QPCA.....	53
5 ΥΛΟΠΟΙΗΣΗ ΑΛΓΟΡΙΘΜΩΝ ΣΕ ΚΛΑΣΣΙΚΟ ΚΑΙ ΚΒΑΝΤΙΚΟ HARDWARE.....	58
5.1 ΥΛΟΠΟΙΗΣΗ ΑΛΓΟΡΙΘΜΟΥ PCA ΣΕ ΠΡΑΓΜΑΤΙΚΑ ΠΡΟΒΛΗΜΑΤΑ.....	59
5.2 ΥΛΟΠΟΙΗΣΗ ΑΛΓΟΡΙΘΜΟΥ QPCA ΣΤΟΝ ΠΡΟΤΥΠΟ ΚΒΑΝΤΙΚΟ ΥΠΟΛΟΓΙΣΤΗ ΤΗΣ IBM.....	62
5.3 ΥΛΟΠΟΙΗΣΗ ΑΛΓΟΡΙΘΜΟΥ QPCA ΣΕ PYTHON SIMULATOR.....	65
5.4 ΣΥΓΚΡΙΣΗ ΚΛΑΣΣΙΚΟΥ(PCA) ΚΑΙ ΚΒΑΝΤΙΚΟΥ(QPCA) ΑΛΓΟΡΙΘΜΟΥ ΚΑΙ ΚΑΤ’ΕΠΕΚΤΑΣΗ ΚΛΑΣΣΙΚΟΥ ΚΑΙ ΚΒΑΝΤΙΚΟΥ ΥΠΟΛΟΓΙΣΤΗ.....	68
6 ΣΥΜΠΕΡΑΣΜΑΤΑ.....	74
6.1 ΜΕΛΛΟΝΤΙΚΗ ΔΟΥΛΕΙΑ.....	75
7 ΒΙΒΛΙΟΓΡΑΦΙΑ.....	77
8 ΠΑΡΑΡΤΗΜΑΤΑ.....	80

1 ΕΙΣΑΓΩΓΗ

Οι κβαντικοί υπολογιστές μπορούν να επεξεργάζονται δεδομένα και να εκτελούν υπολογισμούς, χρησιμοποιώντας τις ιδιότητες της κβαντομηχανικής, όπως για παράδειγμα την αρχή της υπέρθεσης και του εναγκαλισμού καταστάσεων.

Η επιστήμη, που εξετάζει την λειτουργία των κβαντικών υπολογιστών και την ανάπτυξη των κβαντικών αλγορίθμων, ονομάζεται κβαντικός υπολογισμός. Σε αντίθεση, με τους κλασσικούς υπολογιστές, που χρησιμοποιούν σαν μονάδα το bit, οι κβαντικοί υπολογιστές χρησιμοποιούν το κβαντικό bit, που ονομάζεται qubit.

Εξαιτίας του κβαντικού παραλληλισμού, που είναι δυνατόν να επιτευχθεί λόγω του φαινομένου της κβαντικής υπέρθεσης, οι κβαντικοί υπολογιστές μπορούν κατ' ένα τρόπο να επεξεργάζονται πολλές καταστάσεις ταυτόχρονα. Αυτή είναι και η ουσιαστική διαφορά από τους κλασσικούς υπολογιστές. Οι κβαντικοί υπολογιστές με n qubits



Εικόνα: Ο πρότυπος κβαντικός υπολογιστής της Google βασισμένος στην τεχνολογία των ψυχρών υπεραγώγιμων κυκλωμάτων

μπορούν θεωρητικά να βρίσκονται σε υπέρθεση έως 2^n καταστάσεων, ενώ οι κλασσικοί μπορούν να βρίσκονται μόνο σε μία κατάσταση.

Ο βασικός λόγος μελέτης και χρήσης κβαντικών υπολογιστών, που ώθησε την επιστημονική κοινότητα προς αυτή την κατεύθυνση, απορρέει από το σκεπτικό του νόμου του Moore. Ο νόμος του Moore λέει ότι ο αριθμός των τρανζίστορ ενός ολοκληρωμένου κυκλώματος διπλασιάζεται κάθε δύο χρόνια. Αυτό δεν μπορεί να συνεχιστεί, γιατί είναι αδύνατο να συμπυκνώσουμε περισσότερο τα ηλεκτρονικά μέρη πάνω στα υποστρώματα πυριτίου. Επομένως, για να συνεχιστεί η αύξηση της υπολογιστικής ισχύος, είναι αναγκαία η χρησιμοποίηση κβαντικών υπολογιστών. Το μέγεθος των τρανζίστορ επάνω στα τσιπ ήδη πλησιάζει το μέγεθος, όπου η κβαντομηχανική είναι ο κύριος ρυθμιστικός παράγοντας.

1.1 Από την Κβαντική Φυσική στην Κβαντική Πληροφορία

Η κβαντομηχανική μπόρεσε να εξηγήσει φαινόμενα και πειράματα, που η Νευτώνια μηχανική δεν μπορούσε να περιγράψει. Αποτελεί μια θεωρία της Φυσικής, η οποία περιγράφει το πως συμπεριφέρεται η ύλη σε μοριακό, ατομικό και υποατομικό επίπεδο και η οποία έχει επαληθευτεί πειραματικά εκατοντάδες φορές. Ο θεμελιωτής της κβαντικής θεωρίας είναι ο Max Planck, ο οποίος το 1900 μελετώντας την ακτινοβολία του μέλανος σώματος ανακάλυψε την κβάντωση του φωτός, δηλαδή ότι το φως εκπέμπεται μόνο σε συγκεκριμένα ποσά ενέργειας (κβάντα) ανάλογα με τη συχνότητά του ($E = h\nu$, h = σταθερά του Planck). Το 1905 ο Einstein γενικεύει την ιδέα του Planck, με σκοπό να εξηγήσει το φωτοηλεκτρικό φαινόμενο.

Η κβαντική θεωρία έως το 1925 ήταν ένα σύνολο από υποθέσεις, εμπειρικούς κανόνες, θεωρήματα και όχι μια συνεκτική θεωρία. Αυτό ήρθαν να το αλλάξουν οι Heisenberg και Schrodinger. Τότε ο Heisenberg αναπτύσσει τη θεωρία των πινάκων και το 1926 ο Schrodinger προτείνει την περίφημη εξίσωση Schrodinger (κυματοσυναρτήσεις) που περιγράφει τα κβαντικά συστήματα. Την ίδια περίοδο ο Dirac προτείνει την μορφή του κβαντομηχανικού συμβολισμού. Το 1927 ο Neyman βάζει τα θεμέλια της μαθηματικής βάσης της κβαντομηχανικής, με κεντρικό στοιχείο τους γραμμικούς τελεστές που δρουν πάνω σε χώρους Hilbert. Ταυτόχρονα ο Bohr δείχνει τη σχέση της έννοιας της πιθανότητας με τις κυματοσυναρτήσεις του

Schrodinger και υποστηρίζει ότι το τετράγωνο του μέτρου της κυματοσυνάρτησης είναι η πυκνότητα πιθανότητας να βρεθεί το εξεταζόμενο σύστημα στις συντεταγμένες x, y, z, t .

Πολύ αργότερα, το 1980, ο φυσικός Paul Benioff πρότεινε ένα κβαντικό μηχανικό μοντέλο της μηχανής Turing και το 1982 ο Richard Feynman και ο Yuri Manin έδειξαν πώς ένα κβαντικό σύστημα θα μπορούσε να χρησιμοποιηθεί για να πραγματοποιεί υπολογισμούς, που ένας κλασσικός υπολογιστής δεν θα μπορούσε. Το 1985 ο Deutsch επιβεβαίωσε τον Feynman προτείνοντας την πρώτη 'οικουμενική' μηχανή Turing, ανοίγοντας τον δρόμο για ένα μοντέλο κβαντικού κυκλώματος. Φτάνοντας στη δεκαετία του '90 έχουμε τη δημιουργία 3 σημαντικών κβαντικών αλγορίθμων. Το 1992 οι Deutsch και Jozsa ανακάλυψαν τον ομώνυμο κβαντικό αλγόριθμο. Το 1994 ο Shor πρότεινε τον δικό του κβαντικό αλγόριθμο παραγοντοποίησης ακεραίων, που δίνει την δυνατότητα αποκρυπτογράφησης RSA-κρυπτογραφημένων επικοινωνιών και το 1996, ο Grover δημοσίευσε τον κβαντικό αλγόριθμο αναζήτησης σε βάσεις δεδομένων. Στα τέλη της δεκαετίας του 1990, παρά την συνεχόμενη πειραματική πρόοδο, οι περισσότεροι επιστήμονες πίστευαν ότι ο κβαντικός υπολογισμός δεν είναι ανθεκτικός σε σφάλματα και έτσι δεν θα είχε πρακτικές εφαρμογές σε ρεαλιστικά προβλήματα.

Από το 2000 και μετά γίνονται σοβαρές και σημαντικές προσπάθειες στον τομέα των κβαντικών υπολογιστών τόσο στον δημόσιο όσο και στον ιδιωτικό τομέα. Αποκορύφωμα η 23^η Οκτωβρίου 2019, που για πρώτη φορά επιτεύχθηκε η λεγόμενη 'κβαντική υπεροχή'. Μια ομάδα επιστημόνων της Google AI σε συνεργασία με τη NASA, ισχυρίστηκε ότι πραγματοποίησε έναν κβαντικό υπολογισμό που ήταν ανέφικτος σε οποιονδήποτε κλασσικό υπολογιστή. Συγκεκριμένα, υποστήριξαν ότι ο κβαντικός υπολογιστής τους, έκανε σε 200 δευτερόλεπτα υπολογισμούς, που οι πιο ισχυροί συμβατικοί υπερυπολογιστές θα χρειαζόνταν 10.000 χρόνια για να τους εκτελέσουν. Ο υπολογισμός λύνει ένα μαθηματικό πρόβλημα και δεν έχει πρακτικές εφαρμογές.

Στη συνέχεια θα αναφερθούμε στα μοντέλα του κβαντικού υπολογισμού. Αυτά είναι το κβαντικό μοντέλο κυκλώματος, η κβαντική μηχανή Turing, ο αδιαβατικός κβαντικός υπολογιστής, ο one-way κβαντικός υπολογιστής και διάφορα κβαντικά κυτταρικά αυτόματα. Το πιο διαδεδομένο μοντέλο είναι το κβαντικό κύκλωμα. Τα κβαντικά κυκλώματα βασίζονται στο κβαντικό bit, ή "qubit", το οποίο είναι κάπως ανάλογο με το bit στον κλασσικό υπολογισμό. Ενώ το bit μπορεί να πάρει κάθε φορά την τιμή 0 ή 1, το qubit μπορεί να βρίσκεται ταυτόχρονα και στις δύο καταστάσεις,

δηλαδή μπορεί να βρίσκεται σε υπέρθεση των καταστάσεων 0 και 1. Ωστόσο, όταν μετρούνται τα qubits, το αποτέλεσμα της μέτρησης είναι πάντα 0 ή 1. Οι πιθανότητες αυτών των δύο αποτελεσμάτων εξαρτώνται από την κβαντική κατάσταση, στην οποία τα qubits ήταν αμέσως πριν από τη μέτρηση. Ο υπολογισμός πραγματοποιείται με χειρισμό qubits με κβαντικές λογικές πύλες, οι οποίες είναι κάπως ανάλογες με τις κλασσικές λογικές πύλες.

Οι βασικές προσεγγίσεις για τη φυσική εφαρμογή των κβαντικών υπολογιστών είναι οι αναλογικές και οι ψηφιακές. Οι αναλογικές προσεγγίσεις χωρίζονται περαιτέρω σε κβαντική προσομοίωση και αδιαβατικό κβαντικό υπολογισμό. Οι ψηφιακές προσεγγίσεις χρησιμοποιούν κβαντικές λογικές πύλες για να κάνουν υπολογισμούς. Και οι δύο προσεγγίσεις χρησιμοποιούν κβαντικά bit ή αλλιώς qubits. Υπάρχουν επί του παρόντος ορισμένα σημαντικά εμπόδια στον τρόπο κατασκευής χρήσιμων κβαντικών υπολογιστών. Πρώτον, είναι δύσκολο να διατηρηθούν οι κβαντικές καταστάσεις των qubits, καθώς είναι επιρρεπείς σε κβαντική αποκωδικοποίηση. Δεύτερον, οι κβαντικοί υπολογιστές απαιτούν σημαντική διόρθωση σφαλμάτων, καθώς είναι πολύ πιο επιρρεπείς σε σφάλματα από τους κλασσικούς υπολογιστές.

Αξιωματικά οποιοδήποτε υπολογιστικό πρόβλημα, που μπορεί να λυθεί από έναν κλασσικό υπολογιστή, μπορεί επίσης να λυθεί από έναν κβαντικό υπολογιστή. Οι κβαντικοί υπολογιστές δεν παρέχουν επιπλέον πλεονεκτήματα σε σχέση με τους κλασσικούς υπολογιστές από την άποψη της υπολογιστικής δυνατότητας. Θεωρητικά επιτρέπουν το σχεδιασμό αλγορίθμων για ορισμένα προβλήματα, που έχουν σημαντικά χαμηλότερες χρονικές πολυπλοκότητες από τους γνωστούς κλασσικούς αλγόριθμους. Συγκεκριμένα, οι κβαντικοί υπολογιστές θεωρείται ότι μπορούν να λύσουν γρήγορα ορισμένα προβλήματα, που κανένας κλασσικός υπολογιστής δεν μπορούσε να λύσει σε οποιοδήποτε εφικτό χρονικό διάστημα - ένα επίτευγμα γνωστό ως "κβαντική υπεροχή". Τέλος, η μελέτη της υπολογιστικής πολυπλοκότητας των προβλημάτων σε σχέση με τους κβαντικούς υπολογιστές είναι γνωστή ως θεωρία κβαντικής πολυπλοκότητας.

ΚΕΦΑΛΑΙΟ 1: ΕΙΣΑΓΩΓΗ

2 ΒΑΣΙΚΑ ΣΤΟΙΧΕΙΑ ΚΒΑΝΤΙΚΟΥ ΥΠΟΛΟΓΙΣΜΟΥ

Εδώ θα μελετήσουμε αρχικά τις βασικές αρχές της Κβαντικής Θεωρίας. Στη συνέχεια, θα ασχοληθούμε με το τι είναι το qubit και πως περιγράφεται μια κβαντική κατάσταση. Επίσης, θα αναφερθούμε στον κβαντικό καταχωρητή, στον πολύ σημαντικό κβαντικό εναγκαλισμό, στις κβαντικές πύλες και στον τελεστή πυκνότητας. Τέλος, θα μιλήσουμε για τα κβαντικά κυκλώματα και θα δούμε τρεις κβαντικούς αλγορίθμους, τον κβαντικό αλγόριθμο του Deutsch, τον κβαντικό αλγόριθμο του μετασχηματισμού Fourier (QFT) και τον κβαντικό αλγόριθμο εκτίμησης φάσης (QPE).

2.1 Βασικές Αρχές Κβαντικής Θεωρίας

Οι βασικές αρχές που διέπουν την κβαντική θεωρία είναι $\{[B1],[B2],[B3],[B4],[B5]\}$:

1. Σε ένα κβαντικό σύστημα, η κατάσταση του μπορεί να περιγραφεί από ένα διάνυσμα, το οποίο βρίσκεται μέσα στο διανυσματικό χώρο Hilbert στο σύνολο των μιγαδικών αριθμών \mathbb{C} , στον οποίο ορίζεται το εσωτερικό γινόμενο δύο διανυσμάτων: $\langle \phi | \psi \rangle$, $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$ και $|\phi\rangle = \gamma|0\rangle + \delta|1\rangle$
2. Συσχέτιση ενός γραμμικού μετασχηματισμού (τελεστή) στο χώρο Hilbert με κάθε φυσικό μέγεθος. Ένας γραμμικός τελεστής συμβολίζεται με το σύμβολο \hat{A} και έχει σαν ιδιότητα ότι πρέπει να είναι ερμιτιανός, δηλαδή $\hat{A} = \hat{A}^\dagger$, όπου το \hat{A}^\dagger αποτελεί τον αναστροφосуζυγή του πίνακα \hat{A} . Η παραπάνω ιδιότητα είναι

σημαντική, γιατί οι ερμιτιανοί τελεστές έχουν πραγματικές ιδιοτιμές, που είναι απαραίτητο για ένα φυσικό μέγεθος.

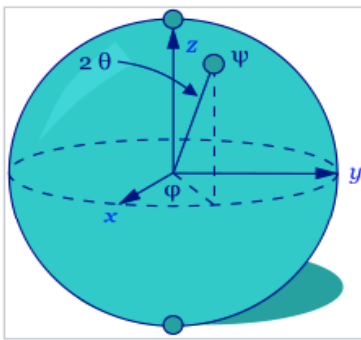
3. Σε ένα κβαντικό σύστημα, που βρίσκεται σε μια κατάσταση $|\psi\rangle$, το αποτέλεσμα μίας μοναδικής μέτρησης ενός φυσικού μεγέθους A μπορεί να είναι μόνο μία από τις ιδιοτιμές του αντίστοιχου γραμμικού τελεστή \hat{A} του φυσικού μεγέθους. Μετά τη μέτρηση το κβαντικό σύστημα μεταβαίνει σε μια καινούργια κατάσταση, η οποία αντιστοιχεί στο ιδιοδιάνυσμα του γραμμικού τελεστή.
4. Το τετράγωνο του μέτρου της προβολής της $|\psi\rangle$ πάνω στο αντίστοιχο ιδιοδιάνυσμα $|\psi_i\rangle$ είναι η πιθανότητα το αποτέλεσμα της μέτρησης ενός κβαντικού συστήματος που βρίσκεται σε μια κατάσταση $|\psi\rangle$, να είναι μία από τις ιδιοτιμές λ_i ενός τελεστή: $P(\lambda_i) = |\langle\psi_i|\psi\rangle|^2$

2.2 Qubit και Κβαντική Κατάσταση

Το κβαντικό bit ή αλλιώς qubit $^{*}([B1],[B4],[B5])$ των κβαντικών υπολογιστών είναι το κβαντικό ανάλογο του bit στους κλασσικούς $^{*}([A1],[A4],[A5])$. Όπως αναφέραμε και παραπάνω το bit μπορεί να πάρει κάθε φορά τις τιμές 0 ή 1, ενώ το qubit μπορεί να βρίσκεται και στις δύο καταστάσεις ταυτόχρονα. Τί είναι το qubit από φυσικής άποψης; Για να το δημιουργήσουμε, χρησιμοποιούμε μόρια, άτομα, ηλεκτρόνια και γενικά μικροσωματίδια με δύο τρόπους. Ο πρώτος είναι με παγίδες ιόντων και ο δεύτερος με υπεραγώγιμα υλικά για δημιουργία συστημάτων με διακριτές ενεργειακές στάθμες, σαν αυτές των ατόμων, δηλαδή τεχνητά άτομα που χρησιμοποιούμε σαν qubits. Το περιγράφουμε χρησιμοποιώντας τον κβαντικό φορμαλισμό Dirac: bra – ket. Το ket αποτελεί ένα διάνυσμα στήλης: $|\psi\rangle = \begin{pmatrix} \alpha \\ \beta \end{pmatrix}$ και το bra αποτελεί ένα διάνυσμα γραμμής: $\langle\varphi| = (\gamma \quad \delta)$. Το εσωτερικό γινόμενο των δύο παραπάνω διανυσμάτων υπολογίζεται ως εξής: $\langle\varphi|\psi\rangle = (\gamma \quad \delta) \begin{pmatrix} \alpha \\ \beta \end{pmatrix} = \gamma\alpha + \delta\beta$. Αν το εσωτερικό γινόμενο είναι μηδέν, τότε τα διανύσματα καταστάσεων είναι ορθογώνια μεταξύ τους. Το $|\psi\rangle^\dagger = \langle\psi| = (\alpha^* \quad \beta^*)$ είναι το αναστροφοσυζυγές διάνυσμα του $|\psi\rangle$. Μια κβαντική κατάσταση πρέπει να είναι κανονικοποιημένη, δηλαδή να ισχύει: $\langle\psi|\psi\rangle = 1$. Το qubit έχει δύο βασικές καταστάσεις την $|0\rangle$ και την $|1\rangle$ και σε αντίθεση με το bit που παίρνει μόνο την τιμή ‘0’ ή μόνο την τιμή ‘1’, μπορεί να γραφτεί σαν υπέρθεση των δύο βασικών καταστάσεων ως εξής: $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$, με $|\alpha|^2 + |\beta|^2 = 1$. Οι δύο βασικές καταστάσεις $|0\rangle$ και $|1\rangle$ είναι ορθοκανονικές μεταξύ τους. Δηλαδή, είναι ορθογώνιες, επομένως το

εσωτερικό τους γινόμενο είναι μηδέν: $\langle 0|1\rangle = \langle 1|0\rangle = 0$ και είναι κανονικοποιημένες: $\langle 0|0\rangle = \langle 1|1\rangle = 1$. Η κβαντική κατάσταση $|\psi\rangle$ είναι ένα διάνυσμα στο χώρο Hilbert, που έχει δύο διαστάσεις. Τα α και β ονομάζονται πλάτη πιθανότητας και είναι μιγαδικοί αριθμοί. Το $|\alpha|^2$ είναι η πιθανότητα η κατάσταση $|\psi\rangle$ να είναι η $|0\rangle$ και το $|\beta|^2$ είναι η πιθανότητα η κατάσταση $|\psi\rangle$ να είναι η $|1\rangle$. Οι δύο βασικές καταστάσεις και η υπέρθεση μπορούν να γραφτούν και σε μορφή πινάκων: $|0\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$, $|1\rangle = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$ και $|\psi\rangle = \alpha \begin{pmatrix} 1 \\ 0 \end{pmatrix} + \beta \begin{pmatrix} 0 \\ 1 \end{pmatrix}$.

Το qubit ακόμα μπορεί να αναπαρασταθεί στην σφαίρα του Bloch ^{*(([B5],[Δ4])}. Η ακτίνα



της σφαίρας του Bloch έχει μέτρο ίσο με τη μονάδα. Η κβαντική κατάσταση $|\psi\rangle$ είναι στην ουσία ένα διάνυσμα με αρχή το κέντρο της σφαίρας και πέρας την επιφάνεια της σφαίρας, που έχει μέτρο ίσο με τη μονάδα. Η $|\psi\rangle$ μπορεί να γραφτεί και σαν:

$$|\psi\rangle = \cos\theta|0\rangle + e^{i\varphi}\sin\theta|1\rangle = \cos\theta|0\rangle + (\cos\varphi +$$

$i\sin\varphi)\sin\theta|1\rangle$, όπου $0 \leq \theta < \frac{\pi}{2}$, $0 \leq \varphi < 2\pi$.

Η γωνία φ ονομάζεται γωνία φάσης και μαζί με τη μεταβλητή θ ορίζουν ένα μοναδικό σημείο επάνω στην επιφάνεια της σφαίρας του Bloch, που αποτελεί το πέρας του διανύσματος κατάστασης. Και οι δύο μεταβλητές φ και θ είναι πραγματικοί αριθμοί.

2.3 Κβαντικός Καταχωρητής

Στους καταχωρητές αποθηκεύονται τιμές μεταβλητών. Ο κβαντικός καταχωρητής ^{*(([B1],[B2])}, ο οποίος αποτελείται από qubits, τα οποία είναι συνήθως διατεταγμένα σε σειρά, αποτελεί το κβαντικό ανάλογο του κλασσικού καταχωρητή που αποτελείται από bits. Το σημαντικό είναι ότι σε έναν κβαντικό καταχωρητή μπορεί να αποθηκευτεί περισσότερη πληροφορία, σε αντίθεση με έναν κλασσικό. Για παράδειγμα, σε έναν κλασσικό καταχωρητή των 2 bit γίνεται να αποθηκευτεί μία τιμή από τις 00 ή 01 ή 10 ή 11, ενώ στον αντίστοιχο κβαντικό καταχωρητή των 2 καταστάσεων γίνεται θεωρητικά να αποθηκευτούν ταυτόχρονα και οι 4 τιμές. Για να κατανοήσουμε καλύτερα, πως λειτουργεί ένας κβαντικός καταχωρητής, πρέπει να εισάγουμε την έννοια του ταυυστικού γινομένου. Έστω ότι έχουμε δύο χώρους Hilbert H_1 και H_2 με

διαστάσεις ο καθένας αντίστοιχα N_1 και N_2 . Ένας μεγαλύτερος χώρος Hilbert H μπορεί να υπολογιστεί σαν το τανυστικό γινόμενο των χώρων H_1 και H_2 ως εξής: $H = H_1 \otimes H_2$ με διαστάσεις $N = N_1 * N_2$. Τώρα, ένας κβαντικός καταχωρητής των 2 καταστάσεων

$$\text{μπορεί να γραφτεί: } |x\rangle = |\psi\rangle \otimes |\phi\rangle = |\psi\phi\rangle = |\psi\rangle = \begin{pmatrix} \alpha \\ \beta \end{pmatrix} \otimes \begin{pmatrix} \gamma \\ \delta \end{pmatrix} = \begin{pmatrix} \alpha\gamma \\ \alpha\delta \\ \beta\gamma \\ \beta\delta \end{pmatrix}.$$

Η γενικευμένη κατάσταση ενός καταχωρητή που αποτελείται από n qubits είναι η εξής:

$$|x_i\rangle = |x_{n-1}\rangle \otimes \dots \otimes |x_1\rangle \otimes |x_0\rangle = |x_{n-1} \dots x_1 x_0\rangle$$

Και σε πιο ανεπτυγμένη μορφή είναι:

$$|x_i\rangle = c_0|0 \dots 00\rangle + c_1|0 \dots 01\rangle + c_2|0 \dots 10\rangle + c_3|0 \dots 11\rangle + \dots + c_{2^{n-1}}|1 \dots 11\rangle = c_0|0\rangle + c_1|1\rangle + c_2|2\rangle + c_3|3\rangle + \dots + c_{2^n-1}|2^n - 1\rangle = \sum_{i=0}^{2^n-1} c_i |i\rangle, \text{ όπου}$$

τα $c_0, c_1, \dots, c_{2^n-1}$ είναι μιγαδικοί αριθμοί και αποτελούν τα πλάτη πιθανότητας των αντίστοιχων βασικών καταστάσεων. Επομένως το διάνυσμα κατάστασης $|x_i\rangle$ βρίσκεται μέσα σε ένα διανυσματικό χώρο, τον λεγόμενο χώρο Hilbert και έχει διαστάσεις 2^n . Επίσης, αποτελείται από 2^n βασικές καταστάσεις, οι οποίες φυσικά είναι όλες ορθογώνιες μεταξύ τους. Το σημαντικότερο όλων, που πρέπει να επισημανθεί εδώ, είναι ότι, όταν ο καταχωρητής τεθεί σε υπέρθεση, έχει τη δυνατότητα να κρατήσει ταυτόχρονα τις 2^n βασικές καταστάσεις. Αυτή η ιδιότητα αποτελεί και τη βάση της κβαντικής παραλληλίας $^{*}([B3],[\Delta3])$.

2.4 Κβαντικός Εναγκαλισμός

Ο κβαντικός εναγκαλισμός $^{*}([B1],[B5],[\Delta4])$ είναι ένα φαινόμενο χωρίς κλασσικό ανάλογο. Ο ορισμός του είναι ότι τα σωματίδια που δημιουργούνται μαζί ή αλληλεπιδρούν συνενώνοντας τις κυματοσυναρτήσεις τους, μπορούν να είναι εναγκαλισμένα, ανεξάρτητα πόσο μακριά βρίσκεται το ένα από το άλλο. Για παράδειγμα, αν το ένα σωματίδιο βρίσκεται στη γη και το άλλο πάει στην άλλη άκρη του σύμπαντος και σε όποιο και από τα δύο συμβεί το οτιδήποτε, το άλλο θα αντιδράσει ακαριαία. Τώρα πρέπει να δούμε πότε μια κβαντική κατάσταση είναι διαχωρίσιμη ή πότε τα σωματίδιά της είναι κβαντικά εναγκαλισμένα. Για να είναι μια κβαντική

$$\text{κατάσταση } |\psi\rangle = \begin{pmatrix} \alpha \\ \beta \\ \gamma \\ \delta \end{pmatrix} \text{ διαχωρίσιμη ή σε κατάσταση γινομένου, πρέπει να ισχύει ότι}$$

$\alpha\delta = \beta\gamma$. Αλλιώς, τα σωματίδια της είναι εναγκαλισμένα. Διάσημες καταστάσεις, που τα σωματίδια τους είναι εναγκαλισμένα, είναι οι λεγόμενες Bell States $^{*}([B1],[B2])$.

Αυτές είναι:

$$|b_{00}\rangle = \frac{|00\rangle + |11\rangle}{\sqrt{2}} = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \end{pmatrix}$$

$$|b_{01}\rangle = \frac{|01\rangle + |10\rangle}{\sqrt{2}} = \frac{1}{\sqrt{2}} \begin{pmatrix} 0 \\ 1 \\ 1 \\ 0 \end{pmatrix}$$

$$|b_{10}\rangle = \frac{|00\rangle - |11\rangle}{\sqrt{2}} = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ 0 \\ 0 \\ -1 \end{pmatrix}$$

$$|b_{11}\rangle = \frac{|01\rangle - |10\rangle}{\sqrt{2}} = \frac{1}{\sqrt{2}} \begin{pmatrix} 0 \\ 1 \\ -1 \\ 0 \end{pmatrix}$$

Μια συμπαγής μορφή γραφής των καταστάσεων Bell, δίδεται από τη σχέση:

$$|b_{xy}\rangle = \frac{|0y\rangle + (-1)^x |1y'\rangle}{\sqrt{2}}$$

2.5 Κβαντικές Πύλες

Οι κβαντικές πύλες $^{*}([B1],[B4],[B5])$ είναι κβαντικοί τελεστές $^{*}([A4])$, που μπορούν να αναπαρασταθούν ως πίνακες. Ένα σημαντικό αξίωμα της κβαντικής θεωρίας είναι ότι υπάρχει τελεστής που αντιστοιχεί σε κάθε μετρήσιμο φυσικό μέγεθος, όπως η θέση, η ταχύτητα, η επιτάχυνση και η ενέργεια. Είναι ουσιώδες σε αυτό το σημείο να δούμε ποιοι είναι οι ερμιτιανοί τελεστές. Ονομάζονται οι τελεστές που έχουν ως αναστροφосуζυγή τον εαυτό τους, δηλαδή ισχύει η σχέση:

$$\hat{A} = \hat{A}^\dagger$$

Επίσης, ένας τελεστής είναι μοναδιαίος, αν ισχύει ότι:

$$\hat{U}\hat{U}^\dagger = \hat{U}^\dagger\hat{U} = I$$

Ακόμα, ένας τελεστής είναι κανονικός αν ισχύει ότι:

$$\hat{A}\hat{A}^\dagger = \hat{A}^\dagger\hat{A}$$

Οι ερμιτιανοί και οι μοναδιαίοι τελεστές είναι κανονικοί. Η δράση ενός τελεστή σε μια κατάσταση, τη μετασχηματίζει σε μια άλλη κατάσταση. Δύο απλοί τελεστές είναι ο ταυτοτικός και ο μηδενικός τελεστής, των οποίων η δράση περιγράφεται από τις σχέσεις:

$$\hat{I}|\psi\rangle = |\psi\rangle$$

$$\hat{0}|\psi\rangle = 0$$

Είδαμε παραπάνω, πως η αναπαράσταση των κβαντικών καταστάσεων γίνεται από διανύσματα στήλης. Αντίστοιχα η αναπαράσταση των τελεστών γίνεται με πίνακες, οι οποίοι μας δίνουν τους κανόνες μετασχηματισμού από μια κβαντική κατάσταση σε μια άλλη.

Οι τελεστές στη κβαντομηχανική είναι γραμμικοί. Η δράση ενός γραμμικού τελεστή σε ένα διάνυσμα κατάστασης, που περιγράφεται από ένα γραμμικό συνδυασμό διανυσμάτων, δίνεται από τη σχέση:

$$\hat{A}\left(\sum_i \lambda_i |\psi_i\rangle\right) = \sum_i \lambda_i (\hat{A}|\psi_i\rangle)$$

Σε αυτό το σημείο, θα δούμε τον ταυτοτικό τελεστή και τρεις βασικούς τελεστές, που αποτελούν τους τελεστές Pauli $\sigma_{(B1),[A4]}$ και αργότερα θα αποτελέσουν και τις βασικές κβαντικές πύλες.

- Η κβαντική πύλη αδράνειας σ_0 ή I :

Ο πίνακας της έχει τη μορφή: $I = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$

Και πραγματοποιεί τους εξής μετασχηματισμούς:

$$I|0\rangle = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \end{pmatrix} = |0\rangle \text{ και } I|1\rangle = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \end{pmatrix} = |1\rangle$$

- Η κβαντική πύλη σ_1 ή σ_x ή X ή αλλιώς NOT:

Ο πίνακας της έχει τη μορφή: $X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$

Και πραγματοποιεί τους εξής μετασχηματισμούς:

$$X|0\rangle = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \end{pmatrix} = |1\rangle \text{ και } X|1\rangle = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \end{pmatrix} = |0\rangle$$

- Η κβαντική πύλη σ_2 ή σ_y ή Y :

Ο πίνακας της έχει τη μορφή: $Y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}$

Και πραγματοποιεί τους εξής μετασχηματισμούς:

$$Y|0\rangle = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} = i \begin{pmatrix} 0 \\ 1 \end{pmatrix} = i|1\rangle \text{ και } Y|1\rangle = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix} \begin{pmatrix} 0 \\ 1 \end{pmatrix} = -i \begin{pmatrix} 1 \\ 0 \end{pmatrix} = -i|0\rangle$$

- Η κβαντική πύλη σ_3 ή σ_z ή Z :

Ο πίνακας της έχει τη μορφή: $Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$

Και πραγματοποιεί τους εξής μετασχηματισμούς:

$$Z|0\rangle = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \end{pmatrix} = |0\rangle \text{ και } Z|1\rangle = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \begin{pmatrix} 0 \\ 1 \end{pmatrix} = -\begin{pmatrix} 0 \\ 1 \end{pmatrix} = -|1\rangle$$

Εν γένει, η απεικόνιση ενός τελεστή A σε μια ορθοκανονική βάση, η οποία αποτελείται από N διανύσματα βάσης $\{ |\psi_i\rangle, i = 1 \dots n \}$, είναι ένας πίνακας, τα στοιχεία του οποίου θα είναι $*([B5])$:

$$\hat{A} = \begin{pmatrix} \langle \psi_1 | \hat{A} | \psi_1 \rangle & \langle \psi_1 | \hat{A} | \psi_2 \rangle & \dots & \langle \psi_1 | \hat{A} | \psi_n \rangle \\ \langle \psi_2 | \hat{A} | \psi_1 \rangle & \langle \psi_2 | \hat{A} | \psi_2 \rangle & \dots & \langle \psi_2 | \hat{A} | \psi_n \rangle \\ \dots & \dots & \dots & \dots \\ \langle \psi_n | \hat{A} | \psi_1 \rangle & \langle \psi_n | \hat{A} | \psi_2 \rangle & \dots & \langle \psi_n | \hat{A} | \psi_n \rangle \end{pmatrix}$$

Θα δούμε τώρα την έννοια του εξωτερικού γινομένου. Το γινόμενο ενός ket $|\psi\rangle$ κι ενός bra $\langle\varphi|$, το οποίο γράφεται ως $|\psi\rangle\langle\varphi|$, αποτελεί το εξωτερικό γινόμενο. Η ποσότητα αυτή, είναι τελεστής με τη μορφή πίνακα. Έχουμε $|\psi\rangle = \begin{pmatrix} \alpha \\ \beta \end{pmatrix}$ και $|\varphi\rangle = \begin{pmatrix} \gamma \\ \delta \end{pmatrix}$. Θα είναι λοιπόν:

$$|\psi\rangle\langle\varphi| = \begin{pmatrix} \alpha \\ \beta \end{pmatrix} (\gamma^* \quad \delta^*) = \begin{pmatrix} \alpha\gamma^* & \alpha\delta^* \\ \beta\gamma^* & \beta\delta^* \end{pmatrix}$$

Με $\langle\varphi| = (\gamma^* \quad \delta^*)$ διάνυσμα γραμμής που αποτελεί το αναστροφosuζυγές του $|\varphi\rangle = \begin{pmatrix} \gamma \\ \delta \end{pmatrix}$ διανύσματος στήλης. Ενώ το αντίστροφο:

$$|\varphi\rangle\langle\psi| = \begin{pmatrix} \gamma \\ \delta \end{pmatrix} (\alpha^* \quad \beta^*) = \begin{pmatrix} \gamma\alpha^* & \gamma\beta^* \\ \delta\alpha^* & \delta\beta^* \end{pmatrix}$$

Αν εφαρμόσουμε το εξωτερικό γινόμενο σε ένα αυθαίρετο ket $|x\rangle$ παίρνουμε:

$$(|\psi\rangle\langle\varphi|)|x\rangle = |\psi\rangle\langle\varphi|x\rangle = \langle\varphi|x\rangle|\psi\rangle$$

Κάτι άλλο που πρέπει να γνωρίζουμε είναι ότι ο ταυτοτικός τελεστής σε μια βάση αποτελούμενη από n - διανύσματα $\{|\psi_i\rangle\}$ μπορεί να γραφτεί ως άθροισμα των εξωτερικών γινομένων. Αυτή η σχέση αποτελεί την σχέση κλειστότητας $^{*([B5])}$ και είναι η εξής:

$$\sum_{i=1}^n |\psi_i\rangle\langle\psi_i| = \hat{I}$$

Μερικές χρήσιμες ιδιότητες από τους τελεστές και την άλγεβρα που συναντάται με τη χρήση τελεστών και καταστάσεων, είναι οι παρακάτω $^{*([B5])}$:

- $\langle\alpha|\hat{A}^\dagger|\beta\rangle = \langle\beta|\hat{A}|\alpha\rangle^*$
- $(\alpha\hat{A})^\dagger = \alpha^*\hat{A}^\dagger$
- $|\psi\rangle^\dagger = \langle\psi|$
- $\langle\psi|^\dagger = |\psi\rangle$
- $(\hat{A}\hat{B})^\dagger = \hat{B}^\dagger\hat{A}^\dagger$
- $(\hat{A}|\psi\rangle)^\dagger = \langle\psi|\hat{A}^\dagger$
- $(\hat{A}\hat{B}|\psi\rangle)^\dagger = \langle\psi|\hat{B}^\dagger\hat{A}^\dagger$
- $(\hat{A} + \hat{B} + \hat{C})^\dagger = \hat{A}^\dagger + \hat{B}^\dagger + \hat{C}^\dagger$

Ήρθε η στιγμή να ανακαλύψουμε ένα από τα πιο σημαντικά θεωρήματα της γραμμικής άλγεβρας, που χρησιμοποιούμε στην κβαντομηχανική, αυτό τη φασματικής ανάλυσης. Το θεώρημα αυτό λέει ότι $^{*([B1],[B2])}$:

‘Κάθε τελεστής μπορεί να γραφτεί σαν άθροισμα των ιδιοτιμών και των ιδιοδιανυσμάτων του’

Δηλαδή, για έναν τελεστή \hat{A} , με ιδιοτιμές α_i , στις οποίες αντιστοιχούν τα ιδιοδιανύσματα $|\psi_i\rangle$, μπορούμε να γράψουμε τον τελεστή στη μορφή $^{*([B5])}$:

$$\hat{A} = \sum_{i=1}^n \alpha_i |\psi_i\rangle\langle\psi_i|$$

Θα ορίσουμε στη συνέχεια τον μεταθέτη δύο τελεστών $^{*([B5])}$ ως εξής: $[A,B] = AB - BA$. Οι μεταθέτες είναι γραμμικοί και επιμερίζονται.

- Αν $[A, B] = 0$, τότε τα μεγέθη λέγονται συμβατά, μπορούν να μετρηθούν ταυτόχρονα και έχουν ίδια ιδιοδιανύσματα.
- Αν $[A, B] \neq 0$, τότε τα μεγέθη λέγονται ασύμβατα, δεν μπορούν να μετρηθούν ταυτόχρονα και δεν έχουν ίδια ιδιοδιανύσματα.

Εάν ισχύει ότι $[A, A^\dagger] = 0$, τότε ο τελεστής ονομάζεται κανονικός.

Θα δούμε τώρα ότι η μέση τιμή ενός φυσικού μεγέθους A που μετριέται από τον τελεστή \hat{A} είναι:

$$\langle \hat{A} \rangle = \langle \psi | \hat{A} | \psi \rangle$$

Ένας τελεστής είναι θετικά ημι-ορισμένος αν ισχύει ότι:

$$\langle \psi | \hat{A} | \psi \rangle \geq 0 \text{ για κάθε } |\psi\rangle \in C^n$$

Δηλαδή αν η μέση τιμή του $\langle \hat{A} \rangle$ είναι μηδενική ή θετική.

Η αβεβαιότητα στη μέτρηση του φυσικού μεγέθους A , δίνεται από τη σχέση:

$$\Delta A = \sqrt{\langle A^2 \rangle - \langle A \rangle^2}$$

Ας περάσουμε τώρα στις συναρτήσεις τελεστών. Μια συνάρτηση ενός τελεστή, μπορεί να γραφτεί ως το ανάπτυγμα Taylor $^{*([B1],[B5])}$ του τελεστή ως εξής:

$$f(A) = \sum_{n=0}^{\infty} a_n A^n$$

Μια άλλη συνάρτηση είναι η εκθετική συνάρτηση $^{*([B1],[B5])}$ και μπορούμε να γράψουμε ότι:

$$e^{kA} = 1 + kA + \frac{k^2}{2!} A^2 + \dots + \frac{k^n}{n!} A^n + \dots$$

Αν ένας τελεστής είναι κανονικός και αναλύεται φασματικά σύμφωνα με το θεώρημα φασματικής ανάλυσης $^{*([B1],[B5])}$, τότε μπορεί να γραφτεί ότι:

$$f(A) = \sum_i f(\alpha_i) |\psi_i\rangle \langle \psi_i|$$

Είδαμε λίγο παραπάνω ότι για να είναι ένας τελεστής μοναδιαίος πρέπει να ισχύει ότι:

$$\hat{U} \hat{U}^\dagger = \hat{U}^\dagger \hat{U} = I$$

Εδώ θα επεκταθούμε λίγο περισσότερο στους μοναδιαίους μετασχηματισμούς. Αν έχω έναν ερμιτιανό τελεστή \hat{H} και έναν αριθμό ε , τότε ο τελεστής $\hat{U} = e^{i\varepsilon \hat{H}}$ είναι

μοναδιαίος. Μέσω του θεωρήματος της φασματικής απεικόνισης ο τελεστής \hat{U} μπορεί επίσης να γραφτεί ως εξής:

$$\hat{U} = \sum_i e^{i\epsilon a_i} |\psi_i\rangle \langle \psi_i|$$

Είναι σημαντικό να σημειωθεί ότι οι μοναδιαίοι μετασχηματισμοί στην πραγματικότητα εκφράζουν τη στροφή των διανυσμάτων στο χώρο. Χρησιμοποιώντας μοναδιαίους μετασχηματισμούς, μπορούμε να μεταφερθούμε από την περιγραφή μέσω διανυσμάτων βάσης $|\psi_i\rangle$, σε ένα άλλο σύνολο διανυσμάτων βάσης $|\varphi_i\rangle$. Ο πίνακας αλλαγής βάσης, θα έχει την εξής μορφή:

$$\hat{U} = \begin{pmatrix} \langle \varphi_1 | \psi_1 \rangle & \langle \varphi_1 | \psi_2 \rangle \\ \langle \varphi_2 | \psi_1 \rangle & \langle \varphi_2 | \psi_2 \rangle \end{pmatrix}$$

Στη συνέχεια θα δούμε τους λεγόμενους προβολικούς τελεστές οι οποίοι ορίζονται ως εξής:

$$\hat{P} = |\psi\rangle \langle \psi|$$

Στην ουσία αποτελούν το εξωτερικό γινόμενο ενός ket με ένα bra μιας κβαντικής κατάστασης $|\psi\rangle$ και είναι ερμιτιανοί. Παρακάτω θα δούμε μερικές ιδιότητες των προβολικών τελεστών:

- Αν η κατάσταση $|\psi\rangle$ είναι κανονικοποιημένη τότε ισχύει: $\hat{P}^2 = \hat{P}$.
- Εάν δύο προβολικοί τελεστές μετατίθενται ($P_1 * P_2 = P_2 * P_1$), τότε το γινόμενό τους είναι επίσης προβολικός τελεστής.
- Εάν έχουμε έναν n-διάστατο χώρο με βάση $\{|1\rangle, |2\rangle, |3\rangle, \dots, |n\rangle\}$ και θέλουμε την προβολή ενός διανύσματος αυτού του χώρου σε έναν χώρο με m διαστάσεις ($m > n$), τότε χρησιμοποιούμε τον τελεστή $\hat{P} = \sum_{i=1}^m |i\rangle \langle i|$.
- Η εξίσωση φασματικής αποσύνθεσης του θεωρήματος φασματικής απεικόνισης μέσω του προβολικού τελεστή μπορεί να γραφτεί ως εξής:

$$\hat{A} = \sum_{i=1}^n \alpha_i \hat{P}_i$$

- **Η πιθανότητα να μετρηθεί ένα σύστημα που περιγράφεται από μια κυματοσυνάρτηση $|\psi\rangle$ σε ένα από τα διανύσματα βάσης, έστω το $|m\rangle$, θα δίδεται από τη σχέση:**

$$Pr_{|m\rangle} = \langle \psi | \hat{P}_m | \psi \rangle = \langle \psi | m \rangle \langle m | \psi \rangle = c_m^2$$

Από τα αξιώματα της Κβαντομηχανικής ^{*(B51)}, που παρουσιάζονται στην ενότητα 2.1, χρειάζεται να γνωρίζουμε, ότι η μέτρηση αλλοιώνει την κατάσταση, η οποία μετριέται. Το αξίωμα της μέτρησης λέει, ότι εάν έχουμε τον τελεστή μέτρησης \hat{P}_n , ο οποίος χρησιμοποιείται για να μετρήσει ένα πιθανό αποτέλεσμα m σε μια κυματοκαταστάση $|\psi\rangle$, η πιθανότητα εύρεσης της ιδιοτιμής m θα είναι:

$$\langle\psi|\hat{P}_m|\psi\rangle$$

Μετά τη μέτρηση, το σύστημα θα βρίσκεται στην κατάσταση:

$$|\psi\rangle \rightarrow \frac{\hat{P}_m|\psi\rangle}{\sqrt{\langle\psi|\hat{P}_m|\psi\rangle}}$$

ΚΒΑΝΤΙΚΕΣ ΠΥΛΕΣ

Ήρθε η στιγμή να μιλήσουμε τώρα για τις κβαντικές πύλες. Οι κβαντικές πύλες χωρίζονται σε 3 κατηγορίες:

- Κβαντικές πύλες που δρουν σε ένα qubit
- Κβαντικές πύλες που δρουν σε δύο qubits
- Κβαντικές πύλες που δρουν σε τρία qubits

Κβαντικές πύλες που δρουν σε ένα qubit

Υπάρχουν οι κβαντικές πύλες που δρουν σε ένα qubit και αποτελούν πίνακες διαστάσεων 2×2 . Τέτοιες πύλες είναι οι παρακάτω:

- Η κβαντική πύλη αδράνειας I
- Η κβαντική πύλη X ή αλλιώς NOT
- Η κβαντική πύλη Y
- Η κβαντική πύλη Z

Οι παραπάνω πύλες παρουσιάστηκαν εκτενώς παραπάνω.

Κβαντικές πύλες για περιστροφή στη σφαίρα του Bloch:

- Περιστροφή γύρω από τον άξονα x : $R_x(\theta) = e^{-\frac{i\theta X}{2}}$
- Περιστροφή γύρω από τον άξονα y : $R_y(\theta) = e^{-\frac{i\theta Y}{2}}$
- Περιστροφή γύρω από τον άξονα z : $R_z(\theta) = e^{-\frac{i\theta Z}{2}}$

Μέσω του θεωρήματος της φασματικής απεικόνισης και με τη βοήθεια του αναπτύγματος Taylor και της εκθετικής συνάρτησης έχουμε:

- $R_x(\theta) = e^{-\frac{i\theta X}{2}} = \cos\left(\frac{\theta}{2}\right)I - i\sin\left(\frac{\theta}{2}\right)X = \begin{pmatrix} \cos\left(\frac{\theta}{2}\right) & -i\sin\left(\frac{\theta}{2}\right) \\ -i\sin\left(\frac{\theta}{2}\right) & \cos\left(\frac{\theta}{2}\right) \end{pmatrix}$
- $R_y(\theta) = e^{-\frac{i\theta Y}{2}} = \cos\left(\frac{\theta}{2}\right)I - i\sin\left(\frac{\theta}{2}\right)Y = \begin{pmatrix} \cos\left(\frac{\theta}{2}\right) & -\sin\left(\frac{\theta}{2}\right) \\ \sin\left(\frac{\theta}{2}\right) & \cos\left(\frac{\theta}{2}\right) \end{pmatrix}$
- $R_z(\theta) = e^{-\frac{i\theta Z}{2}} = \cos\left(\frac{\theta}{2}\right)I - i\sin\left(\frac{\theta}{2}\right)Z = \begin{pmatrix} e^{-i\theta/2} & 0 \\ 0 & e^{i\theta/2} \end{pmatrix}$

- Η κβαντική πύλη Hadamard H:

Ο πίνακάς της έχει τη μορφή: $H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$

Και πραγματοποιεί τους εξής μετασχηματισμούς:

$$H|0\rangle = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ 1 \end{pmatrix} = \frac{|0\rangle + |1\rangle}{\sqrt{2}} \text{ και } H|1\rangle = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \begin{pmatrix} 0 \\ 1 \end{pmatrix} \\ = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ -1 \end{pmatrix} = \frac{|0\rangle - |1\rangle}{\sqrt{2}}$$

- Η κβαντική πύλη φάσης R φ (Το $e^{i\varphi}$ υπολογίζεται από το λεγόμενο τύπο του Euler, δηλαδή από την εξίσωση $e^{i\varphi} = \cos(\varphi) + i\sin(\varphi)$):

Ο πίνακάς της έχει τη μορφή: $R\varphi = \begin{pmatrix} 1 & 0 \\ 0 & e^{i\varphi} \end{pmatrix}$

Και πραγματοποιεί τους εξής μετασχηματισμούς:

$$R\varphi|0\rangle = \begin{pmatrix} 1 & 0 \\ 0 & e^{i\varphi} \end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \end{pmatrix} = |0\rangle \text{ και } R\varphi|1\rangle = \begin{pmatrix} 1 & 0 \\ 0 & e^{i\varphi} \end{pmatrix} \begin{pmatrix} 0 \\ 1 \end{pmatrix} = e^{i\varphi} \begin{pmatrix} 0 \\ 1 \end{pmatrix} \\ = e^{i\varphi}|1\rangle$$

- Η κβαντική πύλη S, που αντιστοιχεί στην R φ με $\varphi = \pi/2$:

Ο πίνακάς της έχει τη μορφή: $S = \begin{pmatrix} 1 & 0 \\ 0 & i \end{pmatrix}$

Και πραγματοποιεί τους εξής μετασχηματισμούς:

$$S|0\rangle = \begin{pmatrix} 1 & 0 \\ 0 & i \end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \end{pmatrix} = |0\rangle \text{ και } S|1\rangle = \begin{pmatrix} 1 & 0 \\ 0 & i \end{pmatrix} \begin{pmatrix} 0 \\ 1 \end{pmatrix} = i \begin{pmatrix} 0 \\ 1 \end{pmatrix} = i|1\rangle$$

- Η κβαντική πύλη T, που αντιστοιχεί στην R φ με $\varphi = \pi/4$:

Ο πίνακάς της έχει τη μορφή: $T = \begin{pmatrix} 1 & 0 \\ 0 & e^{i\pi/4} \end{pmatrix}$

Και πραγματοποιεί τους εξής μετασχηματισμούς:

$$T|0\rangle = \begin{pmatrix} 1 & 0 \\ 0 & e^{i\pi/4} \end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \end{pmatrix} = |0\rangle \text{ και } T|1\rangle = \begin{pmatrix} 1 & 0 \\ 0 & e^{i\pi/4} \end{pmatrix} \begin{pmatrix} 0 \\ 1 \end{pmatrix} = e^{i\pi/4} \begin{pmatrix} 0 \\ 1 \end{pmatrix} = e^{i\pi/4} |1\rangle$$

Φυσικές Πύλες:

Θα δούμε τώρα ορισμένες κβαντικές πύλες μεγέθους 2x2, τις λεγόμενες φυσικές πύλες, που είναι συνεχόμενες παραμετροποιημένες πύλες και ορίζονται ως εξής:

$$U_{1(\lambda)} = \begin{pmatrix} 1 & 0 \\ 0 & e^{i\lambda} \end{pmatrix}$$

$$U_{2(\lambda, \varphi)} = \begin{pmatrix} \frac{1}{\sqrt{2}} & \frac{-e^{i\lambda}}{\sqrt{2}} \\ \frac{e^{i\varphi}}{\sqrt{2}} & \frac{e^{i(\lambda+\varphi)}}{\sqrt{2}} \end{pmatrix}$$

$$U_{3(\lambda, \varphi, \theta)} = \begin{pmatrix} \cos\left(\frac{\theta}{2}\right) & -e^{i\lambda} \sin\left(\frac{\theta}{2}\right) \\ e^{i\varphi} \sin\left(\frac{\theta}{2}\right) & e^{i(\lambda+\varphi)} \cos\left(\frac{\theta}{2}\right) \end{pmatrix}$$

Με παραμέτρους λ (ιδιοτιμή), φ (ιδιοδιάνυσμα), θ (γωνία).

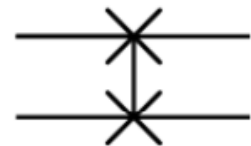
Κβαντικές πύλες που δρουν σε δύο qubits

Ακόμα υπάρχουν κβαντικές πύλες που δρουν σε δύο qubits και αποτελούν πίνακες διαστάσεων 4x4. Τέτοιες πύλες είναι οι παρακάτω:

- Η κβαντική πύλη SWAP, η οποία φαίνεται στη διπλανή εικόνα:

Αυτή η κβαντική πύλη δρα σε 2 qubits και τα ανταλλάσσει.

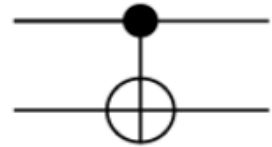
Ο πίνακάς της έχει τη μορφή: $SWAP = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$



Και πραγματοποιεί δηλαδή τον εξής μετασχηματισμό:

$$SWAP|01\rangle = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \end{pmatrix} = |10\rangle$$

- Η κβαντική πύλη CNOT, η οποία φαίνεται στην διπλανή εικόνα:



Αυτή η κβαντική πύλη δρα σε 2 qubits. Το πρώτο είναι το qubit ελέγχου και το δεύτερο το qubit στόχου. Μόνο όταν το qubit ελέγχου βρίσκεται στην κατάσταση $|1\rangle$ η πύλη αλλάζει το qubit στόχου.

Ο πίνακάς της έχει τη μορφή:

$$CNOT = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}$$

Και πραγματοποιεί δηλαδή τους εξής μετασχηματισμούς:

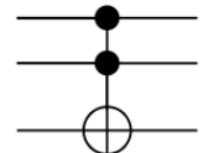
$$CNOT|01\rangle = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \end{pmatrix} = |01\rangle$$

$$\text{και } CNOT|10\rangle = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \end{pmatrix} = |01\rangle$$

Κβαντικές πύλες που δρουν σε τρία qubits

Υπάρχουν, επίσης, και οι κβαντικές πύλες που δρουν σε τρία qubits και αποτελούν πίνακες διαστάσεων 8×8 . Τέτοιες πύλες είναι οι παρακάτω:

- Η κβαντική πύλη Toffoli ή αλλιώς CCNOT, η οποία φαίνεται στη διπλανή εικόνα:



Αυτή η κβαντική πύλη δρα σε 3 qubits. Τα δύο πρώτα είναι τα qubits ελέγχου και το τρίτο είναι το qubit στόχου. Μόνο, όταν τα δύο qubits ελέγχου βρίσκονται στην κατάσταση $|1\rangle$, η πύλη αλλάζει το qubit στόχου.

Ο πίνακάς της έχει τη μορφή:

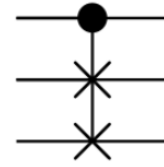
$$Toffoli = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix}$$

Και πραγματοποιεί δηλαδή τους εξής μετασχηματισμούς:

$$Toffoli|001\rangle = |001\rangle \text{ και } Toffoli|110\rangle = |111\rangle$$

- Η κβαντική πύλη Fredkin ή αλλιώς CSWAP, η οποία φαίνεται και στη διπλανή εικόνα:

Αυτή η κβαντική πύλη δρα σε 3 qubits. Το πρώτο είναι το qubit ελέγχου και τα επόμενα δύο είναι τα qubits στόχου. Μόνο, όταν το qubit ελέγχου βρίσκεται στην κατάσταση $|1\rangle$, η πύλη ανταλλάσσει τα qubits στόχου.



Ο πίνακάς της έχει τη μορφή: $CSWAP = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$

Και πραγματοποιεί δηλαδή τους εξής μετασχηματισμούς:

$$CSWAP|001\rangle = |001\rangle \text{ και } CSWAP|110\rangle = |101\rangle$$

Γενίκευση Ελεγχόμενων Κβαντικών Πυλών

Πριν κλείσουμε την ενότητα για τις κβαντικές πύλες, θα αναφερθούμε στην γενική περίπτωση των ελεγχόμενων κβαντικών πυλών. Έστω, λοιπόν, ότι έχουμε μια μοναδιαία πύλη $U = \begin{pmatrix} \alpha & \beta \\ \gamma & \delta \end{pmatrix}$ με μέγεθος πίνακα 2×2 , που δρα σε ένα qubit. Η ελεγχόμενη κβαντική πύλη CU της πύλης U θα έχει μέγεθος πίνακα 4×4 και την εξής μορφή:

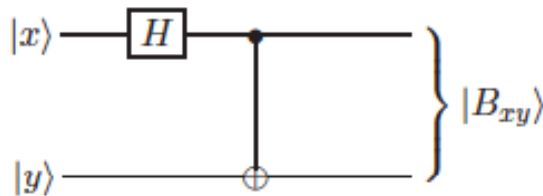
$$CU = \begin{pmatrix} I & 0 \\ 0 & U \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & \alpha & \beta \\ 0 & 0 & \gamma & \delta \end{pmatrix}$$

Η CU θα δρα σε δύο qubits, το qubit ελέγχου και το qubit στόχου. Μόνο, όταν το qubit ελέγχου βρίσκεται στην κατάσταση $|1\rangle$, μόνο και μόνο τότε, θα πραγματοποιείται ο μετασχηματισμός της πύλης U επάνω στο qubit στόχου. Πέρα από την CU, έχουμε και την ελεγχόμενη κβαντική πύλη CCU της πύλης CU, η οποία θα έχει μέγεθος πίνακα 8×8 και την εξής μορφή:

$$CCU = \begin{pmatrix} I & 0 \\ 0 & CU \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \alpha & \beta \\ 0 & 0 & 0 & 0 & 0 & 0 & \gamma & \delta \end{pmatrix}$$

Η CCU θα δρα σε τρία qubits, ένα qubit ελέγχου και δύο qubits στόχου. Μόνο, όταν το qubit ελέγχου βρίσκεται στην κατάσταση $|1\rangle$, μόνο και μόνο τότε, θα πραγματοποιείται ο μετασχηματισμός της πύλης CU επάνω στα qubits στόχου.

Τέλος, εφόσον μάθαμε για τις κβαντικές πύλες, μπορούμε να κατανοήσουμε το απλό κβαντικό κύκλωμα από το οποίο προκύπτουν οι Bell States $*(|B_4\rangle)$, που φαίνεται στη παρακάτω εικόνα. Βλέπουμε ότι αποτελείται από δύο κβαντικές πύλες. Μια Hadamard και μια CNOT.



2.6 Τελεστής Πυκνότητας

Σε αυτή την ενότητα θα παρουσιαστεί ο τελεστής πυκνότητας ρ $*(|B_1\rangle)$. Θα δούμε τον ορισμό και κάποιες ιδιότητες του, ώστε να έχουμε μια ιδέα τι αποτελεί, μιας και θα χρησιμοποιηθεί στον κβαντικό αλγόριθμο QPCA. Ο τελεστής πυκνότητας ρ , λοιπόν, είναι ένας πίνακας που περιγράφει την στατιστική κατάσταση ενός συστήματος στη κβαντική μηχανική. Αποτελεί το κβαντικό – μηχανικό ανάλογο της κατανομής πιθανότητας της θέσης και της ορμής στην κλασσική στατιστική μηχανική.

Έχει τη μορφή: $\hat{\rho} = \begin{pmatrix} \langle 0|\rho|0\rangle & \langle 0|\rho|1\rangle \\ \langle 1|\rho|0\rangle & \langle 1|\rho|1\rangle \end{pmatrix}$. Είναι σημαντικό να σημειωθεί ότι ο ίδιος τελεστής πυκνότητας μπορεί να περιγράφει ένα ή παραπάνω από ένα διανύσματα κατάστασης.

$$\hat{\rho} = \sum_{i=1}^n p_i |\psi_i\rangle \langle \psi_i|, \text{ όπου } \sum_i p_i = 1$$

- Για ένα διάνυσμα κατάστασης $\{|\psi_i\rangle, p_i\} \rightarrow$ ο $\hat{\rho}$ είναι μοναδικός.
- Ένας τελεστής πυκνότητας $\hat{\rho} \rightarrow$ μπορεί να περιγράφει πολλά διανύσματα κατάστασης $\{|\psi_i\rangle, p_i\}$.

Τώρα, θα δούμε κάποιες ιδιότητες του τελεστή πυκνότητας $\hat{\rho}$:

- Είναι ερμιτιανός: $\hat{\rho}^\dagger = \hat{\rho}$
- Έχει ίχνος: $\text{Tr}(\hat{\rho}) = 1$
- $\langle A \rangle = \text{Tr}(\hat{\rho} A)$
- $\hat{\rho}^2 \leq \hat{\rho}$
- Είναι θετικά ημι-ορισμένος και δεν έχει αρνητικές ιδιοτιμές:

$$\langle u | \hat{\rho} | u \rangle \geq 0$$

- Μπορεί να είναι και απείρων διαστάσεων.

Στη συνέχεια θα δούμε πως είναι ο τελεστής πυκνότητας μιας απλής κατάστασης (pure state) και τις ιδιότητές του, καθώς και αντίστοιχα μιας μικτής (mixed state). Μια pure state μπορούμε να τη φανταστούμε, σαν ένα διάνυσμα με αρχή το κέντρο της σφαίρας του Bloch και το πέρας του να φτάνει στην επιφάνειά της. Ενώ από την άλλη, μια mixed state μπορούμε να την φανταστούμε πάλι, σαν ένα διάνυσμα με αρχή το κέντρο της σφαίρας του Bloch, αλλά το πέρας του δεν φτάνει την επιφάνεια της σφαίρας. Μπορούμε και να το σκεφτούμε σαν το διάνυσμα να είναι μέσα στη σφαίρα, δηλαδή να βρίσκεται στο εσωτερικό της σφαίρας του Bloch.

Ο τελεστής πυκνότητας μιας pure state είναι: $\hat{\rho} = |\psi\rangle\langle\psi|$ και έχει τις εξής ιδιότητες:

- $\text{Tr}(\hat{\rho}^2) = 1$
- Τα μη-διαγώνια στοιχεία του πίνακα δεν είναι μηδέν:
$$\hat{\rho}_{mn} \neq 0, \text{ με } m \neq n$$
- $|n| = 1, n \rightarrow \text{διάνυσμα του Bloch}$, αυτό σημαίνει ότι τα διανύσματα κατάστασης φτάνουν στην επιφάνεια της σφαίρας του Bloch

Ο τελεστής πυκνότητας μιας mixed state είναι: $\hat{\rho} = \sum_{i=1}^n p_i |\psi_i\rangle\langle\psi_i|$ και έχει τις παρακάτω ιδιότητες:

- $\text{Tr}(\hat{\rho}^2) < 1$
- Τα μη-διαγώνια στοιχεία του πίνακα είναι μηδέν: $\hat{\rho}_{mn} = 0, \text{ με } m \neq n$
- $|n| < 1, n \rightarrow \text{διάνυσμα του Bloch}$, αυτό σημαίνει ότι τα διανύσματα κατάστασης βρίσκονται εσωτερικά της σφαίρας του Bloch

2.7 Κβαντικά Κυκλώματα, Υπολογισμός και Αλγόριθμοι

Τα κβαντικά κυκλώματα $^{*([B1],[B4],[\Delta4])}$ αποτελούν το μέσο υλοποίησης των κβαντικών αλγορίθμων. Αποτελούνται από κβαντικές πύλες, που δρουν σε qubits. Για να έχουμε καλύτερη οπτικοποίηση, του πως μοιάζει ένα κβαντικό κύκλωμα και πως υλοποιείται και τρέχει, θα παρουσιάσουμε παρακάτω ένα παράδειγμα. Μέσω του παραδείγματος θα κατανοήσουμε καλύτερα την μορφή ενός κβαντικού κυκλώματος, την υλοποίησή του και πως υπολογίζει και βγάζει αποτελέσματα.

Το απλό παράδειγμα, που θα παρουσιάσουμε, περιλαμβάνει τον **κβαντικό αλγόριθμο του Deutsch** $^{*([B4])}$. Ο αλγόριθμος του Deutsch ανακαλύφθηκε το 1984 από τον ίδιο και προοριζόταν για έναν υπολογιστή με δύο μόνο qubits. Το πρόβλημα, που έλυνε ο συγκεκριμένος κβαντικός αλγόριθμος, είναι:

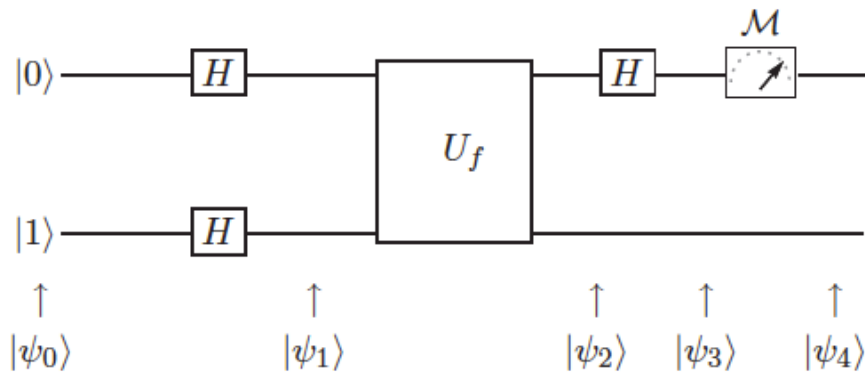
‘Πότε μια συνάρτηση τύπου Boole είναι σταθερή και πότε είναι ισοζυγισμένη.’

Μια συνάρτηση είναι σταθερή όταν: $f(0) = f(1) = \begin{cases} 0 \\ 1 \end{cases}$

Μια συνάρτηση είναι ισοζυγισμένη όταν: $f(0) \neq f(1)$, $\begin{cases} f(0) = 0 \text{ και } f(1) = 1 \\ f(0) = 1 \text{ και } f(1) = 0 \end{cases}$

Με τον κλασσικό τρόπο και μέσω ενός κλασσικού υπολογιστή, το αποτέλεσμα μπορεί να βρεθεί με 2 πράξεις. Υπολογίζοντας τα $f(0)$ και $f(1)$, μπορούμε μετά να αποφανθούμε τι είδους είναι η συνάρτηση. Η ‘μαγεία’ είναι, ότι με το κβαντικό αλγόριθμο του Deutsch, το αποτέλεσμα μπορεί να βρεθεί με 1 πράξη, λόγω του κβαντικού παραλληλισμού.

Το κβαντικό κύκλωμα του αλγορίθμου $^{*([B4])}$ έχει την εξής μορφή:



Προηγουμένως στις κβαντικές πύλες είδαμε πώς λειτουργεί η κβαντική πύλη Hadamard, όταν δρα σε ένα qubit. Πριν προχωρήσουμε στα βήματα και στον υπολογισμό του κβαντικού κυκλώματος, πρέπει να δούμε πώς λειτουργεί η πύλη U_f :

$$U_f |x\rangle \frac{|0\rangle - |1\rangle}{\sqrt{2}} = (-1)^{f(x)} |x\rangle \frac{|0\rangle - |1\rangle}{\sqrt{2}}$$

$$U_f |x, y\rangle = |x, y \oplus f(x)\rangle$$

Βήματα υπολογισμού κβαντικού κυκλώματος του αλγορίθμου Deutsch:

1. Κατάσταση εισόδου: $|\psi_0\rangle = |0\rangle \otimes |1\rangle = |0\rangle |1\rangle = |01\rangle$
2. Δρούμε την πύλη Hadamard σε κάθε ένα από τα 2 qubit:

$$|\psi_1\rangle = (H \otimes H) |\psi_0\rangle = (H \otimes H) |01\rangle = (H|0\rangle)(H|1\rangle) = \left(\frac{|0\rangle + |1\rangle}{\sqrt{2}}\right) \left(\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right)$$

3. Δρούμε την πύλη U_f στα 2 qubit:

$$|\psi_2\rangle = U_f |\psi_1\rangle = \frac{1}{\sqrt{2}} (-1)^{f(0)} |0\rangle \frac{|0\rangle - |1\rangle}{\sqrt{2}} + \frac{1}{\sqrt{2}} (-1)^{f(1)} |1\rangle \frac{|0\rangle - |1\rangle}{\sqrt{2}}$$

- Αν $f(0) = f(1)$ τότε $|\psi_2\rangle = \left(\frac{|0\rangle + |1\rangle}{\sqrt{2}}\right) \left(\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right)$
- Αν $f(0) \neq f(1)$ τότε $|\psi_2\rangle = \left(\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right) \left(\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right)$

4. Δρούμε την πύλη Hadamard μόνο στο πρώτο qubit:

$$|\psi_3\rangle = (H \otimes I) |\psi_2\rangle$$

- Αν η συνάρτηση είναι σταθερή: $f(0) = f(1) = \begin{cases} 0 \\ 1 \end{cases}$,
τότε το αποτέλεσμα είναι: $|\psi_3\rangle = -|0\rangle \left(\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right)$.
- Αν η συνάρτηση είναι ισοζυγισμένη: $f(0) \neq f(1)$, $\begin{cases} f(0) = 0 \text{ και } f(1) = 1 \\ f(0) = 1 \text{ και } f(1) = 0 \end{cases}$,
τότε το αποτέλεσμα είναι: $|\psi_3\rangle = \pm |1\rangle \left(\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right)$.

Επομένως, για να αποφανθούμε αν η συνάρτηση είναι σταθερή ή ισοζυγισμένη, αρκεί να μετρήσουμε το πρώτο qubit της $|\psi_3\rangle$. Αν αυτό είναι $|0\rangle$ τότε είναι σταθερή. Ενώ, αν αυτό είναι $|1\rangle$ τότε είναι ισοζυγισμένη.

2.8 Κβαντικός Αλγόριθμος Μετασχηματισμού Fourier

Σε αυτή την ενότητα θα αναλύσουμε μια από τις πιο σημαντικές υπορουτίνες στον κβαντικό υπολογισμό, τον κβαντικό μετασχηματισμό Fourier ^{*(_{[B1],[B2],[Δ6]})}. Γνωρίζουμε, ότι ο κλασσικός μετασχηματισμός Fourier χρησιμοποιείται για παράδειγμα για την επεξεργασία των σημάτων ή στη θεωρία πολυπλοκότητας και έχει χρονική πολυπλοκότητα $O(n2^n)$. Στον κβαντικό υπολογισμό, ο κβαντικός αλγόριθμος QFT είναι ένας γραμμικός μετασχηματισμός, που δρα πάνω σε qubits και αποτελεί το κβαντικό ανάλογο του κλασσικού αντίστροφου διακριτού μετασχηματισμού Fourier, έχοντας χρονική πολυπλοκότητα $O(N^2)$. Με άλλα λόγια, ο κβαντικός αλγόριθμος QFT είναι η κβαντική υλοποίηση του κλασσικού αντίστροφου διακριτού μετασχηματισμού Fourier πάνω στα πλάτη μιας κυματοσυνάρτησης. Ο κβαντικός αλγόριθμος QFT αποτελεί κομμάτι άλλων κβαντικών αλγορίθμων, όπως για παράδειγμα ο κβαντικός αλγόριθμος εκτίμησης φάσης, που θα μας απασχολήσει παρακάτω.

Ο **κλασσικός διακριτός μετασχηματισμός Fourier(DFT)** ^{*(_[Δ10])} N σημείων μιας ακολουθίας x_j πεπερασμένου μήκους N, που μηδενίζεται έξω από το διάστημα $[0:N-1]$, ορίζεται ως:

$$x_j = \sum_{k=0}^{N-1} y_k \omega_N^{-jk}, j \in [0:N-1]$$

με

$$\omega_N^{jk} = e^{2\pi i \frac{jk}{N}} \text{ και } N = 2^n$$

Για τον υπολογισμό του DFT N σημείων ενός σήματος με χρήση του ορισμού απαιτούνται N^2 μιγαδικοί πολλαπλασιασμοί και $N(N-1)$ μιγαδικές προσθέσεις. Επομένως η πολυπλοκότητα του DFT είναι της τάξης $O(N^2)$.

Ο **κλασσικός αντίστροφος διακριτός μετασχηματισμός Fourier(IDFT)** ^{*(_[Δ10])} δρα σε ένα διάνυσμα x_j με $j = 0 \dots N-1$ και το χαρτογραφεί στο διάνυσμα y_k με $k = 0 \dots N-1$, σύμφωνα με την εξίσωση:

$$y_k = \frac{1}{\sqrt{N}} \sum_{j=0}^{N-1} x_j \omega_N^{jk}, k \in [0:N-1]$$

με

$$\omega_N^{jk} = e^{2\pi i \frac{jk}{N}} \text{ και } N = 2^n$$

Ο **κλασσικός γρήγορος μετασχηματισμός Fourier (FFT)** ^{*([Δ10])} είναι ένας αλγόριθμος που υπολογίζει τον DFT ή τον IDFT μιας ακολουθίας πιο γρήγορα. Γενικά σαν FFT μετασχηματισμούς ονομάζουμε το σύνολο των αλγορίθμων για τον γρήγορο υπολογισμό του DFT. Ο FFT με βάση 2 βασίζεται στην τεχνική “διαίρει και βασίλευε” και η βασική του ιδέα είναι η διάσπαση του σήματος σε δύο σήματα με μισό μήκος $N/2$ το καθένα. Η διάσπαση επαναλαμβάνεται μέχρι να φτάσουμε στον απλό υπολογισμό του DFT δύο σημείων και μπορεί να γίνει τόσο στο πεδίο του χρόνου όσο και στο πεδίο των συχνοτήτων.

Για τον υπολογισμό του διακριτού DFT N σημείων ενός σήματος, με χρήση FFT, απαιτούνται $\frac{N}{2} \log N$ μιγαδικοί πολλαπλασιασμοί και $N \log N$ μιγαδικές προσθέσεις. Επομένως η χρονική πολυπλοκότητα του FFT είναι της τάξης $O(N \log N)$. Η βέλτιστη λειτουργία του FFT επιτυγχάνεται όταν $N = 2^n$, δηλαδή το μήκος του N είναι δύναμη του 2. Άρα ο FFT είναι ιδιαίτερα ταχύς και μάλιστα παρουσιάζει τη μεγαλύτερη δυνατή εξοικονόμηση χρόνου σε σχέση με τον αλγόριθμο, που βασίζεται στην χρήση του ορισμού του DFT. Η χειρότερη περίπτωση για τον FFT, στην οποία δεν υπάρχει εξοικονόμηση χρόνου, είναι όταν το μήκος N είναι πρώτος αριθμός. Στην περίπτωση που το N δεν είναι δύναμη του 2, ούτε πρώτος αριθμός, υπάρχουν FFT αλγόριθμοι που είναι ταχείς. Πρώτον ο FFT με βάση r , με βασική ιδέα τη διάσπαση του σήματος σε r σήματα μήκους N/r και δεύτερον ο FFT πρώτων παραγόντων, με βασική ιδέα τη διάσπαση του σήματος, χρησιμοποιώντας παραγοντοποίηση του N σε γινόμενο πρώτων αριθμών.

Αντίστοιχα, ο **κβαντικός αλγόριθμος του μετασχηματισμού Fourier (QFT)** ^{*([Δ10])} δρα σε μία κβαντική κατάσταση $\sum_{j=0}^{N-1} x_j |j\rangle$ και την χαρτογραφεί στη κβαντική κατάσταση $\sum_{j=0}^{N-1} y_j |j\rangle$. Η χαρτογράφηση έχει ως εξής:

$$QFT_N |x\rangle = \frac{1}{\sqrt{N}} \sum_{y=0}^{N-1} \omega_N^{xy} |y\rangle$$

με

$$\omega_N^{xy} = e^{2\pi i \frac{xy}{N}} \text{ και } N = 2^n$$

Ο κβαντικός αλγόριθμος QFT μπορεί να αποτελέσει μια κβαντική πύλη. Ο πίνακάς του έχει την εξής μορφή:

$$F_N = \frac{1}{\sqrt{N}} \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} \omega_N^{xy} |y\rangle \langle x| = \frac{1}{\sqrt{N}} \begin{pmatrix} 1 & 1 & 1 & 1 & \dots & 1 \\ 1 & \omega & \omega^2 & \omega^3 & \dots & \omega^{N-1} \\ 1 & \omega^2 & \omega^4 & \omega^6 & \dots & \omega^{2(N-1)} \\ 1 & \omega^3 & \omega^6 & \omega^9 & \dots & \omega^{3(N-1)} \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \omega^{(N-1)} & \omega^{2(N-1)} & \omega^{3(N-1)} & \dots & \omega^{(N-1)(N-1)} \end{pmatrix}$$

με $N = 2^n$, $\omega = \omega_N = e^{2\pi i \frac{1}{N}} = \cos\left(\frac{2\pi}{N}\right) + i \sin\left(\frac{2\pi}{N}\right)$ από τύπο Euler. Ο πίνακας F_N είναι μοναδιαίος τελεστής. Κοινώς ισχύει: $F_N^\dagger F_N = F_N F_N^\dagger = I$.

Πριν προχωρήσουμε στα μαθηματικά και στο κβαντικό κύκλωμα του QFT, πρέπει να έχουμε μια αίσθηση του τι κάνει. Ο QFT πραγματοποιεί μετασχηματισμούς μεταξύ της υπολογιστικής βάσης Z και της βάσης Fourier. Όλες οι πολυκβαντικές καταστάσεις στην υπολογιστική βάση Z έχουν αντίστοιχες καταστάσεις στη βάση Fourier. Όπως, για παράδειγμα, η πύλη Hadamard ενός qubit που αποτελεί την μικρότερη κβαντική πύλη QFT, η οποία δρα σε ένα qubit και μετασχηματίζει τις καταστάσεις $|0\rangle$ και $|1\rangle$ της Z βάσης, στις καταστάσεις $|+\rangle = \frac{|0\rangle+|1\rangle}{\sqrt{2}}$ και $|-\rangle = \frac{|0\rangle-|1\rangle}{\sqrt{2}}$ της X -βάσης.

Θα δούμε τώρα πως μετράμε στην Z -βάση. Έστω ότι έχουμε 3 qubits q_1, q_2, q_3 και μετράμε από το 0 έως το 7. Η κβαντική κατάσταση έχει τη μορφή $|q\rangle = |q_3 q_2 q_1\rangle$. Σε δυαδικό σύστημα η κβαντική κατάσταση από το 0 έως το 7 είναι η παρακάτω:

$$|000\rangle, |001\rangle, |010\rangle, |011\rangle, |100\rangle, |101\rangle, |110\rangle, |111\rangle$$

Το qubit q_1 παρατηρούμε ότι αλλάζει ανάμεσα στην κβαντική κατάσταση $|0\rangle$ και την $|1\rangle$ σε κάθε αύξηση της μέτρησης. Το qubit q_2 παρατηρούμε ότι αλλάζει μετά από δύο αυξήσεις της μέτρησης και το qubit q_3 ότι αλλάζει μετά από 4 αυξήσεις. Καταλαβαίνουμε, λοιπόν, ότι αποθηκεύουμε αριθμούς χρησιμοποιώντας τις κβαντικές καταστάσεις $|0\rangle$ και $|1\rangle$.

Στη βάση Fourier είναι διαφορετικά. Αποθηκεύουμε αριθμούς χρησιμοποιώντας περιστροφές γύρω από τον άξονα- Z . Ισχύει ότι : $n \text{ qubits} = \frac{1}{2^n}$ περιστροφές. Για παράδειγμα, για τα 3 qubits της κβαντικής κατάστασης $|q\rangle =$

$|q_3 q_2 q_1\rangle$ έχουμε $\frac{1}{2^3} = \frac{1}{8}$ περιστροφές. Ο αριθμός, που θέλουμε να αποθηκεύσουμε, καθορίζει σε κάθε qubit τη γωνία, που αυτό περιστρέφεται γύρω από τον άξονα-Z.

Στην κβαντική κατάσταση $|000\rangle$ όλα τα qubits είναι στην $|+\rangle = \frac{|0\rangle+|1\rangle}{\sqrt{2}}$.

Καθώς μετράμε από το 0 έως το 7, και για παράδειγμα πάρουμε την μέτρηση 3, που είναι η κβαντική κατάσταση $|q\rangle = |q_3 q_2 q_1\rangle = |011\rangle$, τότε το qubit q_1 θα περιστρέφεται κατά $3/8$, το qubit q_2 κατά το διπλάσιο, δηλαδή $6/8$ και το qubit q_3 κατά τη διπλάσια περιστροφή του qubit q_2 , δηλαδή $12/8$. Πρέπει να τονιστεί ότι το qubit q_1 έχει τη μικρότερη συχνότητα και το qubit q_3 την μεγαλύτερη.

Αφού κατανοήσαμε τα παραπάνω, θα περάσουμε στα μαθηματικά του αλγορίθμου. Θα εφαρμόσουμε τον QFT για $N = 2^n$.

$$QFT_N|x\rangle = \frac{1}{\sqrt{N}} \sum_{y=0}^{N-1} \omega_N^{xy} |y\rangle$$

Γνωρίζουμε ότι $\omega_N^{xy} = e^{2\pi i \frac{xy}{N}}$ και $N = 2^n$. Άρα:

$$QFT_N|x\rangle = \frac{1}{\sqrt{N}} \sum_{y=0}^{N-1} e^{2\pi i \frac{xy}{2^n}} |y\rangle$$

Ξέρουμε ότι $\frac{y}{2^n} = \sum_{k=1}^n \frac{y_k}{2^k}$ και $|y\rangle = |y_1 \dots y_n\rangle$. Επομένως:

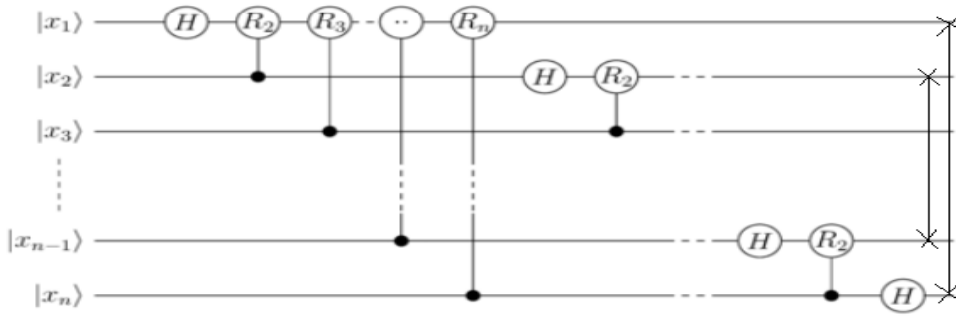
$$QFT_N|x\rangle = \frac{1}{\sqrt{N}} \sum_{y=0}^{N-1} e^{2\pi i (\sum_{k=1}^n \frac{y_k}{2^k})x} |y_1 \dots y_n\rangle$$

Στην συνέχεια το εκθετικό άθροισμα: $e^{2\pi i (\sum_{k=1}^n \frac{y_k}{2^k})x}$, γίνεται εκθετικά γινόμενα:

$\prod_{k=1}^n e^{2\pi i x \frac{y_k}{2^k}}$. Οπότε έχουμε:

$$QFT_N|x\rangle = \frac{1}{\sqrt{N}} \sum_{y=0}^{N-1} \prod_{k=1}^n e^{2\pi i x \frac{y_k}{2^k}} |y_1 \dots y_n\rangle = \frac{1}{\sqrt{N}} \left(|0\rangle + e^{2\pi i \frac{x}{2}} |1\rangle \right) \otimes \left(|0\rangle + e^{2\pi i \frac{x}{2^2}} |1\rangle \right) \otimes \dots \otimes \left(|0\rangle + e^{2\pi i \frac{x}{2^{n-1}}} |1\rangle \right) \otimes \left(|0\rangle + e^{2\pi i \frac{x}{2^n}} |1\rangle \right)$$

Το κβαντικό κύκλωμα του αλγορίθμου απεικονίζεται παρακάτω:



Στο κβαντικό κύκλωμα παρατηρούμε ότι περιέχονται οι παρακάτω πύλες:

- $H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$, $H|0\rangle = \frac{|0\rangle + |1\rangle}{\sqrt{2}}$, $H|1\rangle = \frac{|0\rangle - |1\rangle}{\sqrt{2}}$
- $R_k = \begin{pmatrix} 1 & 0 \\ 0 & e^{\frac{2\pi i}{2^k}} \end{pmatrix}$, $R_k|0\rangle = |0\rangle$, $R_k|1\rangle = e^{\frac{2\pi i}{2^k}}|1\rangle$
- $CR_k = \begin{pmatrix} I & 0 \\ 0 & R_k \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & e^{\frac{2\pi i}{2^k}} \end{pmatrix}$

Οι μετασχηματισμοί τους είναι:

- $H|x_k\rangle = |0\rangle + e^{2\pi i \frac{x_k}{2}}|1\rangle$
- $CR_k|0x_k\rangle = |0x_k\rangle$
- $CR_k|1x_k\rangle = e^{2\pi i \frac{x_k}{2^k}}|1x_k\rangle$

Τώρα θα δούμε βήμα το βήμα το κβαντικό κύκλωμα με κβαντική κατάσταση εισόδου $|x\rangle = |x_1 \dots x_n\rangle$ και $N = 2^n, n - \text{qubit}$.

Στο 1^ο βήμα δρα η κβαντική πύλη Hadamard στο πρώτο qubit $|x_1\rangle$:

$$(H \otimes I \otimes I \dots \otimes I)|x_1 \dots x_n\rangle = \frac{1}{\sqrt{N}} \left(|0\rangle + e^{2\pi i \frac{x_1}{2}} |1\rangle \right) \otimes |x_2 \dots x_n\rangle$$

Στο 2^ο βήμα δρα η R_2 στο πρώτο qubit $|x_1\rangle$, με qubit ελέγχου το $|x_2\rangle$:

$$\frac{1}{\sqrt{N}} \left(|0\rangle + e^{2\pi i \frac{x_2}{2^2} + 2\pi i \frac{x_1}{2}} |1\rangle \right) \otimes |x_2 \dots x_n\rangle = \frac{1}{\sqrt{N}} \left(|0\rangle + e^{2\pi i \frac{x_2}{2^2}} * e^{2\pi i \frac{x_1}{2}} |1\rangle \right) \otimes |x_2 \dots x_n\rangle$$

Στο n^ο βήμα δρα η R_n στο πρώτο qubit $|x_1\rangle$, με qubit ελέγχου το $|x_n\rangle$:

$$\begin{aligned} & \frac{1}{\sqrt{N}} \left(|0\rangle + e^{2\pi i \frac{x_n}{2^n} + 2\pi i \frac{x_{n-1}}{2^{n-1}} + \dots + 2\pi i \frac{x_2}{2^2} + 2\pi i \frac{x_1}{2}} |1\rangle \right) \otimes |x_2 \dots x_n\rangle = \\ & = \frac{1}{\sqrt{N}} \left(|0\rangle + e^{2\pi i \frac{x_n}{2^n}} * e^{2\pi i \frac{x_{n-1}}{2^{n-1}}} * \dots * e^{2\pi i \frac{x_2}{2^2}} * e^{2\pi i \frac{x_1}{2}} |1\rangle \right) \otimes |x_2 \dots x_n\rangle \end{aligned}$$

Σε αυτό το σημείο παρατηρούμε το εξής:

$$x = 2^{n-1}x_1 + 2^{n-2}x_2 + \dots + 2^1x_{n-1} + 2^0x_n$$

Επομένως μπορούμε να γράψουμε:

$$\frac{1}{\sqrt{N}} \left(|0\rangle + e^{2\pi i \frac{x}{2^n}} |1\rangle \right) \otimes |x_2 \dots x_n\rangle$$

Ακολουθούμε την ίδια διαδικασία για καθένα από τα υπόλοιπα qubits ($|x_2\rangle$ έως $|x_n\rangle$) της κβαντικής κατάστασης εισόδου και έχουμε:

$$\frac{1}{\sqrt{N}} \left(|0\rangle + e^{2\pi i \frac{x}{2^n}} |1\rangle \right) \otimes \left(|0\rangle + e^{2\pi i \frac{x}{2^{n-1}}} |1\rangle \right) \otimes \dots \otimes \left(|0\rangle + e^{2\pi i \frac{x}{2^2}} |1\rangle \right) \otimes \left(|0\rangle + e^{2\pi i \frac{x}{2}} |1\rangle \right)$$

Τελικά μετά την δράση των κβαντικών πυλών SWAP έχουμε:

$$\frac{1}{\sqrt{N}} \left(|0\rangle + e^{2\pi i \frac{x}{2}} |1\rangle \right) \otimes \left(|0\rangle + e^{2\pi i \frac{x}{2^2}} |1\rangle \right) \otimes \dots \otimes \left(|0\rangle + e^{2\pi i \frac{x}{2^{n-1}}} |1\rangle \right) \otimes \left(|0\rangle + e^{2\pi i \frac{x}{2^n}} |1\rangle \right)$$

Η υλοποίηση του κυκλώματος του QFT πραγματοποιείται χρησιμοποιώντας $O(n^2)$ βασικές πύλες. Αυτό είναι εκθετικά ταχύτερο ακόμα και από τον γρήγορο FFT, που έχει χρονική πολυπλοκότητα της τάξης $O(N \log N) = O(n2^n)$. Εφόσον έχουμε n qubits στο κύκλωμα του QFT, σε αυτά μπορούν να δράσουν το πολύ n κβαντικές πύλες. Άρα στο συνολικό κύκλωμα χρησιμοποιούνται το πολύ n^2 πύλες. Οι περισσότερες πύλες στο κύκλωμα είναι οι πύλες φάσης R , τις οποίες μπορούμε και να τις παραλείψουμε, όσον αφορά τον υπολογισμό της χρονικής πολυπλοκότητας. Επομένως έχουμε $O(\log n)$ πύλες για κάθε qubit και συνολικές πύλες στο κύκλωμα $O(n \log n)$.

Συγκεντρωτικά οι χρονικές πολυπλοκότητες είναι:

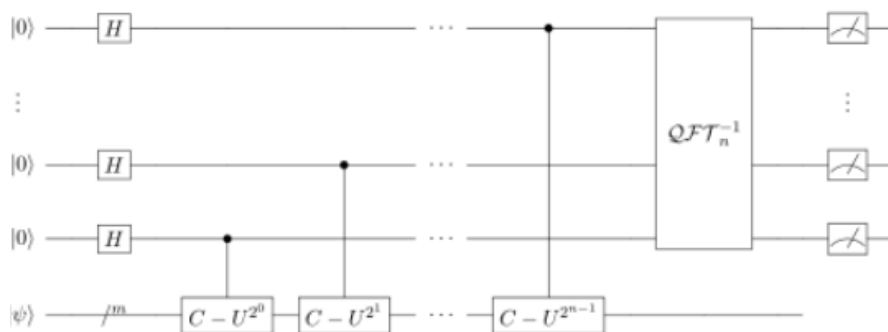
$N = 2^n$			
$n = \log N$	<i>DFT/IDFT</i>	<i>FFT</i>	<i>QFT</i>
	$O(N^2) = O(2^{2n})$	$O(N \log N) = O(n2^n)$	$O(n^2) \rightarrow \text{απλοποίηση } O(n \log n)$

2.9 Κβαντικός Αλγόριθμος Εκτίμησης Φάσης

Μια ακόμα από τις πιο σημαντικές υπορουτίνες στον κβαντικό υπολογισμό είναι ο κβαντικός αλγόριθμος εκτίμησης φάσης (QPE) ^{*([B1],[B2],[Δ6])}, ο οποίος χρησιμοποιεί την υπορουτίνα QFT, που περιγράψαμε στην προηγούμενη ενότητα. Σκοπός του είναι ο εξής: ‘Έστω ότι έχουμε έναν μοναδιαίο τελεστή U . Ο αλγόριθμος υπολογίζει τη φάση θ στην εξίσωση $U|\psi\rangle = e^{2\pi i\theta}|\psi\rangle$, όπου η κβαντική κατάσταση $|\psi\rangle$ αποτελεί ένα ιδιοδιάνυσμα του U και η τιμή $e^{2\pi i\theta}$ την αντίστοιχη ιδιοτιμή του. Επειδή ο πίνακας U είναι μοναδιαίος, όλες οι ιδιοτιμές του έχουν νόρμα ίση με 1’.

Ο QPE χρησιμοποιεί την επαναφορά φάσης(phase kickback), για να γράψει τη φάση του πίνακα U , που είναι γραμμένη στη βάση Fourier, στα t qubits του καταχωρητή μέτρησης. Για να γίνει εφικτό αυτό, πρέπει να βρούμε έναν μετασχηματισμό, που να μας μετασχηματίζει τη βάση Fourier στη βάση υπολογισμού, γιατί μόνο στην τελευταία μπορούμε να πάρουμε μέτρηση. Αυτός ο μετασχηματισμός, προφανώς, είναι δυνατόν να πραγματοποιηθεί από τον αντίστροφο QFT ($QFT^{-1} = QFT^\dagger$). Πιο απλά, όταν χρησιμοποιούμε ένα qubit για να ελέγχουμε την κβαντική πύλη U , λόγω της επαναφοράς φάσης, το qubit θα γυρίσει αναλογικά προς τη φάση $e^{2\pi i\theta}$. Αυτό το γεγονός μας δίνει τη δυνατότητα, να χρησιμοποιήσουμε διαδοχικές ελεγχόμενες πύλες CU , για να επαναλάβουμε αυτήν την περιστροφή τόσες φορές, όσες χρειαζόμαστε μέχρι να κωδικοποιήσουμε τη φάση θ ως αριθμό μεταξύ 0 και 2^t στη βάση Fourier. Τελικά θα χρησιμοποιήσουμε τον αντίστροφο QFT, με σκοπό τον μετασχηματισμό από τη βάση Fourier στην υπολογιστική βάση και έτσι θα μετρήσουμε τη θ .

Το κβαντικό κύκλωμα του αλγορίθμου είναι το παρακάτω:



Στο παραπάνω κύκλωμα παρατηρούμε ότι ο πάνω καταχωρητής ή καταχωρητής μέτρησης περιέχει t qubits και ο κάτω καταχωρητής περιέχει qubits στην κβαντική κατάσταση $|\psi\rangle$.

Στη συνέχεια, θα αναφερθούμε στα βήματα του κβαντικού κυκλώματος του αλγορίθμου και στα μαθηματικά του. Αρχικά, η είσοδος του κυκλώματος είναι: $|\psi_0\rangle = |0\rangle^{\otimes n} |\psi\rangle$, όπου το $|0\rangle^{\otimes n}$ είναι ο καταχωρητής μέτρησης (n qubits) και $|\psi\rangle$ είναι ένα σετ από καταχωρητές qubits. Στο 1^ο βήμα πραγματοποιούμε υπέρθεση στα qubits του καταχωρητή μέτρησης μέσω της πύλης Hadamard ($H^{\otimes n}$) ως εξής:

$|\psi_1\rangle = \frac{1}{\sqrt{2^n}} (|0\rangle + |1\rangle)^{\otimes n} |\psi\rangle$. Στο 2^ο βήμα θα ασχοληθούμε με τις μοναδιαίες ελεγχόμενες πύλες CU, οι οποίες έχουν διαστάσεις πίνακα 4×4 . Γνωρίζουμε πλέον ότι μια τέτοια πύλη δρα σε δύο qubits και πραγματοποιεί τον μετασχηματισμό της στο qubit στόχου, αν και μόνο αν, το qubit ελέγχου βρίσκεται στην κατάσταση $|1\rangle$. Έχουμε, λοιπόν ότι: $U^{2^j} |\psi\rangle = U^{2^{j-1}} U |\psi\rangle$. Γνωρίζουμε ότι: $U |\psi\rangle = e^{2\pi i \theta} |\psi\rangle$. Επομένως έχουμε: $U^{2^j} |\psi\rangle = U^{2^{j-1}} e^{2\pi i \theta} |\psi\rangle$ και τελικά $U^{2^j} |\psi\rangle = e^{2\pi i 2^j \theta} |\psi\rangle$.

Συνεχίζοντας το 2^ο βήμα, αν δράσουμε όλες τις πύλες CU^{2^j} με $j = 0 \dots n-1$ και γνωρίζοντας ότι:

$$|0\rangle \otimes |\psi\rangle + |1\rangle \otimes e^{2\pi i \theta} |\psi\rangle = |0\rangle \otimes |\psi\rangle + e^{2\pi i \theta} |1\rangle \otimes |\psi\rangle = (|0\rangle + e^{2\pi i \theta} |1\rangle) \otimes |\psi\rangle$$

Έχουμε:

$$|\psi_2\rangle = \frac{1}{\sqrt{2^n}} (|0\rangle + e^{2\pi i \theta 2^{n-1}} |1\rangle) \otimes (|0\rangle + e^{2\pi i \theta 2^{n-2}} |1\rangle) \otimes \dots \otimes (|0\rangle + e^{2\pi i \theta 2^2} |1\rangle) \otimes (|0\rangle + e^{2\pi i \theta 2^1} |1\rangle) \otimes (|0\rangle + e^{2\pi i \theta 2^0} |1\rangle) \otimes |\psi\rangle$$

Τελικά η $|\psi_2\rangle$ είναι: $|\psi_2\rangle = \frac{1}{\sqrt{2^n}} \sum_{\varphi=0}^{2^n-1} e^{2\pi i \theta \varphi} |\varphi\rangle \otimes |\psi\rangle$, όπου φ η ακέραια αναπαράσταση των δυαδικών αριθμών n .

Μετέπειτα στο 3^ο βήμα, θα δράσουμε τον αντίστροφο QFT στα qubits του καταχωρητή μέτρησης ώστε, όπως είπαμε παραπάνω, να μετασχηματιστούν από τη βάση Fourier στη βάση υπολογισμού, για να μπορέσουμε να πάρουμε μέτρηση.

Έχουμε, λοιπόν ότι: $QFT_n^\dagger |\psi_2\rangle = |\psi_3\rangle = \frac{1}{2^n} \sum_{x=0}^{2^n-1} \sum_{\varphi=0}^{2^n-1} e^{-2\pi i \frac{(x-2^n\theta)}{2^n} \varphi} |x\rangle \otimes |\psi\rangle$. Στο τελευταίο 4^ο βήμα παίρνουμε τη μέτρηση. Παρατηρούμε ότι η κατάσταση $|\psi_3\rangle$ έχει peak όταν $e^{-2\pi i \frac{(x-2^n\theta)}{2^n} \varphi} = 1$, δηλαδή όταν $x - 2^n\theta = 0 \Leftrightarrow x = 2^n\theta$.

- Αν $x = 2^n \theta$ είναι ακέραιος, τότε η μέτρηση στην υπολογιστική βάση δίνει τη φάση θ στον βοηθητικό καταχωρητή ή καταχωρητή μέτρησης (n-qubits), με υψηλή πιθανότητα. Επομένως η τελική κατάσταση εξόδου είναι: $|\psi_4\rangle = |2^n \theta\rangle \otimes |\psi\rangle$.
- Αν $x = 2^n \theta$ δεν είναι ακέραιος, τότε η κατάσταση $|\psi_3\rangle$ έχει peak κοντά στο $e^{-2\pi i \frac{(x-2^n \theta)}{2^n} \varphi} = 1$, δηλαδή όταν $x - 2^n \theta = 0 \Leftrightarrow x = 2^n \theta$ με πιθανότητα καλύτερη από $\frac{4}{\pi^2} \approx 40\%$.

3 ΜΗΧΑΝΙΚΗ ΜΑΘΗΣΗ ΚΑΙ ΚΛΑΣΣΙΚΟΣ ΑΛΓΟΡΙΘΜΟΣ ΑΝΑΛΥΣΗΣ ΚΥΡΙΩΝ ΣΥΝΙΣΤΩΣΩΝ

Εδώ θα μελετήσουμε τα βασικά στοιχεία της Μηχανικής Μάθησης και τις κατηγορίες της και στη συνέχεια θα ασχοληθούμε με τον κλασσικό αλγόριθμο της Ανάλυσης Κύριων Συνιστωσών(PCA) και τα βήματά του.

3.1 Βασικά Στοιχεία Μηχανικής Μάθησης και Κατηγορίες

Το 1959, ο Arthur Samuel έδωσε τον διάσημο ορισμό του για τη Μηχανική Μάθηση ^{*([Δ8])}, ο οποίος είναι:

‘Η Μηχανική Μάθηση αποτελεί το πεδίο μελέτης, που δίνει στον υπολογιστή την δυνατότητα να μαθαίνει, χωρίς να έχει προγραμματιστεί ρητά.’

Αυτό τον ορισμό, μπορούμε να τον σκεφτούμε και λίγο διαφορετικά, γιατί ένας αλγόριθμος γενικά δεν προσαρμόζεται σε μαθησιακή διαδικασία, αλλά στη συνάρτηση που κωδικοποιεί. Αυτό σημαίνει ότι η σχέση εισόδου-εξόδου σε ένα υπολογιστικό σύστημα συμπληρώνεται από ένα σύνολο δεδομένων, που συνήθως είναι αρκετά μεγάλο. Στη συνέχεια, η μηχανική μάθηση αποτελεί ένα πεδίο της επιστήμης των υπολογιστών, το οποίο αναπτύχθηκε από τη μελέτη δύο τομέων. Αυτόν, της

αναγνώρισης προτύπων και αυτόν της υπολογιστικής θεωρίας μάθησης. Ο τελευταίος αποτελεί κομμάτι της τεχνητής νοημοσύνης. Περαιτέρω, η μηχανική μάθηση έχει να κάνει με την υλοποίηση αλγορίθμων, οι οποίοι έχουν τη δυνατότητα να μαθαίνουν από τα δεδομένα, και μάλιστα λόγω αυτού, να μπορούν να προβλέπουν κιόλας.

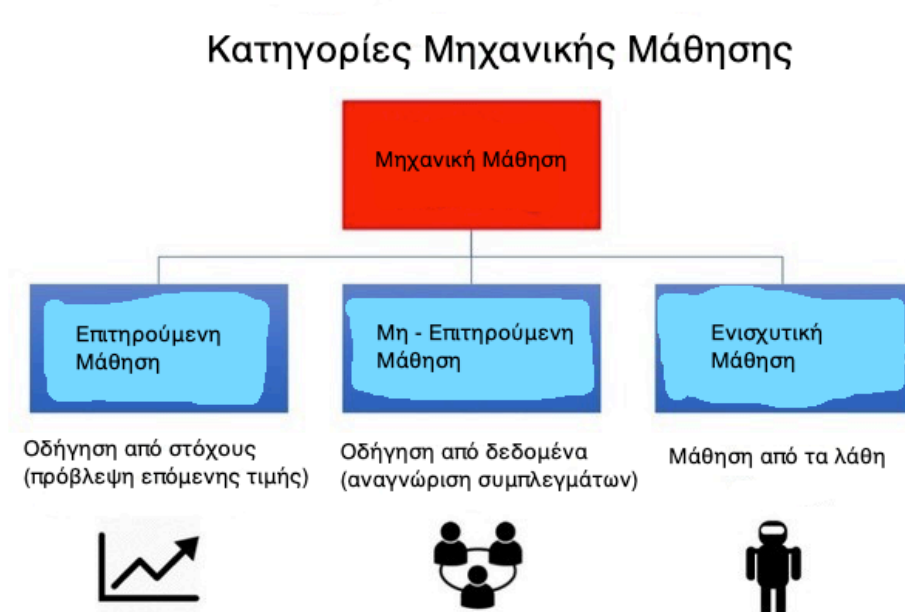
Είσοδος(δεδομένα) \rightarrow Μοντέλο \rightarrow Έξοδος(δεδομένα)

Περισσότερα δεδομένα \rightarrow Καλύτερο Μοντέλο \rightarrow Μεγαλύτερη ακρίβεια

Τέλος, η μηχανική μάθηση αλληλεπιδρά στις ζωές μας καθημερινά και γίνεται ολοένα και πιο ευέλικτη στο να προσαρμόζεται σε συγκεκριμένες ανάγκες μας. Χαρακτηριστικές εφαρμογές που όλοι γνωρίζουμε, αλλά δεν ξέρουμε ότι για αυτές ευθύνεται η μηχανική μάθηση, είναι το Netflix και το Spotify. Άλλες εφαρμογές που μπορεί να έχει η μηχανική μάθηση είναι τα λεγόμενα Self-driving and Self-parking οχήματα.

Σε αυτό το σημείο θα αναφέρουμε τις κατηγορίες της μηχανικής μάθησης και θα τις αναλύσουμε συνοπτικά. Η μηχανική μάθηση χωρίζεται σε 3 κατηγορίες *([Δ8]):

- Στην Επιτηρούμενη Μάθηση (Supervised Learning)
- Στην Μη-Επιτηρούμενη Μάθηση (Unsupervised Learning)
- Στην Ενισχυτική Μάθηση (Reinforcement Learning)



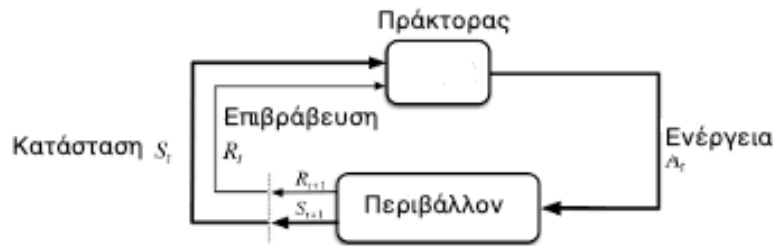
Στην επιτηρούμενη μάθηση $^{*([Δ8])}$, ο υπολογιστής δέχεται παραδείγματα σωστών σχέσεων εισόδου-εξόδου και ο στόχος είναι να ανακαλύψει έναν γενικό κανόνα, ώστε να αντιστοιχίσει τις εισόδους με τα αποτελέσματα. Επομένως, στην επιτηρούμενη μάθηση τα δεδομένα εισόδου-εξόδου παρέχονται και ονομάζονται δεδομένα με ετικέτα. Τέλος, η επιτηρούμενη μάθηση αποτελεί μια μάθηση, που οδηγείται από στόχους (task driven). Η εφαρμογή μια τέτοιας μάθησης για να την κατανοήσουμε καλύτερα είναι:

‘Όταν αναγνωρίζουμε ένα πρόσωπο από διαφορετικές γωνίες και καταστάσεις φωτισμού, ότι ανήκει στο ίδιο άτομο.’

Στην μη επιτηρούμενη μάθηση $^{*([Δ8])}$, ο στόχος είναι να κάνουμε τον υπολογιστή να μάθει να πραγματοποιεί κάτι, χωρίς να του ‘λέμε’ το πώς να το κάνει. Περιγράφει, δηλαδή, την διαδικασία αναζήτησης μοτίβων στα δεδομένα, χωρίς όμως προηγούμενη εμπειρία ή παραδείγματα. Σε αυτήν την κατηγορία μάθησης δεν υπάρχει καθοδήγηση στα δεδομένα. Έχουμε, λοιπόν, τα λεγόμενα δεδομένα χωρίς ετικέτα. Επομένως, κατανοούμε ότι η μη επιτηρούμενη μάθηση οδηγείται από τα δεδομένα (data driven). Η εφαρμογή μια τέτοιας μάθησης για να την κατανοήσουμε καλύτερα είναι:

‘Η έρευνα αγοράς για κάτι.’

Η ενισχυτική μάθηση $^{*([Δ8])}$ είναι ότι πιο κοντινό μπορούμε να συσχετίσουμε με την έκφραση ‘μάθηση’. Η μέθοδος προπόνησης στηρίζεται στην μάθηση μέσω ανατροφοδότησης. Αυτό σημαίνει ότι ο υπολογιστής αλληλεπιδρά με το περιβάλλον του και μαθαίνει από τις λάθος απαντήσεις του. Αν οι απαντήσεις του και η συμπεριφορά του είναι επιθυμητές, τότε επιβραβεύεται. Από την άλλη, αν δεν είναι επιθυμητές, τότε τιμωρείται. Για να κατανοήσουμε καλύτερα, πώς λειτουργεί η ενισχυτική μάθηση, ας πούμε ότι έχουμε σαν δεδομένα ένα πλαίσιο κανόνων και στόχων και έχουμε και έναν πράκτορα, που συνήθως είναι ένα πρόγραμμα υπολογιστή που παίζει ως παίκτης σε ένα παιχνίδι, και ανάλογα με τη στρατηγική που χρησιμοποιεί για να κερδίσει, ανταμείβεται ή τιμωρείται. Κάθε ανταμοιβή ενισχύει την ήδη υπάρχουσα επιλογή στρατηγικής του, ενώ κάθε τιμωρία επιφέρει προσαρμοστικές αλλαγές στη στρατηγική του, με σκοπό την βελτίωσή της. Τέλος, πρέπει να επισημανθεί ότι η ενισχυτική μάθηση αποτελεί έναν κεντρικό μηχανισμό στην ανάπτυξη και μελέτη ευφώνων πρακτόρων. Στην εικόνα της επόμενης σελίδας φαίνεται η συλλογιστική της ενισχυτικής μάθησης:



3.2 Κλασσικός Αλγόριθμος Ανάλυσης Κύριων Συνιστωσών(PCA)

Σε αυτή την ενότητα θα μελετήσουμε έναν αλγόριθμο μη - επιτηρούμενης μηχανικής μάθησης. Ο αλγόριθμος αυτός ονομάζεται: Ανάλυση Κύριων Συνιστωσών (Principal Component Analysis) ^{*([Δ9])}. Είναι ένας αλγόριθμος, ο οποίος ανήκει στους κλασσικούς αλγορίθμους, που τρέχουν σε κλασσικούς υπολογιστές. Ο αλγόριθμος αυτός αποτελεί μία μέθοδο μείωσης διαστάσεων, που συχνά χρησιμοποιείται για τη μείωση των διαστάσεων μεγάλου όγκου δεδομένων, καθώς μετασχηματίζει ένα μεγάλο αριθμό μεταβλητών σε ένα μικρότερο, που όμως περιέχει ακόμα την περισσότερη πληροφορία. Απλούστερα, με άλλα λόγια, μειώνει τον αριθμό των μεταβλητών ενός σετ δεδομένων, καθώς παράλληλα διατηρεί την περισσότερη δυνατή πληροφορία.

Τα χαρακτηριστικά του συγκεκριμένου αλγορίθμου είναι η ομαδοποίηση(clustering) και ότι προβλέπει συνεχείς μεταβλητές. Τίθεται όμως το ερώτημα γιατί χρειαζόμαστε μείωση διαστάσεων δεδομένων και κατ'επέκταση γιατί χρειαζόμαστε το συγκεκριμένο αλγόριθμο. Είναι πολύ απλό βασικά. Διότι πετυχαίνουμε καλύτερη οπτικοποίηση των δεδομένων, έχουμε μείωση θορύβου, μπορούμε να διατηρήσουμε χρήσιμες πληροφορίες σε χαμηλή μνήμη και πετυχαίνουμε μικρότερη χρονική και χωρική πολυπλοκότητα. Η 'μαγεία' και το 'κλειδί' του συγκεκριμένου αλγορίθμου είναι ότι έχουμε τη δυνατότητα να μειώσουμε σε μεγάλο βαθμό τα δεδομένα, κρατώντας όμως τη μέγιστη δυνατή πληροφορία, που αυτά τα δεδομένα μας δίνουν. Κοινώς, δεν χάνουμε πληροφορία με τη μείωση του όγκου των δεδομένων. Και ύστερα, τα μειωμένα αυτά δεδομένα μπορούμε να τα χρησιμοποιήσουμε ως είσοδο σε άλλους αλγορίθμους.

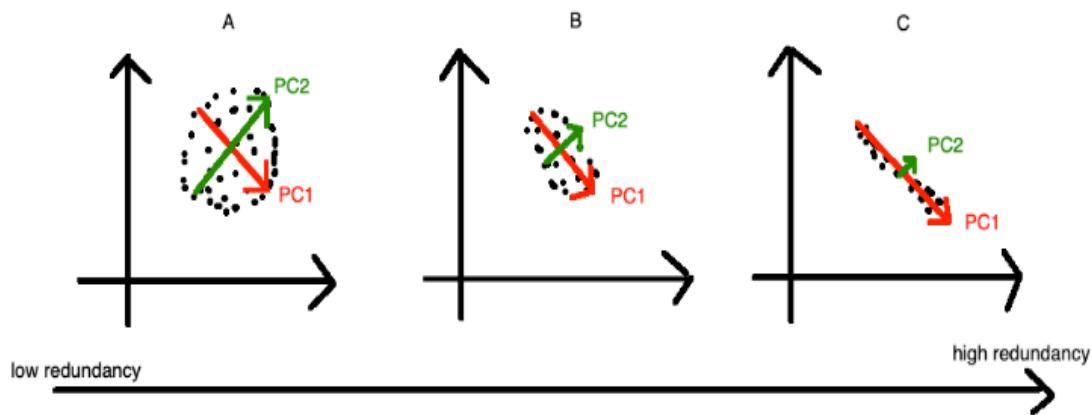
Ο κλασσικός αλγόριθμος μηχανικής μάθησης της Ανάλυσης Κύριων Συνιστωσών(PCA) έχει αρκετές και διάφορες εφαρμογές. Γενικά ο PCA

χρησιμοποιείται για τη μείωση διαστάσεων δεδομένων, για την καλύτερη οπτικοποίηση δεδομένων και για την αναγνώριση μοτίβων. Ειδικότερα έχει και άλλες εφαρμογές

*([Δ9]):

- Στα δεδομένα Έκφρασης Γονιδίων (gene expression data) : Επειδή ο αριθμός των γονιδίων είναι τεράστιος, τα βιολογικά δίκτυα έχουν μεγάλη πολυπλοκότητα. Εδώ χρησιμοποιείται ο PCA για να μειώσει τον όγκο των δεδομένων έκφρασης γονιδίων, χωρίς να χάνεται σημαντικό ποσοστό της πληροφορίας. Αυτό βοηθά τους ερευνητές να πραγματοποιούν τις τεχνικές τους σε μικρότερο όγκο δεδομένων, που περιέχουν την περισσότερη πληροφορία και έτσι να φτάνουν πιο εύκολα σε συμπεράσματα, λόγω χαμηλής πολυπλοκότητας.
- Στην Ποσοτική Χρηματοδότηση (quantitative finance): Η πρώτη εφαρμογή που έχει ο PCA στην ποσοτική χρηματοδότηση είναι στην διαχείριση κινδύνου των χαρτοφυλακίων παραγώγων επιτοκίου. Αυτό γίνεται εφικτό, καθώς μειώνει τη διαπραγμάτευση πολλαπλών πράξεων ανταλλαγής σε 3 ή 4 βασικά στοιχεία, που αντιπροσωπεύουν τη διαδρομή των επιτοκίων σε μακροοικονομική βάση. Δεύτερη εφαρμογή, έχει στον κίνδυνο χαρτοφυλακίου μετοχών, καθώς μειώνει τον κίνδυνο του χαρτοφυλακίου, γιατί εφαρμόζονται στρατηγικές κατανομής στα «κύρια» χαρτοφυλάκια και όχι στα υποκείμενα αποθέματα. Τρίτη εφαρμογή, έχει στην ενίσχυση της απόδοσης των χαρτοφυλακίων, χρησιμοποιώντας τα κύρια στοιχεία για την επιλογή μετοχών με ανοδική πορεία.
- Και στην Νευροεπιστήμη (neuroscience): Η πρώτη εφαρμογή, που έχει ο PCA στη νευροεπιστήμη, είναι ότι χρησιμοποιείται για να διακρίνει την ταυτότητα ενός νευρώνα από το σχήμα του δυναμικού δράσης του. Κατά την ταξινόμηση των δυναμικών δράσης, χρησιμοποιείται πρώτα ο PCA, για να μειώσει τη διάσταση του χώρου των δυναμικών κυματομορφών δράσης και εν συνεχεία εκτελείται η ανάλυση ομαδοποίησης, για να συσχετίσει συγκεκριμένα δυναμικά δράσης με μεμονωμένους νευρώνες. Δεύτερη εφαρμογή, αποτελεί η ανίχνευση συντονισμένων δραστηριοτήτων μεγάλων νευρωνικών συνόλων. Με αυτήν καθορίζεται η σειρά των παραμέτρων κατά τη διάρκεια των μεταβάσεων φάσης στον εγκέφαλο.

Πριν περάσουμε στα βήματα του αλγορίθμου θα δούμε ένα γράφημα για να κατανοήσουμε καλύτερα την έννοια της ‘Κύριας Συνιστώσας(PC)’.



Στο παραπάνω γράφημα βλέπουμε 3 διαγράμματα A, B και C από αριστερά προς δεξιά. Οι κουκκίδες αποτελούν τα δεδομένα και τα 2 διανύσματα τις 2 Κύριες Συνιστώσες. Το κόκκινο διάνυσμα αποτελεί την πρώτη Κύρια Συνιστώσα(PC1) και το πράσινο διάνυσμα την δεύτερη(PC2). Αρχικά πρέπει να επισημανθεί ότι από αριστερά προς τα δεξιά έχουμε αύξηση του πλεονασμού.

- Στο διάγραμμα A: Οι Κύριες Συνιστώσες PC1 και PC2 έχουν και οι δύο σημασία για την πληροφορία. Επομένως έχουμε μικρό πλεονασμό.
- Στο διάγραμμα B: Η Κύρια Συνιστώσα PC1 έχει την ίδια σημασία για την πληροφορία όπως ακριβώς στο διάγραμμα A, αλλά η Κύρια Συνιστώσα PC2 έχει λιγότερη σημασία για την πληροφορία.
- Στο διάγραμμα C: Μόνο η Κύρια Συνιστώσα PC1 έχει την περισσότερη πληροφορία. Επομένως βλέπουμε ότι έχουμε μεγάλο πλεονασμό και αυτό μας δίνει τη δυνατότητα να απορρίψουμε εντελώς την Κύρια Συνιστώσα PC2.

3.3 Βήματα Αλγορίθμου PCA

Ήρθε η ώρα τώρα σε αυτήν την ενότητα να εξερευνήσουμε τα μαθηματικά και τα βήματα ^([19]) του αλγορίθμου της Ανάλυσης Κύριων Συνιστωσών(PCA). Αρχικά έχουμε τα διανύσματα στήλης(μεταβλητές) $x_i = \begin{bmatrix} w_1 \\ \vdots \\ w_n \end{bmatrix}$ με $i = 1, 2, \dots, m$ και στοιχεία w_k , με $k = 1, 2, \dots, n$, το κάθε ένα από αυτά.

**ΚΕΦΑΛΑΙΟ 3: ΜΗΧΑΝΙΚΗ ΜΑΘΗΣΗ ΚΑΙ ΚΛΑΣΣΙΚΟΣ ΑΛΓΟΡΙΘΜΟΣ ΑΝΑΛΥΣΗΣ
ΚΥΡΙΩΝ ΣΥΝΙΣΤΩΣΩΝ**

Στο 1^ο βήμα κεντράρουμε τα δεδομένα γύρω από το 0 και δημιουργούμε τον πίνακα των κεντραρισμένων δεδομένων X . Για να γίνει αυτό πρέπει να βρούμε την μέση τιμή \bar{x}_i για κάθε $i = 1, 2, \dots, m$ των στοιχείων w_k , με $k = 1, 2, \dots, n$, κάθε διανύσματος στήλης(μεταβλητής) x_i με $i = 1, 2, \dots, m$ ως εξής: $\bar{x}_i = \frac{\sum_{k=1}^n w_k}{n}$. Στη συνέχεια, για να βρούμε τα κεντραρισμένα διανύσματα(κεντραρισμένες μεταβλητές) z_i , με $i = 1, 2, \dots, m$, πρέπει να αφαιρέσουμε τη μέση τιμή \bar{x}_i από κάθε ένα στοιχείο w_k , με $k = 1, 2, \dots, n$, κάθε διανύσματος στήλης(μεταβλητής) x_i , με $i = 1, 2, \dots, m$, ως εξής: $z_i = w_k - \bar{x}_i$. Επομένως έχουμε τα κεντραρισμένα διανύσματα στήλης(κεντραρισμένες μεταβλητές) z_i , με $i = 1, 2, \dots, m$ και στοιχεία y_k , όπου $k = 1, 2, \dots, n$ και μορφή $z_i = \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix}$.

Μετά απ'αυτό δημιουργούμε τον πίνακα των κεντραρισμένων δεδομένων X με διαστάσεις $n \times m$, όπου $n \rightarrow$ τα στοιχεία y_k , με $k = 1, 2, \dots, n$ και $m \rightarrow$ τα κεντραρισμένα διανύσματα(κεντραρισμένες μεταβλητές) z_i , με $i = 1, 2, \dots, m$. Άρα ο πίνακας X έχει την εξής μορφή:

$$X = \begin{bmatrix} [z_1] & [z_2] & \dots & [z_m] \end{bmatrix},$$

όπου z_1, z_2, \dots, z_m διανύσματα στήλης.

Στο 2^ο βήμα υπολογίζουμε τον λεγόμενο Πίνακα Συνδιασποράς (Covariance Matrix). Αν έχουμε κεντραρισμένα διανύσματα στήλης(κεντραρισμένες μεταβλητές) z_i , με $i = 1, 2, \dots, m$, ο πίνακας συνδιασποράς θα έχει μέγεθος $m \times m$.

Ο πίνακας συνδιασποράς υπολογίζεται από την εξίσωση:

$$C = \frac{1}{n-1} X^T X$$

όπου X^T ο ανάστροφος πίνακας του X με διαστάσεις $m \times n$ και n το πλήθος των στοιχείων y_k , με $k = 1, 2, \dots, n$, κάθε κεντραρισμένου διανύσματος(κεντραρισμένης μεταβλητής) z_i , με $i = 1, 2, \dots, m$.

Σε μορφή πίνακα ο C είναι:

$$C = \begin{pmatrix} var(z_1)^2 & covar(z_1, z_2)^2 & \dots & covar(z_1, z_m)^2 \\ covar(z_2, z_1)^2 & var(z_2)^2 & \dots & covar(z_2, z_m)^2 \\ covar(z_3, z_1)^2 & covar(z_3, z_2)^2 & \dots & covar(z_3, z_m)^2 \\ \vdots & \vdots & \dots & \vdots \\ covar(z_m, z_1)^2 & covar(z_m, z_2)^2 & \dots & var(z_m)^2 \end{pmatrix}$$

Η $var(z_i)^2$ είναι η διασπορά της μεταβλητής z_i , με $i = 1, 2, \dots, m$ και έχει τύπο:

$$\text{var}(z_i)^2 = \frac{1}{n-1} z_i^T z_i$$

Η $\text{covar}(z_i, z_j)^2 = \text{covar}(z_j, z_i)^2$ είναι η συνδιασπορά των μεταβλητών z_i και z_j .

Έχει τύπο: $\text{covar}(z_i, z_j)^2 = \text{covar}(z_j, z_i)^2 = \frac{1}{n-1} z_i^T z_j$ ή $\frac{1}{n-1} z_j^T z_i$.

- Αν η συνδιασπορά είναι θετική, τότε όταν αυξάνεται η μια μεταβλητή, το ίδιο αυξάνεται και η άλλη.
- Αν η συνδιασπορά είναι αρνητική, τότε όταν αυξάνεται η μια μεταβλητή, η άλλη μειώνεται το ίδιο.
- Αν η συνδιασπορά είναι 0, τότε οι μεταβλητές είναι ορθογώνιες μεταξύ τους, κοινώς δεν είναι συσχετισμένες μεταξύ τους.

Στο 3^ο βήμα υπολογίζουμε τις ιδιοτιμές και τα ιδιοδιανύσματα του πίνακα συνδιασποράς C. Αρχικά βρίσκουμε τις ιδιοτιμές λ από την εξίσωση: $\det|C-\lambda I| = 0$. Και στη συνέχεια, τις χρησιμοποιούμε στην εξίσωση $Cv = \lambda v$, με σκοπό να βρούμε τα ιδιοδιανύσματα v .

Στο 4^ο βήμα τα ιδιοδιανύσματα με τις μεγαλύτερες ιδιοτιμές είναι αυτά που περιέχουν την περισσότερη πληροφορία. Επομένως, μπορούμε να τα κρατήσουμε και να απορρίψουμε τα άλλα. Μετά δημιουργούμε τον πίνακα ιδιοδιανυσμάτων V, ο οποίος έχει διαστάσεις $m \times r$, όπου r (στήλες) είναι οι κύριες συνιστώσες(PC's) που κρατήσαμε, δηλαδή τα ιδιοδιανύσματα με τις μεγαλύτερες ιδιοτιμές, κοινώς αυτά που περιέχουν την περισσότερη πληροφορία.

Στο 5^ο και τελευταίο βήμα βρίσκουμε τα τελικά δεδομένα μειωμένα σε διαστάσεις(όγκος), που όμως περιέχουν την περισσότερη πληροφορία. Έχουμε, λοιπόν: $Y = X * V$ τελικά δεδομένα με διαστάσεις $n \times r$, αφού $X(n \times m) * V(m \times r) = Y(n \times r)$.

Παρακάτω ένα απλό παράδειγμα του αλγορίθμου PCA σε δικό μου κώδικα σε γλώσσα Python για καλύτερη κατανόηση των βημάτων.

Για να έχουμε καλύτερη αίσθηση των βημάτων του αλγορίθμου, θα δούμε ένα παράδειγμα με 2 διανύσματα(μεταβλητές) των 10 στοιχείων το καθένα. Το παράδειγμα υπάρχει υλοποιημένο, με δικό μου κώδικα, στη γλώσσα Python στο ΠΑΡΑΡΤΗΜΑ 1 με ονομασία PCA_Initial_Example_In_presentation.py.

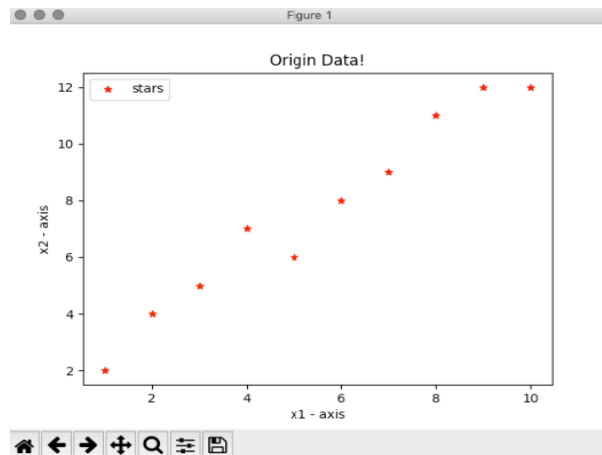
Τα αρχικά δεδομένα(δύο διανύσματα στήλης με μέγεθος 10x1 το καθένα) είναι:

Άξονας x: $x_1 = [1,2,3,4,5,6,7,8,9,10]$ Άξονας y: $x_2 = [2,4,5,7,6,8,9,11,12,12]$

ΚΕΦΑΛΑΙΟ 3: ΜΗΧΑΝΙΚΗ ΜΑΘΗΣΗ ΚΑΙ ΚΛΑΣΣΙΚΟΣ ΑΛΓΟΡΙΘΜΟΣ ΑΝΑΛΥΣΗΣ ΚΥΡΙΩΝ ΣΥΝΙΣΤΩΣΩΝ

```
Origin Data
x1 is : [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
x2 is : [2, 4, 5, 7, 6, 8, 9, 11, 12, 12]
```

Το διάγραμμα των αρχικών δεδομένων είναι:



ΒΗΜΑ 1^ο

Στο πρώτο βήμα βρίσκουμε τη μέση τιμή \bar{x}_1 και \bar{x}_2 των στοιχείων κάθε ενός διανύσματος x_1 και x_2 . Αυτά είναι τα: $\bar{x}_1 = 5,5$ και $\bar{x}_2 = 7,6$.

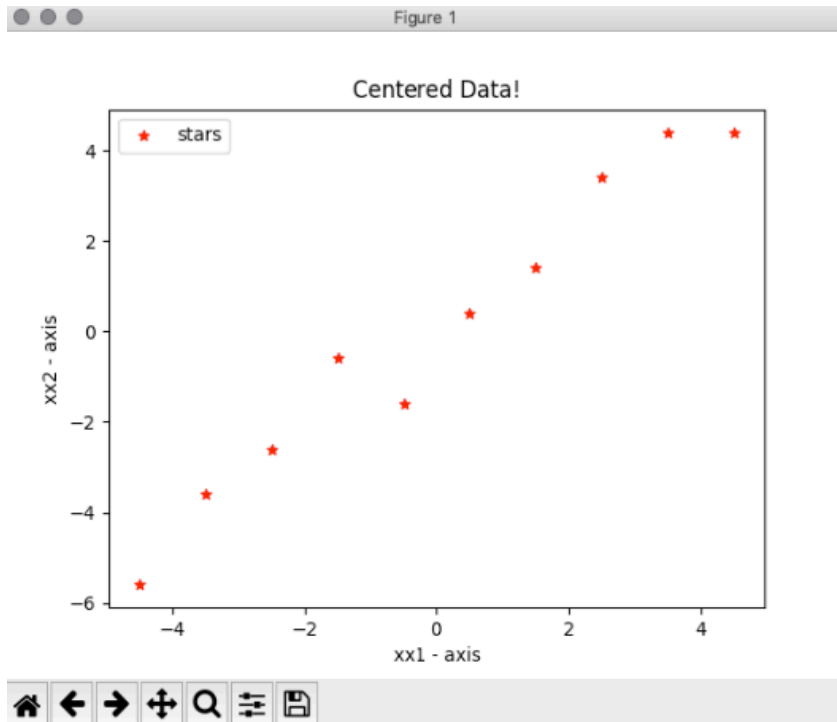
```
meanx1 is : 5.5
meanx2 is : 7.6
```

Από κάθε στοιχείο των διανυσμάτων x_1 και x_2 αφαιρούμε την αντίστοιχη μέση τιμή. Υπολογίζουμε δηλαδή τα $z_1 = x_1 - \bar{x}_1$ και $z_2 = x_2 - \bar{x}_2$ για κάθε στοιχείο των x_1 και x_2 , με σκοπό το κεντράρισμα των δεδομένων.

Τα κεντραρισμένα δεδομένα είναι:

```
Centered Data
xx1 is : [-4.5, -3.5, -2.5, -1.5, -0.5, 0.5, 1.5, 2.5, 3.5, 4.5]
xx2 is : [-5.6, -3.6, -2.6, -0.6, -1.6, 0.4, 1.4, 3.4, 4.4, 4.4]
```

Το διάγραμμα των κεντραρισμένων δεδομένων είναι:



Δημιουργούμε τον πίνακα X που έχει ως 1^η στήλη τα στοιχεία $z_1 = x_1 - \bar{x}_1$ και ως δεύτερη στήλη τα στοιχεία $z_2 = x_2 - \bar{x}_2$. Ο X με διαστάσεις 10×2 είναι:

```
X-matrix:
[[-4.5 -5.6]
 [-3.5 -3.6]
 [-2.5 -2.6]
 [-1.5 -0.6]
 [-0.5 -1.6]
 [ 0.5  0.4]
 [ 1.5  1.4]
 [ 2.5  3.4]
 [ 3.5  4.4]
 [ 4.5  4.4]]
```

ΒΗΜΑ 2^ο

Στο δεύτερο βήμα του αλγορίθμου υπολογίζουμε τον λεγόμενο Πίνακα Συνδιασποράς (Covariance Matrix). Εφόσον έχουμε 2 διανύσματα z_1 και z_2 , ο πίνακας συνδιασποράς θα έχει μέγεθος 2×2 .

Ο πίνακας συνδιασποράς υπολογίζεται από την εξίσωση:

$$C = \frac{1}{n-1} X^T X$$

όπου X^T ο ανάστροφος πίνακας του X με διαστάσεις 2×10 και n το πλήθος των στοιχείων κάθε κεντραρισμένου διανύσματος στήλης(κεντραρισμένης μεταβλητής).

Σε μορφή πίνακα ο C είναι:

$$C = \begin{pmatrix} var(z_1)^2 & covar(z_1, z_2)^2 \\ covar(z_2, z_1)^2 & var(z_2)^2 \end{pmatrix}$$

Η $var(z_1)^2$ είναι η διασπορά του κεντραρισμένου διανύσματος z_1 και έχει τύπο:

$$var(z_1)^2 = \frac{1}{n-1} z_1^T z_1$$

Η $var(z_2)^2$ είναι η διασπορά του κεντραρισμένου διανύσματος z_2 και έχει τύπο:

$$var(z_2)^2 = \frac{1}{n-1} z_2^T z_2$$

Η $covar(z_1, z_2)^2 = covar(z_2, z_1)^2$ είναι η συνδιασπορά των διανυσμάτων z_1 και z_2 .

Έχει τύπο: $covar(z_1, z_2)^2 = covar(z_2, z_1)^2 = \frac{1}{n-1} z_1^T z_2 = \frac{1}{n-1} z_2^T z_1$.

Ο πίνακας συνδιασποράς C του παραδείγματός μας είναι:

```
Covariance-matrix(C):
[[ 9.16666667 10.22222222]
 [10.22222222 11.82222222]]
```

ΒΗΜΑ 3°

Στο τρίτο βήμα του αλγορίθμου βρίσκουμε τις ιδιοτιμές και τα ιδιοδιανύσματα του πίνακα συνδιασποράς C. Αρχικά βρίσκουμε τις ιδιοτιμές λ από την εξίσωση: $\det|C-\lambda I| = 0$. Και στη συνέχεια, τις χρησιμοποιούμε στην εξίσωση $Cv = \lambda v$, με σκοπό να βρούμε τα ιδιοδιανύσματα.

ΒΗΜΑ 4°

Το ιδιοδιάνυσμα με τη μεγαλύτερη ιδιοτιμή είναι αυτό που περιέχει την περισσότερη πληροφορία. Επομένως, μπορούμε να το κρατήσουμε και να απορρίψουμε το άλλο ιδιοδιάνυσμα. Μετά δημιουργούμε τον πίνακα ιδιοδιανυσμάτων V, ο οποίος έχει διαστάσεις στο συγκεκριμένο παράδειγμά μας 2×1 . Κοινώς είναι ένα διάνυσμα στήλης. Και περιέχει μόνο το ιδιοδιάνυσμα που κρατήσαμε, αυτό δηλαδή με τη μεγαλύτερη ιδιοτιμή.

Τα αποτελέσματα του παραδείγματος είναι:

```
Eigenvectors
[[-0.75126867 -0.6599965 ]
 [ 0.6599965  -0.75126867]]

Eigenvalues
[ 0.18634952 20.80253937]

We calculate which eigenvalue has the most information!

Percentage of first eigenvalue: 0.00887848423973284

Percentage of second eigenvalue: 0.9911215157602672

Eigenvectors-matrix(V):
[[ 0.6599965 ]
 [-0.75126867]]
```

Όπως φαίνεται παραπάνω, η ιδιοτιμή $e_1 = 0.18$ έχει το 0,8% της πληροφορίας ενώ η ιδιοτιμή $e_2 = 20.8$ έχει το 99,2% της πληροφορίας. Επομένως στον πίνακα ιδιοδιανυσμάτων V κρατάμε το ιδιοδιάνυσμα της e_2 ιδιοτιμής μόνο και απορρίπτουμε αυτό της e_1 .

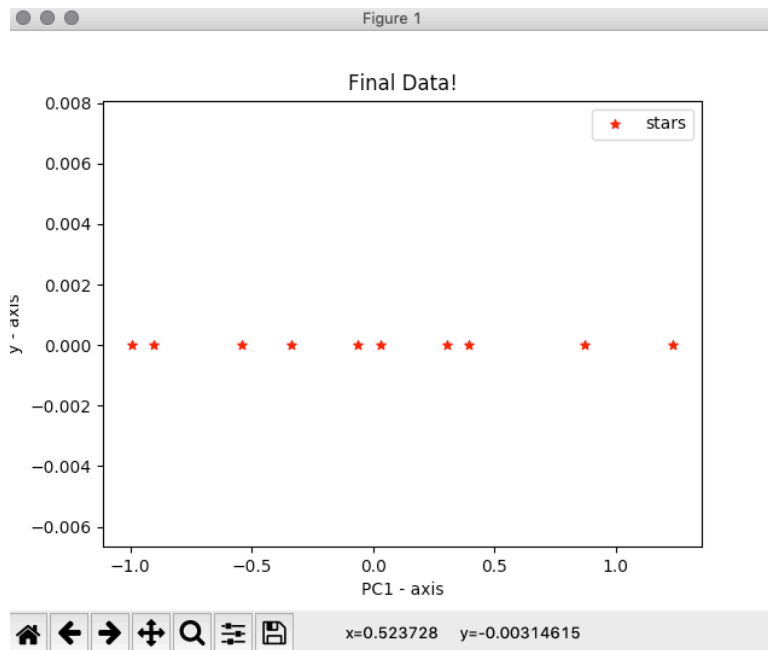
ΒΗΜΑ 5^ο

Στο τελευταίο βήμα υπολογίζονται τα τελικά δεδομένα που αποτελούν ένα διάνυσμα στήλης 10×1 . Τα τελικά δεδομένα υπολογίζονται ως εξής: $Y = XV \rightarrow 10 \times 1 = 10 \times 2 * 2 \times 1$.

Τα τελικά δεδομένα του παραδείγματος είναι:

```
Final Data Y = XV:
[[ 1.23712029]
 [ 0.39457945]
 [ 0.30330728]
 [-0.53923355]
 [ 0.87203162]
 [ 0.02949078]
 [-0.06178138]
 [-0.90432222]
 [-0.99559439]
 [-0.33559788]]
```


Το διάγραμμα των τελικών δεδομένων είναι:



Το πολύ σημαντικό συμπέρασμα, το οποίο απορρέει από το συγκεκριμένο παράδειγμα, είναι ότι μειώσαμε τις 2-διαστάσεις δεδομένων σε 1-διάσταση δεδομένων και αυτό το πετύχαμε χωρίς να χαθεί μεγάλο κομμάτι της πληροφορίας, που πήγαζε από τα δεδομένα του παραδείγματος. Μάλιστα, διατηρήθηκε και το 99,2% της πληροφορίας.

4 ΚΒΑΝΤΙΚΟΣ ΑΛΓΟΡΙΘΜΟΣ ΑΝΑΛΥΣΗΣ ΚΥΡΙΩΝ ΣΥΝΙΣΤΩΣΩΝ

Σε αυτό το κεφάλαιο θα δούμε τον κβαντικό ανάλογο του κλασσικού αλγορίθμου της Ανάλυσης Κύριων Συνιστωσών(Principal Component Analysis - PCA), που είναι ο κβαντικός αλγόριθμος της Ανάλυσης Κύριων Συνιστωσών(Quantum Principal Component Analysis - QPCA). Στο προηγούμενο κεφάλαιο είδαμε ότι ο αλγόριθμος PCA βασίζεται στην ιδιο-αποσύνθεση του πίνακα συνδιασποράς. Επομένως, υπάρχει η δυνατότητα σε αυτό το σημείο να επιτευχθεί κβαντική επιτάχυνση. Ο κβαντικός αλγόριθμος της Ανάλυσης Κύριων Συνιστωσών παρουσιάστηκε από τους Lloyd, Mohseni και Rebentrost ^{*([\[Δ2\]](#))} και είναι εκθετικά ταχύτερος από οποιονδήποτε κλασσικό αλγόριθμο.

Το ‘κλειδί’ είναι ότι χρησιμοποιεί την κβαντική υπορουτίνα PhaseEstim σε έναν εκθέτη του πίνακα πυκνότητας ρ . Παρακάτω θα κατανοήσουμε καλύτερα, γιατί αυτό είναι σημαντικό για την κβαντική επιτάχυνση του αλγορίθμου. Η κβαντική υπορουτίνα PhaseEstim ^{*([\[Δ6\]](#))} υπολογίζει ιδιοτιμές και ιδιοδιανύσματα. Ο αλγόριθμος QPCA παράγει τις κβαντικές καταστάσεις, οι οποίες περιέχουν όλες τις ιδιοτιμές και τα ιδιοδιανύσματα του πίνακα πυκνότητας ρ . Μπορούμε σιγά – σιγά να φανταζόμαστε το πίνακα πυκνότητας ρ σαν τον πίνακα συνδιασποράς C .

4.1 Βήματα Αλγορίθμου QPCA

Σε αυτή την ενότητα θα ανακαλύψουμε τα βήματα και τα μαθηματικά ^{*(_[Δ6])} του κβαντικού αλγορίθμου QPCA. Τα αρχικά κλασσικά δεδομένα είναι τα N διαστατικά διανύσματα προπόνησης (training vectors). Για κάθε παράδειγμα προπόνησης $x^{(i)}$, με $i = \{1, \dots, M\}$ έχουμε:

$$x^{(i)} \rightarrow x^{(i)} - x'$$

$$x' = \frac{1}{M} \sum_{i=1}^M x^{(i)}$$

Πριν κάνουμε οτιδήποτε άλλο πρέπει να κανονικοποιήσουμε τα αρχικά κλασσικά δεδομένα ως εξής:

$$x^{(i)} \rightarrow \frac{x^{(i)}}{|x^{(i)}|}$$

$$|x^{(i)}| = \sqrt{\sum_{k=1}^N x_k^2}$$

Εν συνεχεία, θα παρουσιάσουμε τα κλασσικά δεδομένα σαν κβαντικές καταστάσεις:

$$x \rightarrow |x\rangle = \sum_{k=1}^N x_k |k\rangle$$

$$\langle x|x\rangle = 1$$

$$N = \log_2 n \text{ qubits}$$

Παρατηρούμε ότι τα στοιχεία x_k του κλασσικού διανύσματος προπόνησης x είναι κωδικοποιημένα σαν πλάτη των κβαντικών καταστάσεων. Κατόπιν, θα δούμε το κβαντικό ανάλογο του πίνακα συνδιασποράς, που είναι ο πίνακας πυκνότητας, και τη μορφή του.

Ο πίνακας πυκνότητας ορίζεται από την:

$$\rho = \frac{1}{M} \sum_{i=1}^M |x^{(i)}\rangle \langle x^{(i)}|$$

Το $|x^{(i)}\rangle\langle x^{(i)}|$ υπολογίζεται ως εξής:

$$|x^{(i)}\rangle\langle x^{(i)}| = \sum_{k=1}^N \sum_{m=1}^N x_k^{(i)} x_m^{(i)} |k\rangle\langle m| = \begin{pmatrix} x_1^{(i)} x_1^{(i)} & x_1^{(i)} x_2^{(i)} & \dots & x_1^{(i)} x_N^{(i)} \\ x_2^{(i)} x_1^{(i)} & x_2^{(i)} x_2^{(i)} & \dots & x_2^{(i)} x_N^{(i)} \\ \vdots & \vdots & \ddots & \vdots \\ x_N^{(i)} x_1^{(i)} & x_N^{(i)} x_2^{(i)} & \dots & x_N^{(i)} x_N^{(i)} \end{pmatrix}$$

Επομένως ο πίνακας πυκνότητας ρ είναι:

$$\rho = \frac{1}{M} \begin{pmatrix} \sum_i x_1^{(i)} x_1^{(i)} & \sum_i x_1^{(i)} x_2^{(i)} & \dots & \sum_i x_1^{(i)} x_N^{(i)} \\ \sum_i x_2^{(i)} x_1^{(i)} & \sum_i x_2^{(i)} x_2^{(i)} & \dots & \sum_i x_2^{(i)} x_N^{(i)} \\ \vdots & \vdots & \ddots & \vdots \\ \sum_i x_N^{(i)} x_1^{(i)} & \sum_i x_N^{(i)} x_2^{(i)} & \dots & \sum_i x_N^{(i)} x_N^{(i)} \end{pmatrix}$$

Έχουμε φτάσει τώρα στο σημείο, που πρέπει να βρούμε τα ιδιοδιανύσματα του πίνακα πυκνότητας ρ . Αυτό μπορεί να γίνει μέσω της κβαντικής υπορουτίνας PhaseEstim ^{*([Δ6])}. Θα μιλήσουμε για αυτήν πιο εντατικά λίγο παρακάτω. Εδώ υπάρχει ένα μικρό πρόβλημα, διότι η κβαντική υπορουτίνα PhaseEstim δέχεται σαν είσοδο, έναν μοναδιαίο πίνακα U και ο πίνακας πυκνότητας ρ δεν πληρεί τις προϋποθέσεις, για να χρησιμοποιηθεί σαν είσοδος. Μένει επομένως το ερώτημα του τι κάνουμε σε αυτή την περίπτωση. Η απάντηση έγκειται στο μαθηματικό νόμο, ο οποίος λέει ότι ο $U = e^{iH}$ είναι μοναδιαίος πίνακας για κάθε ερμιτιανό πίνακα H . Ο πίνακας πυκνότητας ρ είναι ερμιτιανός, άρα αρχίζουμε να χτίζουμε τον $U = e^{i\rho}$. Μετέπειτα εκθετικοποιούμε τον πίνακα πυκνότητας ρ και παίρνουμε τον εκθέτη: $U = e^{-i\rho t}$, όπου t είναι ένας επιπρόσθετος παράγοντας χρόνου. Η χρονική πολυπλοκότητα της παραπάνω διαδικασίας, δηλαδή της δημιουργίας του εκθετικού πίνακα πυκνότητας είναι: $O(\log N)$. Εφόσον, μιλήσαμε για όλα αυτά παραπάνω, πρέπει να σημειωθεί ότι τα ιδιοδιανύσματα του πίνακα πυκνότητας ρ είναι επίσης ιδιοδιανύσματα του εκθετικού πίνακα $U = e^{-i\rho t}$, και οι ιδιοτιμές του εκθετικοποιούνται: $e^{-i\lambda t}$.

Ήρθε η ώρα να μιλήσουμε για την σημαντική διαφορά ανάμεσα στον πρότυπο κβαντικό αλγόριθμο εκτίμησης φάσης(QPE) και στην κβαντική υπορουτίνα PhaseEstim, την οποία χρησιμοποιεί ο κβαντικός αλγόριθμος QPCA. Ο πρότυπος κβαντικός αλγόριθμος εκτίμησης φάσης παίρνει έναν μοναδιαίο πίνακα U και τον δρα

σε ένα ιδιοδιάνυσμα, πραγματοποιώντας τον μετασχηματισμό: $|0\rangle^{\otimes n}|\varphi\rangle \rightarrow |2^n\theta\rangle|\varphi\rangle$, όπου:

- Το $|\varphi\rangle$ είναι ένα ιδιοδιάνυσμα του U .
- Στο qubit ελέγχου το θ αποτελεί φάση, που μάλιστα μας επιτρέπει να υπολογίσουμε την ιδιοτιμή $\lambda = e^{2\pi i\theta}$.
- Ο n είναι ο αριθμός των qubits ελέγχου, που αποθηκεύουν την εκτίμηση φάσης θ .

Όμως, ο κβαντικός αλγόριθμος QPCA χρησιμοποιεί τον πρότυπο κβαντικό αλγόριθμο εκτίμησης φάσης(QPE) λίγο διαφορετικά. Γι' αυτό το λόγο και η δημιουργία της κβαντικής υπορουτίνας PhaseEstim, που ο κβαντικός αλγόριθμος QPCA χρησιμοποιεί. Η κβαντική υπορουτίνα PhaseEstim, αντί για ιδιοδιάνυσμα, δρα την μοναδιαία πύλη $U = e^{-i\rho t}$ στον πίνακα πυκνότητας ρ . Πιο απλά, με άλλα λόγια ο πρότυπος κβαντικός αλγόριθμος εκτίμησης φάσης έχει ως είσοδο ένα γνωστό ιδιοδιάνυσμα $|\varphi\rangle$, ενώ η κβαντική υπορουτίνα PhaseEstim έχει ως είσοδο ένα πίνακα πυκνότητας ρ , δηλαδή μια μικτή κατάσταση με M άγνωστα ιδιοδιανύσματα $|\varphi^{(i)}\rangle$. Για να δούμε τώρα πως δουλεύει αυτό με κάτι απλό, αρχικά θα δράσουμε τον $U = e^{-i\rho t}$ στην pure state $|x^{(i)}\rangle$:

$$\begin{aligned} U|x^{(i)}\rangle &= e^{-i\rho t}|x^{(i)}\rangle = \sum_{j=1}^M e^{-i\lambda^{(j)}t} |\varphi^{(j)}\rangle\langle\varphi^{(j)}|x^{(i)}\rangle = \sum_{j=1}^M e^{-i\lambda^{(j)}t} \langle\varphi^{(j)}|x^{(i)}\rangle |\varphi^{(j)}\rangle \\ &= \sum_{j=1}^M c^{(i,j)} |\varphi^{(j)}\rangle \end{aligned}$$

Επομένως, ο μετασχηματισμός που θα πραγματοποιηθεί θα είναι ο παρακάτω και η έξοδος του θα είναι μια υπέρθεση: $|0\rangle^{\otimes n}|x^{(i)}\rangle \rightarrow \sum_{j=1}^M c^{(i,j)} |\lambda^{(j)'}\rangle |\varphi^{(j)}\rangle$, όπου:

- $\lambda^{(j)'}$ \rightarrow αποτελεί την κατάσταση ελέγχου
- Από την εξίσωση $\lambda^{(j)'} = 2^n \frac{\lambda^{(j)}t}{2\pi}$ μπορούμε κατευθείαν να υπολογίσουμε την ιδιοτιμή $\lambda^{(j)}$.

Η κατάσταση εξόδου του μετασχηματισμού μπορεί να γραφτεί ως αναπαράσταση του πίνακα πυκνότητας και έχει την παρακάτω μορφή:

$$d^{(i)} = \sum_{j=1}^M |c^{(i,j)}|^2 |\lambda^{(j)'}\rangle\langle\lambda^{(j)'}| \otimes |\varphi^{(j)}\rangle\langle\varphi^{(j)}|$$

Στον κβαντικό αλγόριθμο QPCA τώρα, αντί για μια pure state $|x^{(i)}\rangle$ χρησιμοποιούμε μια μικτή κατάσταση (mixed state) $\rho = \frac{1}{M} \sum_{i=1}^M |x^{(i)}\rangle \langle x^{(i)}|$, η οποία μπορεί να περιγραφεί ως ένας πίνακας πυκνότητας.

Ύστερα από τις απαραίτητες πράξεις ο τελικός πίνακας πυκνότητας έχει τη μορφή:

$$d = \sum_{j=1}^M \sum_{i=1}^M \frac{1}{M} |c^{(i,j)}|^2 |\lambda^{(j)'}\rangle \langle \lambda^{(j)'}| \otimes |\varphi^{(j)}\rangle \langle \varphi^{(j)}|$$

Όπου: $|c^{(i,j)}|^2 = c^{(i,j)} c^{*(i,j)}$ Με: $c^{(i,j)} = e^{-i\lambda^{(j)}t} \langle \varphi^{(j)} | x^{(i)} \rangle$ και $c^{*(i,j)} = e^{i\lambda^{(j)}t} \langle x^{(i)} | \varphi^{(j)} \rangle$

Άρα μπορούμε να υπολογίσουμε το $\sum_{i=1}^M \frac{1}{M} |c^{(i,j)}|^2$:

$$\begin{aligned} \sum_{i=1}^M \frac{1}{M} |c^{(i,j)}|^2 &= \sum_{i=1}^M \frac{1}{M} c^{(i,j)} c^{*(i,j)} = \sum_{i=1}^M \frac{1}{M} e^{-i\lambda^{(j)}t} \langle \varphi^{(j)} | x^{(i)} \rangle e^{i\lambda^{(j)}t} \langle x^{(i)} | \varphi^{(j)} \rangle \\ &= \langle \varphi^{(j)} | \left(\frac{1}{M} \sum_{i=1}^M |x^{(i)}\rangle \langle x^{(i)}| \right) | \varphi^{(j)} \rangle = \langle \varphi^{(j)} | \rho | \varphi^{(j)} \rangle = \lambda^{(j)} \end{aligned}$$

Τελικά ο πίνακας πυκνότητας:

$$d = \sum_{j=1}^M \lambda^{(j)} |\lambda^{(j)'}\rangle \langle \lambda^{(j)'}| \otimes |\varphi^{(j)}\rangle \langle \varphi^{(j)}|$$

Η χρονική πολυπλοκότητα της κβαντικής υπορουτίνας PhaseEstim είναι συγκρίσιμη με αυτήν της δημιουργίας του εκθετικού πίνακα πυκνότητας, δηλαδή $O(\log N)$. Επομένως, η συνολική χρονική πολυπλοκότητα του κβαντικού αλγορίθμου δεν αυξάνεται μέχρι αυτό το σημείο.

Τώρα θα περάσουμε στη δειγματοληψία της τελικής κατάστασης:

$$d = \sum_{j=1}^M \lambda^{(j)} |\lambda^{(j)'}\rangle \langle \lambda^{(j)'}| \otimes |\varphi^{(j)}\rangle \langle \varphi^{(j)}|$$

Ας υποθέσουμε, ότι υπάρχουν R κύριες συνιστώσες (principal components) και ότι το άθροισμα των ιδιοτιμών που αντιστοιχεί σε αυτές είναι κοντά στο 100%. Τότε η δειγματοληψία από την τελική κατάσταση, μας δίνει το ιδιοδιάνυσμα $|\varphi^{(j)}\rangle$ και την ιδιοτιμή $\lambda^{(j)}$ με πιθανότητα $\lambda^{(j)}$. Αν έχουμε m αντίγραφα της τελικής κατάστασης, τότε το ιδιοδιάνυσμα $|\varphi^{(j)}\rangle$ θα γίνει δειγματοληψία κατά μέσο όρο $m \cdot \lambda^{(j)}$ φορές. Με βασική προϋπόθεση ότι το $m \ll R$, τότε η χρονική πολυπλοκότητα της δειγματοληψίας είναι $O(R \log N)$. Τελικώς, θα δούμε τις κύριες συνιστώσες και τα αποτελέσματα. Τα

$|\varphi^{(j)}\rangle$ αποτελούν τις δειγματοληφθείσες κύριες συνιστώσες (PC's). Τα αποτελέσματα μπορούν να ληφθούν από τον υπολογισμό της προβολής ενός ιδιοδιανύσματος πάνω σε ένα διάνυσμα προπόνησης (training vector):

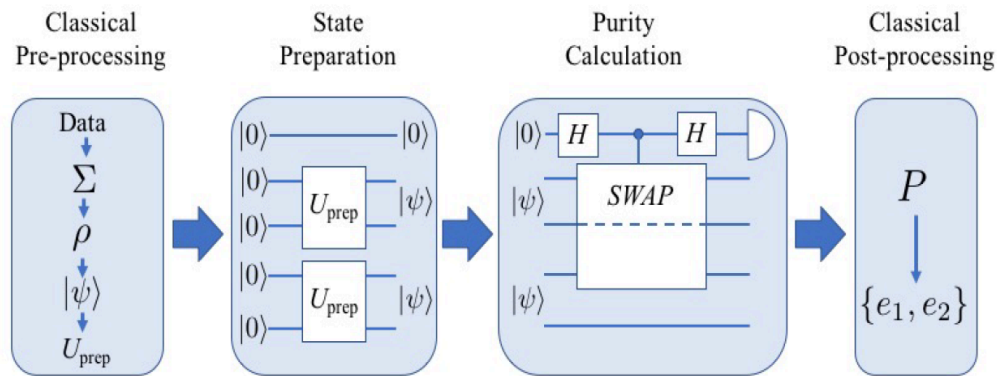
$$\langle x^{(i)} | \varphi^{(j)} \rangle = \sum_{k=1}^N \sum_{m=1}^N x_k^{(i)} \varphi_m^{(j)} \langle k | m \rangle = \sum_{k=1}^N x_k^{(i)} \varphi_k^{(j)} = s_i^{(j)}$$

για κάθε $i = \{1, \dots, M\}$ διανύσματος προπόνησης(training vector) κωδικοποιημένο σε κβαντική κατάσταση. Τελικά το αποτέλεσμα για την j κύρια συνιστώσα(PC) είναι ένα διάνυσμα στήλης: $S^{(j)} = (s_1^{(j)} \ s_2^{(j)} \ \dots \ s_M^{(j)})^\top$.

Για να κλείσει αυτή η παράγραφος είναι σημαντικό να σημειωθεί ότι τα R πρώτα αποτελέσματα(PC's) αντιστοιχούν στις μεγαλύτερες ιδιοτιμές, αναπαριστούν τα συμπίεσμένα δεδομένα και μπορούν να χρησιμοποιηθούν σε άλλους αλγορίθμους μηχανικής μάθησης.

4.2 Κβαντικό Κύκλωμα QPCA

Σε αυτή τη παράγραφο θα παρουσιαστεί το κβαντικό κύκλωμα του κβαντικού αλγορίθμου QPCA ^{*([Δ7])}, το οποίο έχει τη παρακάτω μορφή.



Αποτελείται από 4 στάδια ^{*([Δ7])}:

- Την κλασσική προ-επεξεργασία
- Την κατάσταση προετοιμασίας
- Τον υπολογισμό προβολικοποίησης
- Την κλασσική μετά-επεξεργασία

Πάμε να δούμε τι γίνεται στο 1^ο στάδιο του κβαντικού κυκλώματος, αυτό της κλασσικής προ-επεξεργασίας. Αρχικά τα διανύσματα των ακατέργαστων δεδομένων(Data) μετατρέπονται στον πίνακα συνδιασποράς Σ . Για να καταλάβουμε τα μεγέθη, αν έχουμε 2 μεταβλητές-χαρακτηριστικά, τότε ο πίνακας συνδιασποράς Σ θα έχει διαστάσεις 2×2 . Μετέπειτα κανονικοποιούμε τον Σ στην μορφή $\rho = \Sigma / \text{Tr}(\Sigma)$. Στη συνέχεια, προβολικοποιούμε τον πίνακα πυκνότητας ρ , για να δημιουργήσουμε μια pure state $|\psi\rangle$ σε 2 qubits. Για να καταλάβουμε τα μεγέθη, 2 αντίγραφα της $|\psi\rangle$ χρησιμοποιούν 4 qubits. Η προβολικοποίηση του ρ υπολογίζεται από την: $P = \text{Tr}(\rho^2)$. Στο τέλος του 1^{ου} σταδίου υπολογίζουμε τον μοναδιαίο πίνακα U_{prep} , αφού πρώτα προετοιμαστεί η $|\psi\rangle$ από ένα ζεύγος qubits, που το καθένα αρχικά ήταν στην κατάσταση $|0\rangle$.

Στο 2^ο στάδιο τώρα, αυτό της κατάστασης προετοιμασίας, προετοιμάζουμε τα 2 αντίγραφα της $|\psi\rangle$, χρησιμοποιώντας τον μοναδιαίο πίνακα U_{prep} : $U_{\text{prep}}|00\rangle = |\psi\rangle$. Ο πίνακας U_{prep} βγαίνει από την εξίσωση:

$$U_{\text{prep}} = (U_A \otimes U_B) \text{CNOT}_{AB} (U_{A'} \otimes I_B)$$

Το A αποτελεί το qubit ελέγχου και το B το qubit στόχου. Η μορφή των μοναδιαίων πινάκων $U_A, U_B, U_{A'}$ είναι:

$$U_A = U_B = U_{A'} = \begin{pmatrix} \cos\left(\frac{\theta}{2}\right) & -e^{i\lambda}\sin\left(\frac{\theta}{2}\right) \\ e^{i\varphi}\sin\left(\frac{\theta}{2}\right) & e^{i(\lambda+\varphi)}\cos\left(\frac{\theta}{2}\right) \end{pmatrix}$$

Οι πίνακες αυτοί έχουν διαστάσεις 2×2 . Τα αντίστοιχα θ, λ, φ του κάθε πίνακα υπολογίζονται στο 1^ο στάδιο. Στις κβαντικές πύλες στο 2^ο κεφάλαιο είδαμε τις κβαντικές πύλες CNOT με διαστάσεις 4×4 και I με διαστάσεις 2×2 . Καταλαβαίνουμε, λοιπόν, ότι η U_{prep} θα έχει διαστάσεις 4×4 , λόγω τανυστικού γινομένου. Αυτό είναι απολύτως λογικό, γιατί δρα σε 2 qubits.

Προχωράμε στο 3^ο στάδιο, αυτό του υπολογισμού της προβολικοποίησης. Το κύκλωμα αποτελείται από 5 qubits. Το πρώτο qubit από πάνω προς τα κάτω στο κβαντικό κύκλωμα αποτελεί το βοηθητικό qubit C. Επίσης, ονομάζουμε τα υπόλοιπα 4 qubits με τη σειρά ως A,B,A',B'. Σε αυτό το στάδιο:

- Αρχικά δρα μια κβαντική πύλη Hadamard στο βοηθητικό qubit C.

- Στη συνέχεια, δρα μια κβαντική πύλη CSWAP σε 3 qubits, με το βοηθητικό qubit C να αποτελεί το qubit ελέγχου και τα qubits A,A' να είναι τα qubits στόχου.
- Τελικά, δρα πάλι μια Hadamard στο qubit C.

Το βοηθητικό qubit C μετريέται στην Z βάση: $\langle Z \rangle_c = p_0 - p_1 = \text{Tr}(\rho^2) = P$.

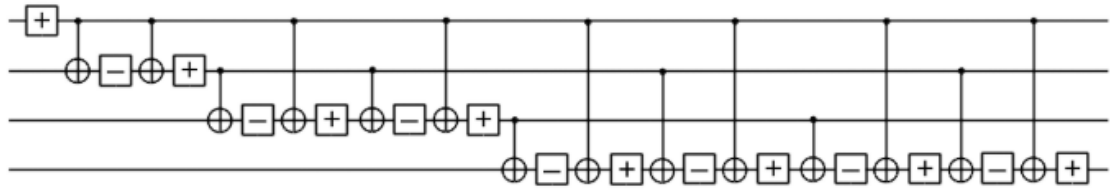
- Η p_0 είναι η πιθανότητα να μετρηθεί το βοηθητικό qubit C στην κατάσταση $|0\rangle$.
- Η p_1 είναι η πιθανότητα να μετρηθεί το βοηθητικό qubit C στην κατάσταση $|1\rangle$.

Μιας και σε αυτό το στάδιο είδαμε τη πύλη CSWAP και επειδή θα χρειαστούμε την αποσύνθεσή της μετέπειτα στην υλοποίηση του κβαντικού αλγορίθμου QPCA μέσω της πλατφόρμας του Qiskit της IBM, θα την παρουσιάσουμε σε αυτό το σημείο.

Πρώτον θα γίνει αποσύνθεση της CSWAP ως εξής:

$$CSWAP_{CAA'} = (I_C \otimes CNOT_{AA'}) Toffoli_{CAA'} (I_C \otimes CNOT_{A'A})$$

Και δεύτερον θα γίνει αποσύνθεση της $Toffoli_{CAA'}$, όπως φαίνεται στην εικόνα:



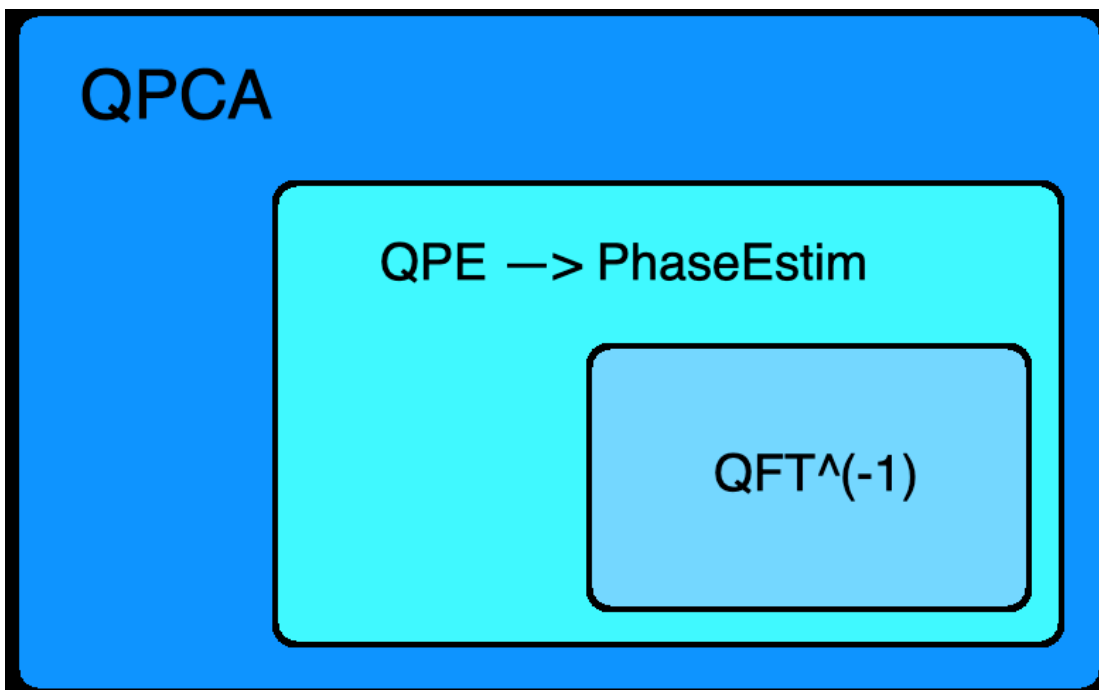
Δηλαδή η αποσύνθεση θα γίνει με 14 πύλες CNOT και 15 πύλες Single-qubit Z

Rotations: $R_Z\left(\pm \frac{\pi}{16}\right) = \begin{pmatrix} 1 & 0 \\ 0 & e^{\pm i\frac{\pi}{8}} \end{pmatrix}$.

Στο 4^ο και τελευταίο στάδιο, αυτό της κλασσικής μετά-επεξεργασίας, υπολογίζονται οι ιδιοτιμές του πίνακα συνδιασποράς Σ με τους εξής παρακάτω τύπους, όπου $P = p_0 - p_1$ που έχει υπολογιστεί στο 3^ο στάδιο:

- $e_1 = \text{Tr}(\Sigma) * \left(\frac{1 + \sqrt{1 - 2(1 - P)}}{2}\right)$
- $e_2 = \text{Tr}(\Sigma) * \left(\frac{1 - \sqrt{1 - 2(1 - P)}}{2}\right)$

Στο παρακάτω σχήμα παρουσιάζονται οι σχέσεις ανάμεσα στον κβαντικό αλγόριθμο QPCA και στις δύο από τις πιο σημαντικές κβαντικές υπορουτίνες QPE και QFT, οι οποίες παρουσιάστηκαν στις ενότητες 2.8 και 2.9 του 2^{ου} κεφαλαίου. Ο σκοπός είναι, μέσω του σχήματος, να αποκτήσουμε μια καλύτερη οπτικοποίηση στο πώς σχετίζονται μεταξύ τους. Από το σχήμα παρατηρούμε εύκολα ότι ο αντίστροφος του κβαντικού αλγορίθμου QFT χρησιμοποιείται σαν κβαντική υπορουτίνα στο κβαντικό κύκλωμα του κβαντικού αλγορίθμου QPE. Και στη συνέχεια ότι ο κβαντικός αλγόριθμος QPE χρησιμοποιείται λίγο διαφορετικά από τον QPCA σαν την κβαντική υπορουτίνα PhaseEstim, όπως προαναφέρθηκε και θεωρητικά παραπάνω.



5 ΥΛΟΠΟΙΗΣΗ ΑΛΓΟΡΙΘΜΩΝ ΣΕ ΚΛΑΣΣΙΚΟ ΚΑΙ ΚΒΑΝΤΙΚΟ HARDWARE

Σε αυτό το κεφάλαιο θα παρουσιαστούν οι υλοποιήσεις του κλασσικού αλγορίθμου PCA και του κβαντικού αλγορίθμου QPCA πάνω σε πραγματικά δεδομένα διαφόρων σπιτιών, που είναι προς πώληση, με σκοπό να δούμε ποιο θα αγοράσουμε. Οι δυο αυτοί αλγόριθμοι θα μας βοηθήσουν, καθώς θα μειώσουν τον όγκο των αρχικών δεδομένων διατηρώντας παράλληλα το μεγαλύτερο ποσοστό πληροφορίας που χρειαζόμαστε για την απόφασή μας. Τα αρχικά δεδομένα περιλαμβάνουν δύο διανύσματα (μεταβλητές), αυτά του αριθμού δωματίων και τα τετραγωνικά μέτρα δωματίων. Το κάθε διάνυσμα περιέχει 15 στοιχεία. Πιο συγκεκριμένα, ο κλασσικός αλγόριθμος PCA υλοποιείται με δικό μου κώδικα σε γλώσσα προγραμματισμού Python και ο κβαντικός αλγόριθμος QPCA υλοποιείται από εμένα στην πλατφόρμα Qiskit της IBM στον κβαντικό προσομοιωτή της (quantum simulator), στο κβαντικό υλικό της (quantum hardware), αλλά και με δικό μου κώδικα σε γλώσσα προγραμματισμού Python. Πρέπει να επισημανθεί, ότι οι υλοποιήσεις πραγματοποιήθηκαν με τόσο μικρό αριθμό διανυσμάτων(μεταβλητών), δηλαδή μόνο δύο, διότι έπρεπε να συγκριθεί η υλοποίηση του PCA σε Python, η υλοποίηση του QPCA στο Qiskit και η υλοποίηση του QPCA σε Python, πράγμα το οποίο ήταν εφικτό μόνο για δύο διανύσματα, αφού το Qiskit δέχεται ως είσοδο μόνο μέχρι 5 qubits. Για να κατανοήσουμε καλύτερα την

κλίμακα, αν έχουμε δύο διανύσματα, αυτά αντιστοιχούν σε 5 qubits στο κβαντικό κύκλωμα του QPCA, ενώ για παράδειγμα αν έχουμε 3, αυτά αντιστοιχούν σε 7 qubits.

5.1 Υλοποίηση Αλγορίθμου PCA σε πραγματικά προβλήματα

Εφόσον αναλύσαμε παραπάνω τα βήματα του αλγορίθμου, σε αυτή την ενότητα θα δούμε ένα άλλο παράδειγμα πάλι υλοποιημένο με δικό μου κώδικα σε γλώσσα Python. Τα δεδομένα μας αυτή τη φορά είναι 2 διανύσματα των 15 στοιχείων το καθένα. Τα συγκεκριμένα δεδομένα αυτού του παραδείγματος χρησιμοποιήθηκαν παρακάτω και στην υλοποίηση του κβαντικού αλγορίθμου της Ανάλυσης Κύριων Συνιστωσών (QPCA), στην πλατφόρμα Qiskit της IBM και στην γλώσσα προγραμματισμού Python. Ο κώδικας σε γλώσσα Python υπάρχει στο ΠΑΡΑΡΤΗΜΑ 1 με ονομασία PCA.py.

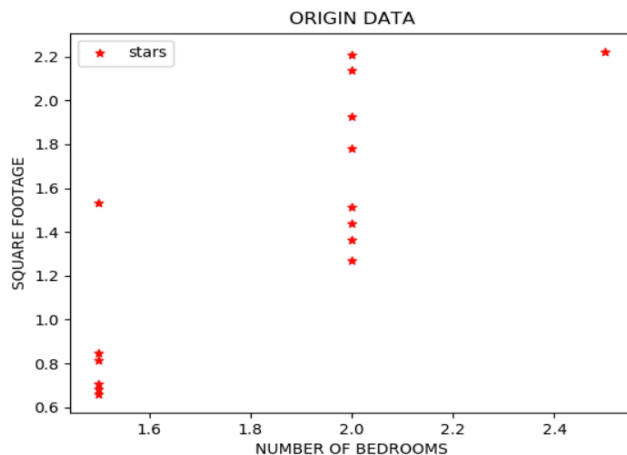
Τα αρχικά δεδομένα(διανύσματα στήλης) είναι: $x_1 \rightarrow$ Αριθμός Δωματίων Διαφόρων Σπιτιών και $x_2 \rightarrow$ Τετραγωνικά Μέτρα Δωματίων Διαφόρων Σπιτιών:

$$x_1 = [2, 1.5, 2, 2, 1.5, 1.5, 1.5, 1.5, 2, 2, 2, 2.5, 2, 1.5, 2]$$

$$x_2 = [1.514, 0.6825, 1.363, 1.869, 0.659, 0.8465, 0.706, 0.816, 1.4375, 1.782, 2.206, 2.222, 2.139, 1.532, 1.9285]$$

```
NUMBER OF BEDROOMS(Origin Data): [2, 1.5, 2, 2, 1.5, 1.5, 1.5, 1.5, 2, 2, 2, 2.5, 2, 1.5, 2]
SQUARE FOOTAGE(Origin Data): [1.514, 0.6825, 1.363, 1.269, 0.659, 0.8465, 0.706, 0.816, 1.4375, 1.782, 2.206, 2.222, 2.139, 1.532, 1.9285]
```

Το διάγραμμα των αρχικών δεδομένων είναι:



ΒΗΜΑ 1^ο

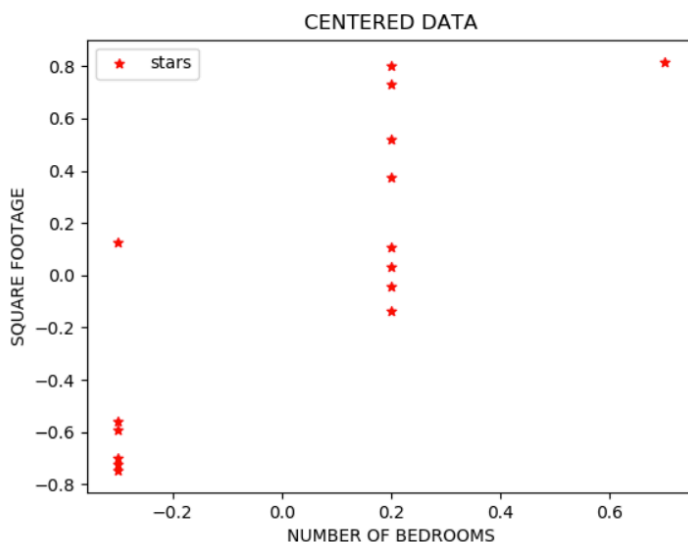
Η μέση τιμή των x_1 και x_2 είναι:

```
The mean of x1 origin data is : 1.8333333333333333
The mean of x2 origin data is : 1.4068666666666667
```

Τα κεντραρισμένα δεδομένα είναι:

```
NUMBER OF BEDROOMS(Centered Data): [0.2, -0.3, 0.2, 0.2, -0.3, -0.3, -0.3, -0.3, 0.2, 0.2, 0.2, 0.7, 0.2, -0.3, 0.2]
SQUARE FOOTAGE(Centered Data): [0.1075, -0.724, -0.0435, -0.1375, -0.7475, -0.56, -0.7005, -0.5905, 0.031, 0.3755, 0.7995, 0.8155, 0.7325, 0.1255, 0.522]
```

Το διάγραμμα των κεντραρισμένων δεδομένων είναι:



Ο πίνακας $X(15 \times 2)$ είναι:

```
Matrix X
1column --> NUMBER OF BEDROOMS(Centered Data)
2column --> SQUARE FOOTAGE(Centered Data)
X=
[[ 0.2    0.1075]
 [-0.3   -0.724 ]
 [ 0.2   -0.0435]
 [ 0.2   -0.1375]
 [-0.3   -0.7475]
 [-0.3   -0.56  ]
 [-0.3   -0.7005]
 [-0.3   -0.5905]
 [ 0.2    0.031 ]
 [ 0.2    0.3755]
 [ 0.2    0.7995]
 [ 0.7    0.8155]
 [ 0.2    0.7325]
 [-0.3    0.1255]
 [ 0.2    0.522 ]]
```

ΒΗΜΑ 2°

Ο πίνακας συνδιασποράς $C(2 \times 2)$ είναι:

```
COVARIANCE MATRIX
C=
[[0.09642857 0.14338214]
 [0.14338214 0.32423355]]
```

ΒΗΜΑ 3° και ΒΗΜΑ 4°

Οι ιδιοτιμές και τα ιδιοδιανύσματα του πίνακα συνδιασποράς C είναι:

```
Eigenvectors of C
[[-0.90056004 -0.43473166]
 [ 0.43473166 -0.90056004]]

Eigenvalues of C
[0.02721302 0.3934491 ]

Calculate which eigenvalue has the most information
eigenvalue_0=
0.06469092760786363

eigenvalue_1=
0.9353090723921363

EigenVectors Matrix --> eigenvector of eigenvalue_1
V=
[[ 0.43473166]
 [-0.90056004]]
```

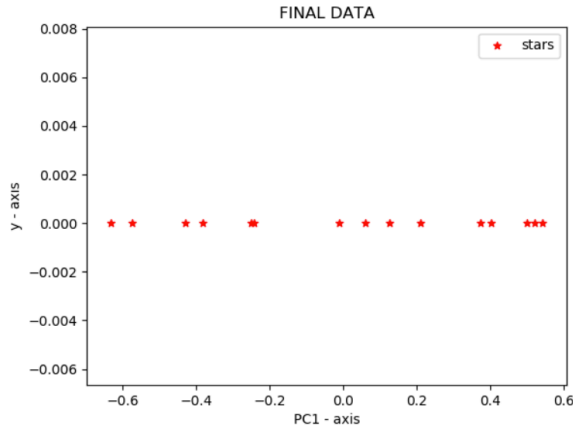
Από το παραπάνω απορρέει και ο πίνακας ιδιοδιανυσμάτων $V(2 \times 1)$.

ΒΗΜΑ 5°

Τα τελικά δεδομένα $Y = XV$ ($15 \times 1 = 15 \times 2 * 2 \times 1$) είναι:

```
FinalData Y = XV
Y=
[[-0.00986387]
 [ 0.52158597]
 [ 0.12612069]
 [ 0.21077334]
 [ 0.54274913]
 [ 0.37389412]
 [ 0.50042281]
 [ 0.40136121]
 [ 0.05902897]
 [-0.25121396]
 [-0.63305142]
 [-0.43009455]
 [-0.5727139 ]
 [-0.24343978]
 [-0.38314601]]
```

Το διάγραμμα των τελικών δεδομένων είναι:



Τα αποτελέσματα που πρέπει να κρατήσουμε από το παραπάνω παράδειγμα είναι:

- Ο πίνακας συνδιασποράς C: $C = \begin{pmatrix} 0.096 & 0.143 \\ 0.143 & 0.324 \end{pmatrix}$
- Το ίχνος του C: $\text{Tr}(C) = 0.096 + 0.324 = 0.42$
- Οι ιδιοτιμές του πίνακα C: $e = \begin{cases} 0.02, & 6.5\% \text{ της πληροφορίας} \\ 0.39, & 93.5\% \text{ της πληροφορίας} \end{cases}$

5.2 Υλοποίηση Αλγορίθμου QPCA στον πρότυπο κβαντικό υπολογιστή της IBM

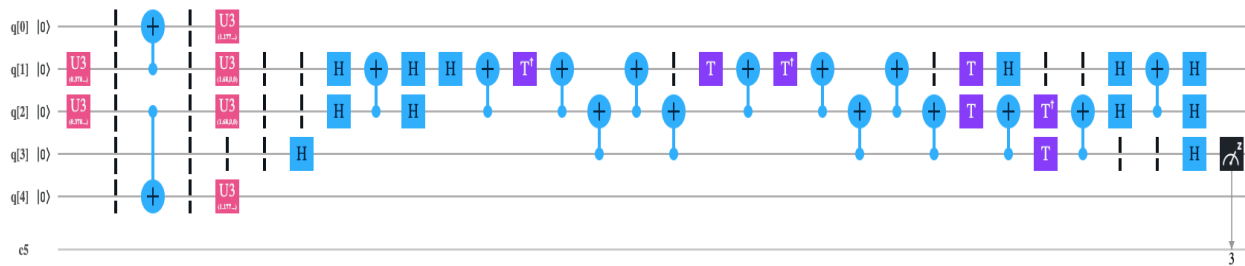
Σε αυτή την ενότητα, θα δούμε την υλοποίηση του κβαντικού κυκλώματος του κβαντικού αλγορίθμου QPCA στην πλατφόρμα Qiskit της IBM. Τα αρχικά δεδομένα, που χρησιμοποιήθηκαν, είναι τα ίδια με αυτά του παραδείγματος του κλασσικού αλγορίθμου PCA παραπάνω.

Τα αρχικά δεδομένα(διανύσματα στήλης) είναι: $x_1 \rightarrow$ Αριθμός Δωματίων Διαφόρων Σπιτιών και $x_2 \rightarrow$ Τετραγωνικά Μέτρα Δωματίων Διαφόρων Σπιτιών:

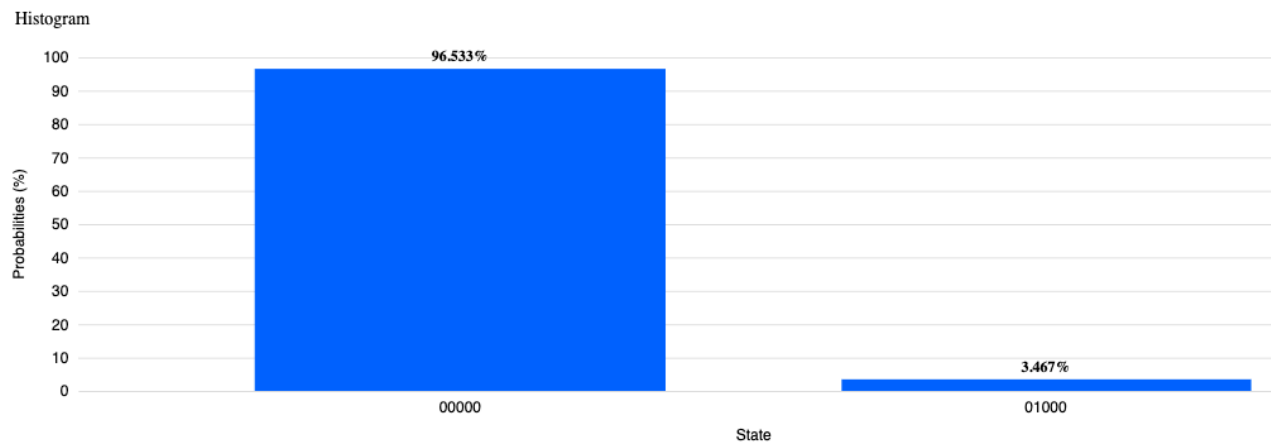
$$x_1 = [2, 1.5, 2, 2, 1.5, 1.5, 1.5, 1.5, 2, 2, 2, 2.5, 2, 1.5, 2]$$

$$x_2 = [1.514, 0.6825, 1.363, 1.869, 0.659, 0.8465, 0.706, 0.816, 1.4375, 1.782, 2.206, 2.222, 2.139, 1.532, 1.9285]$$

Το κβαντικό κύκλωμα του QPCA υλοποιημένο στην πλατφόρμα Qiskit της IBM είναι:



Οι 3 πρώτες χρονοθήκες (time slots) αντιστοιχούν στο 2^ο στάδιο του κβαντικού αλγορίθμου, δηλαδή στην κατάσταση προετοιμασίας. Οι χρονοθήκες που ακολουθούν αντιστοιχούν στο 3^ο στάδιο του κβαντικού αλγορίθμου, δηλαδή στον υπολογισμό προβολικοποίησης. Το qubit $q[3]$ αποτελεί το βοηθητικό qubit και το qubit ελέγχου της CSWAP. Τα qubits $q[1]$ και $q[2]$ αποτελούν τα qubits στόχου της CSWAP. Όπως αναφέρθηκε και προηγουμένως η CSWAP έχει υποστεί αποσύνθεση. Οι χρονοθήκες 7 μέχρι 24 αποτελούν την CSWAP. Σε αυτό το σημείο, θα δούμε τα αποτελέσματα του κβαντικού αλγορίθμου QPCA στον 5-qubits simulator της IBM. Το παρακάτω διάγραμμα μας δείχνει την ποσοστιαία μέτρηση του βοηθητικού qubit $|q[3]\rangle$ στην κβαντική κατάσταση $|0\rangle$ και στην $|1\rangle$.



Πρέπει να επισημανθεί ότι μετράμε μόνο το βοηθητικό qubit $|q[3]\rangle$ από τα 5 qubits της τελικής κβαντικής κατάστασης $|q[4]q[3]q[2]q[1]q[0]\rangle$.

Η πιθανότητα να βρεθεί το qubit $|q[3]\rangle$ στην κατάσταση $|0\rangle$ είναι $p_0 = 96.5\%$.

Η πιθανότητα να βρεθεί το qubit $|q[3]\rangle$ στην κατάσταση $|1\rangle$ είναι $p_1 = 3.5\%$.

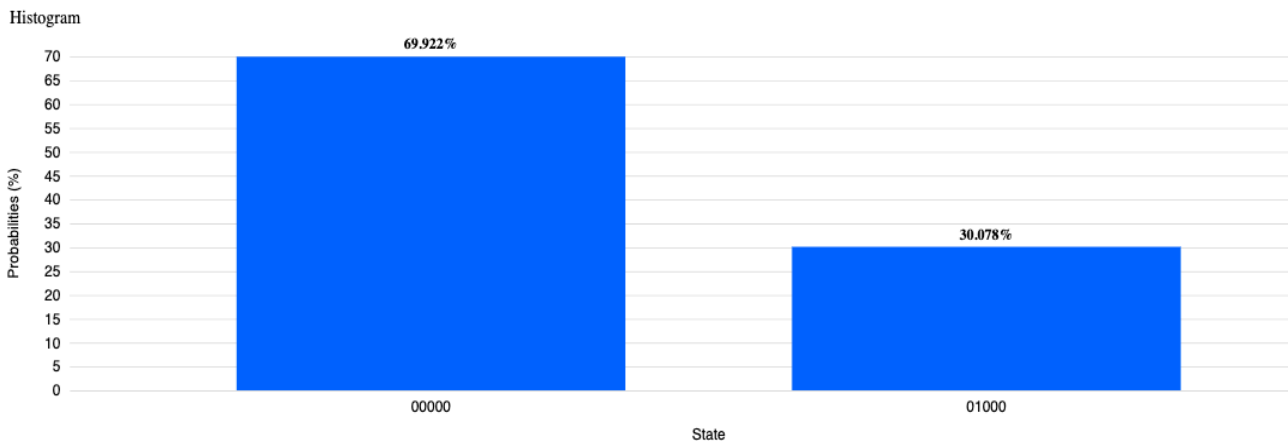
Ο πίνακας συνδιασποράς Σ είναι ίδιος με αυτόν του παραδείγματος του κλασσικού αλγορίθμου PCA: $\Sigma = \begin{pmatrix} 0.096 & 0.143 \\ 0.143 & 0.324 \end{pmatrix}$ με ίχνος $Tr(\Sigma) = 0.096 + 0.324 = 0.42$.

Η προβολικοποίηση είναι: $P = p_0 - p_1 = 0.965 - 0.035 = 0.93 \approx 0.9$.

Επομένως οι ιδιοτιμές που προκύπτουν είναι:

- $e_1 = Tr(\Sigma) * \left(\frac{1 + \sqrt{1 - 2(1 - P)}}{2} \right) = 0.39$
- $e_2 = Tr(\Sigma) * \left(\frac{1 - \sqrt{1 - 2(1 - P)}}{2} \right) = 0.02$

Η προσομοίωση έδωσε τη σωστή απάντηση, αφού οι ιδιοτιμές είναι πραγματικές. Παρατηρούμε, επίσης, ότι οι ιδιοτιμές, που προέκυψαν, είναι ίδιες με αυτές του παραδείγματος του κλασσικού αλγορίθμου PCA. Λογικό μιας και σαν αρχικά δεδομένα έχουμε τα ίδια. Περισσότερα θα ειπωθούν στα συμπεράσματα του 6^{ου} κεφαλαίου. Τώρα, θα δούμε τα αποτελέσματα του κβαντικού αλγορίθμου QPCA στον 5-qubits London, έναν κβαντικό υπολογιστή της IBM. Το παρακάτω διάγραμμα μας δείχνει την ποσοστιαία μέτρηση του βοηθητικού qubit $|q[3]\rangle$ στην κβαντική κατάσταση $|0\rangle$ και στην $|1\rangle$.



Πρέπει να ξανά επισημανθεί ότι μετράμε μόνο το βοηθητικό qubit $|q[3]\rangle$ από τα 5 qubits της τελικής κβαντικής κατάστασης $|q[4]q[3]q[2]q[1]q[0]\rangle$.

Η πιθανότητα να βρεθεί το qubit $|q[3]\rangle$ στην κατάσταση $|0\rangle$ είναι $p_0 = 70\%$.

Η πιθανότητα να βρεθεί το qubit $|q[3]\rangle$ στην κατάσταση $|1\rangle$ είναι $p_1 = 30\%$.

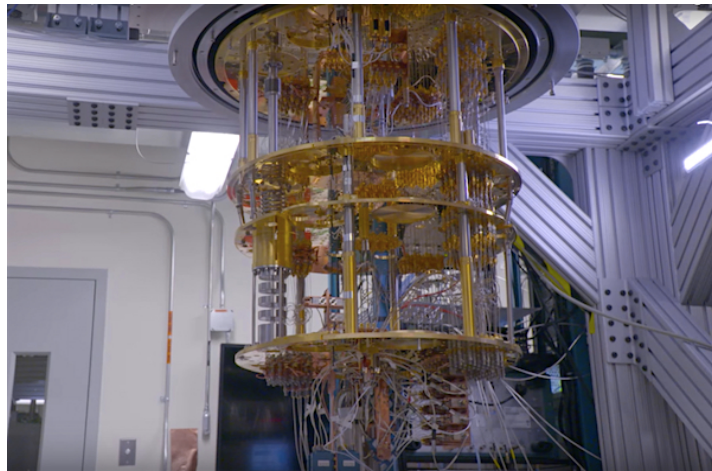
Ο πίνακας συνδιασποράς Σ είναι ίδιος με αυτόν του παραδείγματος του κλασσικού αλγορίθμου PCA: $\Sigma = \begin{pmatrix} 0.096 & 0.143 \\ 0.143 & 0.324 \end{pmatrix}$ με ίχνος $Tr(\Sigma) = 0.096 + 0.324 = 0.42$.

Η προβολικοποίηση είναι: $P = p_0 - p_1 = 0.7 - 0.3 = 0.4$.

Επομένως οι ιδιοτιμές που προκύπτουν είναι:

- $e_1 = Tr(\Sigma) * \left(\frac{1 + \sqrt{1 - 2(1-P)}}{2} \right) = 0.21 + 0.09i$
- $e_2 = Tr(\Sigma) * \left(\frac{1 - \sqrt{1 - 2(1-P)}}{2} \right) = 0.21 - 0.09i$

Η προσομοίωση του κβαντικού αλγορίθμου QPCA σε πραγματικό υλικό (hardware) δεν έδωσε λογικό αποτέλεσμα, αφού οι ιδιοτιμές δεν είναι πραγματικές. Σε αυτό έπαιξε σημαντικό ρόλο η αποσύνθεση της CSWAP, που έκανε το κύκλωμα μεγάλο. Επίσης ένα από τα μεγαλύτερα προβλήματα των κβαντικών υπολογιστών είναι η αποσυνεκτικότητα. Οφείλεται στην αλληλεπίδραση των qubits με το περιβάλλον, που έχει ως αποτέλεσμα τη στροφική εκτροπή της φάσης τους. Ειδικότερα η απώλεια ενέργειας (φυσικός θόρυβος) από το κβαντικό σύστημα προκαλείται από την εκπομπή φωτονίων, δηλαδή από την αποδιέγερση των ηλεκτρονίων στα άτομα. Ο κώδικας για την πλατφόρμα Qiskit της IBM των παραπάνω υλοποιήσεων βρίσκεται στο ΠΑΡΑΡΤΗΜΑ 3.



Εικόνα: Κβαντικός Υπολογιστής 5-qubits London της IBM

5.3 Υλοποίηση Αλγορίθμου QPCA σε Python Simulator

Στην παρούσα ενότητα, θα αναφερθούμε στην υλοποίηση του κβαντικού αλγορίθμου QPCA με δικό μου κώδικα σε γλώσσα Python. Ο κώδικας σε γλώσσα Python υπάρχει στο ΠΑΡΑΡΤΗΜΑ 1 με την ονομασία QPCA.py. Ο κώδικας του συγκεκριμένου αρχείου χρησιμοποιεί και τον κώδικα του αρχείου Quantum_Library.py, που είναι και αυτός γραμμένος από εμένα σε γλώσσα Python και βρίσκεται στο ΠΑΡΑΡΤΗΜΑ 1 με την προαναφερόμενη ονομασία.

ΚΒΑΝΤΙΚΗ ΜΗΧΑΝΙΚΗ ΜΑΘΗΣΗ, ΕΦΑΡΜΟΓΕΣ ΚΑΙ ΥΛΟΠΟΙΗΣΗ ΣΕ ΚΒΑΝΤΙΚΟ HARDWARE

Τα αρχικά δεδομένα που χρησιμοποιήθηκαν είναι τα ίδια με αυτά του παραδείγματος του κλασσικού αλγορίθμου PCA στην ενότητα 5.1.

Τα αρχικά δεδομένα(διανύσματα στήλης) είναι: $x_1 \rightarrow$ Αριθμός Δωματίων Διαφόρων Σπιτιών και $x_2 \rightarrow$ Τετραγωνικά Μέτρα Δωματίων Διαφόρων Σπιτιών:

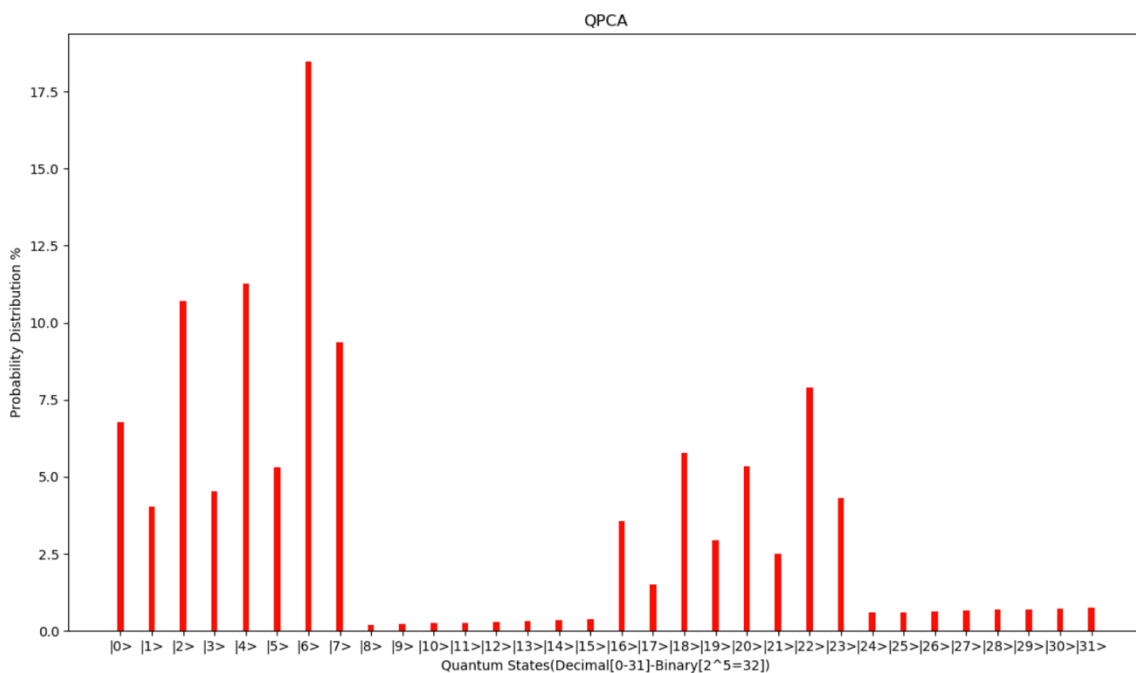
$$x_1 = [2,1.5,2,2,1.5,1.5,1.5,1.5,2,2,2,2.5,2,1.5,2]$$

$$x_2 = [1.514,0.6825,1.363,1.869,0.659,0.8465,0.706,0.816,1.4375,1.782,2.206, \\ 2.222,2.139,1.532,1.9285]$$

Η τελική κατάσταση και η μέτρησή της από την προσομοίωση του κβαντικού αλγορίθμου QPCA σε γλώσσα Python φαίνεται αριθμητικά παρακάτω:

```
Final State:
<The Quantum State that is created is:
[0.23890433-0.10776697j], [0.07008632-0.16036007j], [0.33483175-0.02149152j], [0.15623709-0.1605777j], [0.33483175-0.02149152j], [0.15623709-0.1605777j], [0.41090656+0.12511312j], [0.25956811-0.12202131j], [0.+0.j], [0.+0.j], [0.+0.j], [0.+0.j], [0.+1.38777878e-17j], [0.+0.j], [0.-1.38777878e-17j], [0.+0.j], [0.+0.j], [0.07008632-0.16036007j], [-0.0370888-0.1108186j], [0.15623709-0.1605777j], [0.00548897-0.1495032j], [0.15623709-0.1605777j], [0.00548897-0.1495032j], [0.25956811-0.12202131j], [0.07393939-0.17667391j], [0.+0.j], [0.+0.j], [0.+0.j], [0.+0.j], [0.+0.j], [0.+0.j], [0.+0.j], [0.+0.j]>
The measurement of Final State is:
(6.787109375, 4.0283203125, 10.7177734375, 4.5166015625, 11.279296875, 5.2978515625, 18.45703125, 9.3505859375, 0.1953125, 0.2197265625, 0.244140625, 0.2685546875, 0.29296875, 0.3173828125, 0.341796875, 0.3662109375, 3.564453125, 1.513671875, 5.7861328125, 2.9541015625, 5.3466796875, 2.5146484375, 7.91015625, 4.3212890625, 0.5859375, 0.6103515625, 0.634765625, 0.6591796875, 0.68359375, 0.7080078125, 0.732421875, 0.7568359375)
```

Και εδώ σε μορφή διαγράμματος:



Πρέπει να επισημανθεί ότι μετράμε και τα 5 qubits της τελικής κβαντικής κατάστασης $|q[4]q[3]q[2]q[1]q[0]\rangle$. Οι κβαντικές καταστάσεις που προκύπτουν είναι $2^5 = 32$. Δηλαδή, από $|0\rangle$ έως $|31\rangle$ σε δεκαδική αρίθμηση. Σε δυαδική αρίθμηση κάθε κβαντική κατάσταση περιγράφεται από 5 qubits. Ένα παράδειγμα για να το κατανοήσουμε καλύτερα είναι ότι η κβαντική κατάσταση π.χ. $|3\rangle$ σε δεκαδική αρίθμηση αντιστοιχεί στην $|00011\rangle$ στην δυαδική.

Επομένως για να βρούμε την πιθανότητα να βρεθεί το qubit $|q[3]\rangle$ στην κατάσταση $|0\rangle$ πρέπει να αθροίσουμε τις πιθανότητες των κβαντικών καταστάσεων $|0\rangle$ έως $|7\rangle$ και $|16\rangle$ έως $|23\rangle$, διότι είναι οι κβαντικές καταστάσεις, όπου το qubit $|q[3]\rangle$ βρίσκεται στην κατάσταση $|0\rangle$. Τελικά η πιθανότητα είναι $p_0 = 92.5\%$. Αντίστοιχα για να βρούμε την πιθανότητα να βρεθεί το qubit $|q[3]\rangle$ στην κατάσταση $|1\rangle$ πρέπει να αθροίσουμε τις πιθανότητες των κβαντικών καταστάσεων $|8\rangle$ έως $|15\rangle$ και $|24\rangle$ έως $|31\rangle$, διότι είναι οι κβαντικές καταστάσεις, όπου το qubit $|q[3]\rangle$ βρίσκεται στην κατάσταση $|1\rangle$. Τελικά η πιθανότητα είναι $p_1 = 7.5\%$. Ο πίνακας συνδιασποράς Σ είναι ίδιος με αυτόν του παραδείγματος του κλασσικού αλγορίθμου PCA:

$$\Sigma = \begin{pmatrix} 0.096 & 0.143 \\ 0.143 & 0.324 \end{pmatrix} \text{ με ίχνος } Tr(\Sigma) = 0.096 + 0.324 = 0.42.$$

Η προβολικοποίηση είναι: $P = p_0 - p_1 = 0.925 - 0.075 = 0.85 \approx 0.9$.

Επομένως οι ιδιοτιμές που προκύπτουν είναι:

- $e_1 = Tr(\Sigma) * \left(\frac{1 + \sqrt{1 - 2(1 - P)}}{2} \right) = 0.39$
- $e_2 = Tr(\Sigma) * \left(\frac{1 - \sqrt{1 - 2(1 - P)}}{2} \right) = 0.02$

Παρατηρούμε ότι οι τιμές των ιδιοτιμών στον κβαντικό αλγόριθμο QPCA βγήκαν ίδιες με αυτές στον κλασσικό αλγόριθμο PCA. Αυτό είναι άκρως λογικό, διότι δεν έχουμε διαφορά στο αποτέλεσμα, αλλά στην χρονική πολυπλοκότητα. Δηλαδή, πόσο πιο γρήγορα μπορεί να βρεθεί το αποτέλεσμα στον QPCA από τον PCA. Περισσότερα για τις χρονικές πολυπλοκότητες θα δούμε παρακάτω στην ενότητα 5.4 και στα συμπεράσματα του 6^{ου} κεφαλαίου.

5.4 Σύγκριση κλασσικού(PCA) και κβαντικού(QPCA) αλγορίθμου και κατ' επέκταση κλασσικού και κβαντικού υπολογιστή

Σε αυτή την ενότητα θα συγκρίνουμε τον κλασσικό PCA και τον αντίστοιχό του, τον κβαντικό QPCA και θα καταλήξουμε στη γενικότερη σύγκριση ανάμεσα στον κλασσικό και στον κβαντικό υπολογιστή.

Ο κλασσικός αλγόριθμος PCA βασίζεται στην ιδιοαποσύνθεση του πίνακα συνδιασποράς C , δηλαδή στην εύρεση των ιδιοτιμών και των ιδιοδιανυσμάτων του. Στο συγκεκριμένο σημείο του κλασσικού αλγορίθμου είναι δυνατόν να πραγματοποιηθεί κβαντική επιτάχυνση με σκοπό ο αλγόριθμος να τρέξει πιο γρήγορα σ' έναν κβαντικό υπολογιστή. Η υλοποίηση του κλασσικού αλγορίθμου PCA έχει χρονική πολυπλοκότητα της τάξης $O(N)$. Αντίστοιχα ο κβαντικός αλγόριθμος QPCA έχει χρονική πολυπλοκότητα της τάξης $O(\log N)$.

Αυτό επιτυγχάνεται, κυρίως, λόγω της αναπαράστασης των κλασσικών δεδομένων σαν κβαντικές καταστάσεις. Έτσι μας επιτρέπεται να χρησιμοποιήσουμε την πολύ σημαντική ιδιότητα των qubits, αυτήν της κβαντικής υπέρθεσης και κατ' επέκταση του κβαντικού παραλληλισμού και εναγκαλισμού. Οι ιδιότητες αυτές μας δίνουν τη δυνατότητα να έχουμε τα qubits συνδεδεμένα αναμεταξύ τους και να πραγματοποιούμε ταυτόχρονους υπολογισμούς σε αντίθεση με τους κλασσικούς που κάνουν μια-μια πράξη γραμμικά. Ένα δεύτερο σημείο που επιτυγχάνεται κβαντική επιτάχυνση, αποτελεί η δημιουργία του εκθετικού πίνακα πυκνότητας U , μέσω του μαθηματικού νόμου $U = e^{iH}$, H -ερμιτιανός, που αποτελεί έναν μοναδιαίο πίνακα. Το τρίτο και πιο βασικό σημείο της κβαντικής επιτάχυνσης αποτελεί η κβαντική υπορουτίνα PhaseEstim. Αυτή η υπορουτίνα δέχεται ως είσοδο το μοναδιαίο πίνακα U , που δημιουργήθηκε παραπάνω και υπολογίζει τα ιδιοδιανύσματα και τις ιδιοτιμές του. Πρέπει να σημειωθεί, ότι η κβαντική υπορουτίνα PhaseEstim αποτελεί ένα κβαντικό κύκλωμα, στο οποίο δρουν κβαντικές πύλες πάνω σε qubits κάνοντας τους απαραίτητους μετασχηματισμούς, για να παρθεί το τελικό αποτέλεσμα.

ΣΥΓΚΕΝΤΡΩΤΙΚΑ ΤΑ ΒΗΜΑΤΑ ΤΩΝ ΔΥΟ ΑΛΓΟΡΙΘΜΩΝ	
Κλασσικός PCA	Κβαντικός QPCA
<p>ΒΗΜΑ 1^ο</p> <p>Κεντράρισμα δεδομένων γύρω από το 0 και δημιουργία του πίνακα δεδομένων X</p>	<p>ΒΗΜΑ 1^ο</p> <p>Αναπαράσταση κλασσικών δεδομένων σαν κβαντικές καταστάσεις</p> <p>Κβαντική επιτάχυνση λόγω κβαντικής υπέρθεσης</p>
<p>ΒΗΜΑ 2^ο</p> <p>Υπολογισμός πίνακα συνδιασποράς</p> $C = \frac{1}{n-1} X^T X$	<p>ΒΗΜΑ 2^ο</p> <p>Δημιουργία πίνακα πυκνότητας ρ</p>
<p>ΒΗΜΑ 3^ο</p> <p>Υπολογισμός ιδιοτιμών και ιδιοδιανυσμάτων του C</p> $\det C - \lambda I = 0$ $Cv = \lambda v$	<p>ΒΗΜΑ 3^ο</p> <p>Εκθετικοποίηση πίνακα ρ στον μοναδιαίο $U = e^{-i\rho t}$</p> <p>Κβαντική επιτάχυνση λόγω δημιουργίας του μοναδιαίου πίνακα U</p> <p>Χρονική πολυπλοκότητα της τάξης $O(\log N)$</p>
<p>ΒΗΜΑ 4^ο</p> <p>Κρατάμε τα ιδιοδιανύσματα με τις μεγαλύτερες ιδιοτιμές, που περιέχουν και την περισσότερη πληροφορία και δημιουργούμε τον πίνακα ιδιοδιανυσμάτων</p>	<p>ΒΗΜΑ 4^ο</p> <p>Υπολογισμός ιδιοτιμών και ιδιοδιανυσμάτων του U μέσω της κβαντικής υπορουτίνας PhaseEstim</p> <p>Κβαντική επιτάχυνση</p> <p>Χρονική πολυπλοκότητα της τάξης $O(\log N)$</p>
<p>ΒΗΜΑ 5^ο</p> <p>Τελικά μειωμένα σε όγκο δεδομένα με</p>	<p>ΒΗΜΑ 5^ο</p> <p>Δράση του U στον ρ για την δημιουργία</p>

**ΚΒΑΝΤΙΚΗ ΜΗΧΑΝΙΚΗ ΜΑΘΗΣΗ, ΕΦΑΡΜΟΓΕΣ ΚΑΙ ΥΛΟΠΟΙΗΣΗ ΣΕ ΚΒΑΝΤΙΚΟ
HARDWARE**

την περισσότερη πληροφορία, που μπορούν να χρησιμοποιηθούν σαν είσοδος σε άλλους αλγορίθμους	του τελικού πίνακα πυκνότητας $d=U\rho$
	ΒΗΜΑ 6 ^ο Δειγματοληψία του d
	ΒΗΜΑ 7 ^ο Κύριες συνιστώσες και αποτελέσματα
Συνολική χρονική πολυπλοκότητα της τάξης $O(N)$	Συνολική χρονική πολυπλοκότητα της τάξης $O(\log N)$

Σε αυτό το σημείο θα προχωρήσουμε στη πιο γενική σύγκριση ανάμεσα σε έναν κλασσικό και έναν κβαντικό υπολογιστή. Στον παρακάτω πίνακα θα δούμε συνοπτικά κάποιες διαφορές ανάμεσά τους και στο τέλος θα αναφερθούμε σε κάποια βασικά προβλήματα των κβαντικών υπολογιστών.

ΣΥΓΚΡΙΣΗ ΚΛΑΣΣΙΚΟΥ ΚΑΙ ΚΒΑΝΤΙΚΟΥ ΥΠΟΛΟΓΙΣΤΗ	
ΚΛΑΣΣΙΚΟΣ ΥΠΟΛΟΓΙΣΤΗΣ	ΚΒΑΝΤΙΚΟΣ ΥΠΟΛΟΓΙΣΤΗΣ
<p>Βασική μονάδα το bit</p> <p>Παίρνει τιμές 0 ή 1</p> <p>Αυτό μεταφράζεται στο "δεν περνά ή περνά" ρεύμα από ένα κύκλωμα</p>	<p>Βασική μονάδα το qubit</p> <p>Παίρνει τιμές ταυτόχρονα και 0 και 1</p> <p>Δύο σημαντικές ιδιότητες των qubits είναι η κβαντική υπέρθεση, που μας επιτρέπει να κάνουμε ταυτόχρονους υπολογισμούς και ο κβαντικός εναγκαλισμός που βοηθά τα qubits να είναι συνδεδεμένα αναμεταξύ τους, ώστε να λειτουργούν σε συνεργασία και να παράγουν αποτελέσματα</p>
Για παράδειγμα αν πάρουμε 2 bit τότε μπορούμε να έχουμε κάθε φορά μία από	Λόγω κβαντικής υπέρθεσης μπορούμε να έχουμε και τις 4 καταστάσεις

τις καταστάσεις 00,01,10,11.	ταυτόχρονα: $ \psi\rangle = \alpha 00\rangle + \beta 01\rangle + \gamma 10\rangle + \delta 11\rangle$, με $ \alpha ^2 + \beta ^2 + \gamma ^2 + \delta ^2 = 1$
Η διόρθωση σφαλμάτων γίνεται εύκολα μέχρι τώρα	Η διόρθωση σφαλμάτων γίνεται δύσκολα ακόμα επί του παρόντος λόγω του κβαντικού hardware
Μικρός σε μέγεθος	Μέγεθος δωματίου
Αποτελείται από επεξεργαστή, RAM, κάρτα γραφικών, δίσκους, περιφερειακά κ.ά.	Αποτελείται από έναν επεξεργαστή από qubits και από μια τεράστια ψύκτρα
Λειτουργεί σε συνθήκες περιβάλλοντος	Λειτουργεί στους 0° Kelvin ($-273,15^\circ C$) Ειδικότερα λειτουργούν με 0,1-10 mK , την ώρα που η θερμοκρασία στο κενό του διαστήματος είναι 2,595 K

Τα βασικά προβλήματα των κβαντικών υπολογιστών είναι:

- 1) Μεγάλο ποσοστό σφαλμάτων: Το κάθε qubit έχει για τιμή έναν συνδυασμό πιθανοτήτων και αυτό καθιστά πιο δύσκολη τη διαδικασία εντοπισμού και διόρθωσης των σφαλμάτων.
- 2) Περιορισμοί στη εισαγωγή δεδομένων: Ακόμη δεν υπάρχει η δυνατότητα στο κβαντικό hardware να μετατρέψουμε μεγάλο όγκο κλασσικών δεδομένων σε κβαντικές καταστάσεις, με σκοπό να τα χρησιμοποιήσουμε σαν είσοδο σ' έναν κβαντικό υπολογιστή, δηλαδή δεν μπορεί ακόμη να υπάρξει μεγάλος αριθμός από qubits σε κβαντική υπέρθεση σε έναν κβαντικό επεξεργαστή. Μέχρι τώρα είμαστε στα 72 qubits της Google, στα 53 της IBM και στα 49 της Intel. Για παράδειγμα αν θέλουμε να τρέξουμε έναν αλγόριθμο με 1000 λογικά qubits θα χρειαζόμασταν 100.000 φυσικά qubits, πράγμα που σήμερα ακόμα αυτό δεν είναι εφικτό. Σαν φυσικό qubit εννοούμε μια συσκευή που συμπεριφέρεται ως ένα κβαντικό σύστημα 2-καταστάσεων και χρησιμοποιείται ως ένα κομμάτι ενός υπολογιστικού συστήματος. Σαν λογικό qubit εννοούμε αυτό, που μπορεί να χρησιμοποιηθεί από τις κβαντικές λογικές πύλες, ώστε

να πραγματοποιηθούν μετασχηματισμοί στο κβαντικό κύκλωμα ενός κβαντικού αλγορίθμου (1 λογικό qubit = 100 φυσικά qubits).

3) Μεγάλη ευαισθησία και αστάθεια σαν συστήματα: Δεν είναι δυνατόν να διατηρηθούν σε ικανοποιητικό χρόνο τα qubits σε κατάσταση υπέρθεσης λόγω οποιασδήποτε παρεμβολής από το περιβάλλον π.χ. θερμοκρασία, επιρροές από τα γειτονικά qubits. Καταρρίπτεται έτσι η κατάστασή τους(υπέρθεση), πριν προλάβουμε να εξάγουμε κάποια χρήσιμα συμπεράσματα. Συνεπώς, όσο προσπαθούμε να προσθέσουμε περισσότερα qubits για να κατασκευάσουμε μεγαλύτερους και πιο χρήσιμους επεξεργαστές, τόσο περισσότερο αυξάνεται το ποσοστό των συνολικών σφαλμάτων και αντίστοιχα μειώνεται ο χρόνος διατήρησης των qubits σε κατάσταση υπέρθεσης.

Κλείνοντας την συγκεκριμένη ενότητα, πρέπει να τονιστεί ότι ο χειρισμός ενός κβαντικού υπολογιστή γίνεται μέσω ενός κλασσικού. Δεν παίρνουμε όμως ένα πρόγραμμα(κλασσικό αλγόριθμο) από έναν κλασσικό υπολογιστή και το τρέχουμε σε έναν κβαντικό, ώστε να τρέξει πιο γρήγορα. Το πρόγραμμα πρέπει να σχεδιαστεί από την αρχή, με σκοπό την αλλαγή του αλγορίθμου ή μέρους του, για να μπορεί να τρέξει στον κβαντικό υπολογιστή. Για να τρέξει πιο γρήγορα ένας αλγόριθμος σε έναν κβαντικό υπολογιστή, πρέπει να μπορεί να μειωθεί ο χρόνος εκτέλεσης κάθε βήματός του, να μπορεί να γίνει παράλληλη εκτέλεση πολλών βημάτων του ή να γίνει επανασχεδιασμός του και να μειωθούν τα απαιτούμενα βήματά του.

6 ΣΥΜΠΕΡΑΣΜΑΤΑ

Είδαμε παραπάνω ότι τα αποτελέσματα, δηλαδή οι ιδιοτιμές του παραδείγματος του κλασσικού αλγορίθμου PCA, της προσομοίωσης στη πλατφόρμα Qiskit της IBM του κβαντικού αλγορίθμου QPCA και της υλοποίησης του κβαντικού αλγορίθμου QPCA σε γλώσσα Python, ήταν ίδια. Το γεγονός αυτό είναι λογικό, διότι ο κβαντικός αλγόριθμος QPCA δεν θα έπρεπε να παράγει διαφορετικά αποτελέσματα από τον κλασσικό αλγόριθμο PCA. Σκοπός είναι ο κβαντικός αλγόριθμος να λύσει το πρόβλημα πιο γρήγορα απ' ό,τι θα το έλυνε ο κλασσικός. Με άλλα λόγια, ο κβαντικός να έχει καλύτερη χρονική πολυπλοκότητα από τον κλασσικό.

Ήρθε η στιγμή να συγκρίνουμε τις χρονικές πολυπλοκότητες των PCA και QPCA και να δούμε τι συμβαίνει. Η χρονική πολυπλοκότητα του PCA είναι $O(N)$. Για να δούμε τη χρονική πολυπλοκότητα του QPCA θα το πάμε σταδιακά. Η πολυπλοκότητα για την δημιουργία του εκθετικού πίνακα πυκνότητας ρ είναι $O(\log N)$. Η πολυπλοκότητα της κβαντικής υπορουτίνας PhaseEstim για την εύρεση ιδιοδιανυσμάτων και ιδιοτιμών είναι $O(\log N)$. Η πολυπλοκότητα της δειγματοληψίας της τελικής κατάστασης είναι $O(R \log N)$. Επομένως η συνολική πολυπλοκότητα του QPCA είναι $O(R \log N)$.

Συμπερασματικά βλέπουμε λοιπόν ότι η χρονική πολυπλοκότητα του κβαντικού αλγορίθμου QPCA είναι καλύτερη από αυτήν του κλασσικού PCA. Άρα ο QPCA βρίσκει τη λύση πιο γρήγορα από τον PCA. Το 'κλειδί' της κβαντικής

επιτάχυνσης είναι κυρίως η αναπαράσταση των κλασσικών δεδομένων ως qubits και η κβαντική υπορουτίνα PhaseEstim. Πρέπει να σημειωθεί ότι ο QPCA είναι αποδοτικός μόνο στην περίπτωση, που ένας μικρός αριθμός από κύριες συνιστώσες(PC's) απαιτούνται για να εξηγήσουν τη διασπορά των δεδομένων, δηλαδή $R \ll N$. Αλλιώς οι χρονικές πολυπλοκότητες και των δύο αλγορίθμων PCA και QPCA είναι $O(N)$.

6.1 Μελλοντική Δουλειά

Με βάση την παρούσα διπλωματική εργασία, υπάρχει η δυνατότητα να πραγματοποιηθούν διάφορα πράγματα μελλοντικά.

Το πρώτο πράγμα, που μπορεί να πραγματοποιηθεί, είναι ότι μπορεί να δημιουργηθεί οποιοδήποτε κβαντικό κύκλωμα, οποιουδήποτε κβαντικού αλγορίθμου θέλει να υλοποιήσει κάποιος. Η συγκεκριμένη διπλωματική εργασία περιλαμβάνει το αρχείο Quantum_Library.py, με δικό μου κώδικα σε γλώσσα Python, το οποίο αποτελεί τη βάση της κβαντικής μου βιβλιοθήκης. Αν δημιουργηθεί ένα νέο αρχείο python.py, εισάγοντας όποια κλάση ή συνάρτηση χρειάζεται από το αρχείο Quantum_Library.py, μπορούν να δημιουργηθούν οι κβαντικές πύλες που χρειάζονται για να υλοποιηθεί το κβαντικό κύκλωμα του κβαντικού αλγορίθμου που θέλει κάποιος να δημιουργήσει. Αφού έχει δημιουργηθεί το κβαντικό κύκλωμα, μέσω της εισαγωγής των συναρτήσεων μέτρησης που έχω υλοποιήσει στο αρχείο Quantum_Library.py, είναι εφικτό να πραγματοποιηθεί μέτρηση και να παρθεί αποτέλεσμα.

Το δεύτερο πράγμα, που μπορεί να γίνει μελλοντικά, είναι η υλοποίηση διαφορετικών παραδειγμάτων του PCA και του QPCA με μεγαλύτερο όγκο δεδομένων. Με άλλα λόγια, δεδομένα περισσότερων διανυσμάτων ή δεδομένα μεγαλύτερων διαστάσεων και αντίστοιχα περισσότερων από 5 qubits εισόδου. Για να κατανοήσουμε καλύτερα το τελευταίο πρέπει να θυμηθούμε κάτι που αναφέρθηκε στην αρχή του 5^{ου} κεφαλαίου. Αυτό που γράφεται εκεί, είναι ότι, όταν έχουμε αρχικά δεδομένα 2 διανυσμάτων, αυτά αντιστοιχούν σε 5 qubits εισόδου του κβαντικού κυκλώματος. Αυτό σημαίνει, ότι αν έχουμε 3 διανύσματα, αυτά αντιστοιχούν σε 7 qubits, τα 4 διανύσματα σε 9 qubits κ.ο.κ. Κοινώς γίνεται κατανοητό ότι όταν έχουμε αρχικά δεδομένα n διανυσμάτων, αυτά αντιστοιχούν σε $2n+1$ qubits εισόδου στο κβαντικό κύκλωμα του κβαντικού αλγορίθμου της Ανάλυσης Κύριων Συνιστωσών (QPCA).

7 ΒΙΒΛΙΟΓΡΑΦΙΑ

Βιβλία:

- [B1] Quantum Computing Explained,
David McMahon,
Published by John Wiley & Sons, Inc., Copyright 2008
- [B2] Quantum Computation and Quantum Information,
Michael A. Nielsen and Isaac L. Chuang,
10th Anniversary edition, published 2010, printed in the United Kingdom at the
University Press, Cambridge
- [B3] Quantum Mechanics: The Theoretical Minimum,
Leonard Susskind and Art Friedman,
Copyright 2014, published by Basic Books, A Member of the Perseus Books
Group
- [B4] Κβαντομηχανική II: Θεμελιώδεις αρχές και μέθοδοι – Κβαντικοί
Υπολογιστές,
Στέφανος Λ. Τραχανάς,
Πανεπιστημιακές Εκδόσεις Κρήτης, Έκδοση 2016
- [B5] Σημειώσεις μαθήματος Κβαντικής Τεχνολογίας του τμήματος HMMY-
Πολυτεχνείο Κρήτης, Καθηγητής Δημήτριος Αγγελάκης

Δημοσιεύσεις:

- [Δ1] Basic concepts in quantum computation,
Artur Ekert, Patric Hayden and Hitoshi Inamori,

Center for Quantum Computation, University of Oxford, Oxford OX13PU,
United Kingdom 16 January 2000

- [Δ2] Quantum Principal Component Analysis,
Seth Lloyd (Massachusetts Institute of Technology, department of Mechanical Engineering and Research Laboratory for Electronics)
Masoud Mohseni (Google Research)
Patrick Rebentrost (MIT, Research Laboratory for Electronics)
arXiv: 1307.0401v2[quant-ph] 16 Sep 2013
- [Δ3] Quantum optical Implementation of quantum information processing,
J.I.Cirak, Luming Duan and P.Zoller,
Institute for theoretical Physics, University of Innsbruck Technikerstrasse 25-2,
A-6020 Innsbruck, Austria (Dated: 2001 10 17)
- [Δ4] Quantum Computing – Lecture Notes,
Mark Oskin,
Department of Computer Science and Engineering, University of Washington
(Spring 2002) ‘Lecture notes are based on the book Quantum Computation and Quantum Information by Michael A. Nielsen and Isaac L. Chuang’
- [Δ5] Quantum Computing,
Abbas Edalat,
Department of Computing, Imperial College London, 18 lectures + 9 tutorials,
Lecture Notes and Exercise Sheets available from: <http://www.doc.ic.ac.uk/~ae>
- [Δ6] Quantum Machine Learning for data scientists,
Dawid Kopczyk,
Quantee Limited, Manchester, United Kingdom
arXiv: 1804.10068v1[quant-ph] 25 Apr 2018
- [Δ7] Quantum Algorithm Implementations for Beginners,
Patrick J. Coles, Stephan Eidenbenz, Scott Pakin, Adetokunbo Adedoyin, John Ambrosiano, Petr Anisimov, William Casper, Gopinath Chennupati, Carleton Coffrin, Hristo Djidjev, David Gunter, Satish Karra, Nathan Lemons, Shizeng Lin, Andrey Lokhov, Alexander Malyzhenkov, David Mascarenas, Susan Mniszewski, Balu Nadiga, Dan O’Malley, Diane Oyen, Lakshman Prasad, Randy Roberts, Phil Romero, Nandakishore Santhi, Nikolai Sinitsyn, Pieter Swart, Marc Vuffray, Jim Wendelberger, Boram Yoon, Richard Zamora, and Wei Zhu
Los Alamos National Laboratory, Los Alamos, New Mexico, USA
arXiv: 1804.03719v1[cs.ET] 10 Apr 2018
- [Δ8] Types of Machine Learning Algorithms,
Taiwo Oladipupo Ayodele,
University of Portsmouth, Published: February 1st 2010
- [Δ9] Principal Component Analysis from Wikipedia
- [Δ10] The classical and quantum Fourier transform,
Ronald de Wolf, February 22, 2011

ΚΕΦΑΛΑΙΟ 7: ΒΙΒΛΙΟΓΡΑΦΙΑ

8 ΠΑΡΑΡΤΗΜΑΤΑ

ΠΑΡΑΡΤΗΜΑ 1: ΚΩΔΙΚΕΣ ΣΕ ΓΛΩΣΣΑ ΡΥΘΜΟΝ	81
ΠΑΡΑΡΤΗΜΑ 2: ΔΗΜΙΟΥΡΓΙΑ ΠΡΟΣΟΜΟΙΩΤΗ ΓΙΑ ΑΝΑΛΥΣΗ ΤΟΥ ΚΒΑΝΤΙΚΟΥ ΚΥΚΛΩΜΑΤΟΣ ΔΗΜΙΟΥΡΓΙΑΣ BELL ΚΑΤΑΣΤΑΣΕΩΝ	102
ΠΑΡΑΡΤΗΜΑ 3: ΚΩΔΙΚΑΣ ΓΙΑ ΤΗΝ ΠΛΑΤΦΟΡΜΑ QISKIT ΤΗΣ IBM	105
ΠΑΡΑΡΤΗΜΑ 4: ΤΑΝΥΣΤΙΚΟ ΓΙΝΟΜΕΝΟ ΠΙΝΑΚΩΝ	107

ΠΑΡΑΡΤΗΜΑ 1: ΚΩΔΙΚΕΣ ΣΕ ΓΛΩΣΣΑ PYTHON

Σε αυτό το παράρτημα περιέχονται οι κώδικες μου σε γλώσσα προγραμματισμού Python:

1. **PCA_Initial_Example_In_Presentation.py**: Το απλό παράδειγμα του PCA για την καλύτερη κατανόηση των βημάτων του αλγορίθμου στο 3^ο κεφάλαιο και συγκεκριμένα στην ενότητα 3.3 με 2 διανύσματα των 10 στοιχείων το καθένα.
2. **PCA.py**: Για την υλοποίηση του PCA στο 5^ο κεφάλαιο και συγκεκριμένα στην ενότητα 5.1 με αρχικά δεδομένα αριθμού δωματίων και τετραγωνικά μέτρα δωματίων διαφόρων σπιτιών, δηλαδή 2 διανυσμάτων των 15 στοιχείων το καθένα.
3. **Quantum_Library.py**: Για την υλοποίηση της κβαντικής μου βιβλιοθήκης, την οποία χρησιμοποιούν για να τρέξουν τα αρχεία `Bell_States_Demonstration.py` και `QPCA.py`. Τα αρχεία `Bell_States_Demonstration.py` και `QPCA.py` πρέπει να βρίσκονται στον ίδιο φάκελο με το αρχείο `Quantum_Library.py`.
4. **Bell_States_Demonstration.py**: Για την υλοποίηση του κβαντικού κυκλώματος της δημιουργίας των Bell States στο Παράρτημα 2.
5. **QPCA.py**: Για την υλοποίηση του QPCA στο 5^ο κεφάλαιο και συγκεκριμένα στην ενότητα 5.3 με αρχικά δεδομένα αριθμού δωματίων και τετραγωνικά μέτρα δωματίων διαφόρων σπιτιών, δηλαδή 2 διανυσμάτων των 15 στοιχείων το καθένα.

1. PCA_Initial_Example_In_Presentation.py:

```
1  import numpy as np
2
3  import matplotlib
4  matplotlib.use('TkAgg')
5  import matplotlib.pyplot as plt
6
7  ##### Origin Data
8  # x-axis values
9  x1 = [1,2,3,4,5,6,7,8,9,10]
10 # y-axis values
11 x2 = [2,4,5,7,6,8,9,11,12,12]
12
13 print("Origin Data\n")
14 print("x1 is :", x1)
15 print("x2 is :", x2)
16
17 plt.figure(1)
18 # plotting points as a scatter plot
19 plt.scatter(x1, x2, label= "stars", color= "red",
20            marker= "*", s=30)
21
22 # x-axis label
23 plt.xlabel('x1 - axis')
24 # frequency label
25 plt.ylabel('x2 - axis')
26 # plot title
27 plt.title('Origin Data!')
28 # showing legend
29 plt.legend()
30
31
```

```
33 ##### Calculate the Mean(meanx1,meanx2)
34
35 import statistics
36
37 meanx1 = statistics.mean(x1)
38 meanx2 = statistics.mean(x2)
39
40 print("\nmeanx1 is :", meanx1) # 5.5
41 print("meanx2 is :", meanx2) # 7.6
42
43 ##### Mean in the center(0,0) -- xx1=x1-meanx1 for each x1 and xx2=x2-meanx2 for each x2
44
45 xx1 = [-4.5,-3.5,-2.5,-1.5,-0.5,0.5,1.5,2.5,3.5,4.5]
46
47 xx2 = [-5.6,-3.6,-2.6,-0.6,-1.6,0.4,1.4,3.4,4.4,4.4]
48
49 print("\nCentered Data\n")
50 print("xx1 is :", xx1)
51 print("xx2 is :", xx2)
52
53 plt.figure(2)
54 # plotting points as a scatter plot
55 plt.scatter(xx1, xx2, label= "stars", color= "red",
56            marker= "*", s=30)
57
58 # x-axis label
59 plt.xlabel('xx1 - axis')
60 # frequency label
61 plt.ylabel('xx2 - axis')
62 # plot title
63 plt.title('Centered Data!')
64 # showing legend
65 plt.legend()
66
```

```

69 ##### X-matrix
70 X = np.stack((xx1,xx2),axis=0).T
71 print("\nX-matrix:\n",X)
72 Xraw = X.T
73 print("\nXraw-matrix:\n",Xraw)
74
75 ##### Calculate covariance matrix C = 1/N-1(X'X) -- N=10
76
77 XrawX = Xraw.dot(X)
78 print("\nXrawX-matrix:\n",XrawX)
79 C = 1/9 * Xraw.dot(X)
80 print("\nCovariance-matrix(C):\n",C)
81
82 ##### EigenVectors and EigenValues from Covariance Matrix C
83 eig_vals, eig_vecs = np.linalg.eig(C)
84 print('\nEigenVectors \n%s' %eig_vecs)
85 print('\nEigenvalues \n%s' %eig_vals)
86
87 ##### We calculate which eigenvalue has the most information
88 e1 = eig_vals[0] / sum(eig_vals) #0,8%
89 e2 = eig_vals[1] / sum(eig_vals) #99,1%
90
91 print("\nWe calculate which eigenvalue has the most information!")
92 print("\nPercentage of first eigenvalue:",e1)
93 print("\nPercentage of second eigenvalue:",e2)
94
95 ##### EigenVectors Matrix V
96 V = np.array([eig_vecs[1]]).T
97 print("\nEigenVectors-matrix(V):\n",V)
98

```

```

98
99 ##### FinalData Y = XV
100 Y = X.dot(V)
101 print("\nFinal Data Y = XV:\n",Y)
102
103 # plotting points as a scatter plot
104 y = [0,0,0,0,0,0,0,0,0,0]
105
106 plt.figure(3)
107 plt.scatter(Y, y, label= "stars", color= "red",
108             marker= "*", s=30)
109
110 # x-axis label
111 plt.xlabel('PC1 - axis')
112 # frequency label
113 plt.ylabel('y - axis')
114 # plot title
115 plt.title('Final Data!')
116 # showing legend
117 plt.legend()
118
119 # function to show the plot
120 plt.show()
121
122
123

```

2. PCA.py:

```

1  import numpy as np
2  import statistics
3  import matplotlib
4  matplotlib.use('TkAgg')
5  import matplotlib.pyplot as plt
6
7  ##### Origin Data
8
9  # x-axis values (Number of bedrooms)
10 x1 = [2,1.5,2,2,1.5,1.5,1.5,1.5,2,2,2,2.5,2,1.5,2]
11 # y-axis values (Square footage)
12 x2 = [1.514,0.6825,1.363,1.269,0.659,0.8465,0.706,0.816,1.4375,1.782,2.206,2.222,2.139,1.532,1.9285]
13
14 print("\nNUMBER OF BEDROOMS(Origin Data):",x1)
15 print("\nSQUARE FOOTAGE(Origin Data):",x2)
16
17
18 plt.figure(1)
19 # plotting points as a scatter plot
20 plt.scatter(x1, x2, label= "stars", color= "red",
21            marker= "*", s=30)
22 # x-axis label
23 plt.xlabel('NUMBER OF BEDROOMS')
24 # frequency label
25 plt.ylabel('SQUARE FOOTAGE')
26 # plot title
27 plt.title('ORIGIN DATA')
28 # showing legend
29 plt.legend()
30
31

```

```

33 ##### Calculate the Mean(meanx1,meanx2)
34
35 meanx1 = statistics.mean(x1)
36 meanx2 = statistics.mean(x2)
37
38 print("\nThe mean of x1 origin data is :", meanx1) # 1.8
39 print("\nThe mean of x2 origin data is :", meanx2) # 1.406
40
41 ##### Mean in the center(0,0) -- xx1=x1-meanx1 for each x1 and xx2=x2-meanx2 for each x2
42
43 xx1 = [0.2,-0.3,0.2,0.2,-0.3,-0.3,-0.3,-0.3,0.2,0.2,0.2,0.7,0.2,-0.3,0.2]
44
45 xx2 =
46     [0.1075,-0.724,-0.0435,-0.1375,-0.7475,-0.56,-0.7005,-0.5905,0.031,0.3755,0.7995,0.8155,0.7325,0.1255,0
47     .522]
48
49
50 print("\nNUMBER OF BEDROOMS(Centered Data):",xx1)
51 print("\nSQUARE FOOTAGE(Centered Data):",xx2)
52
53
54 plt.figure(2)
55 # plotting points as a scatter plot
56 plt.scatter(xx1, xx2, label= "stars", color= "red",
57            marker= "*", s=30)
58 # x-axis label
59 plt.xlabel('NUMBER OF BEDROOMS')
60 # frequency label
61 plt.ylabel('SQUARE FOOTAGE')
62 # plot title
63 plt.title('CENTERED DATA')
64 # showing legend
65 plt.legend()
66
67

```

ΚΕΦΑΛΑΙΟ 8: ΠΑΡΑΡΤΗΜΑΤΑ

```

65
66 ##### X-matrix - Einai o pinakas opou h 1 stlh toy einai ta stoixeia NUMBER OF BEDROOMS(Centered Data)
67 ##### kai h 2 stlh toy ta stoixeia SQUARE FOOTAGE(Centered Data)
68 X = np.stack((xx1,xx2),axis=0).T
69 print("\nMatrix X\n1column --> NUMBER OF BEDROOMS(Centered Data)\n2column --> SQUARE FOOTAGE(Centered
    Data)\n X=\n",X)
70 Xraw = X.T
71 print("\nMatrix Xraw --> Anastrofos toy X\nXraw=\n",Xraw)
72
73 ##### Calculate variacne-covariance matrix C = 1/N-1(X'X) -- N=15
74
75 XrawX = Xraw.dot(X)
76 print("\nMultiply Xraw with X\nXrawX=\n",XrawX)
77 C = 1/14 * Xraw.dot(X)
78 print("\nCOVARIANCE MATRIX\nC=\n",C)
79
80 ##### EigenVectors and EigenValues from Covariance Matrix C
81 eig_vals, eig_vecs = np.linalg.eig(C)
82 print('\nEigenVectors of C \n%s' %eig_vecs)
83 print('\nEigenValues of C \n%s' %eig_vals)
84
85 ##### We calculate which eigenvalue has the most information
86 eigenvalue_0 = eig_vals[0] / sum(eig_vals) #6,5%
87 eigenvalue_1 = eig_vals[1] / sum(eig_vals) #93,5%
88 print("\nCalculate which eigenvalue has the most information")
89 print("eigenvalue_0=\n",eigenvalue_0)
90 print("neigenvalue_1=\n",eigenvalue_1)
91
92 ##### EigenVectors Matrix V (periexei to eigenvector ths eigenvalue_1)
93 V = np.array([eig_vecs[1]]).T
94 print("\nEigenVectors Matrix --> eigenvector of eigenvalue_1\nV=\n",V)
95

```

```

96 ##### FinalData Y = XV
97 Y = X.dot(V)
98 print("\nFinalData Y = XV\nY=\n",Y)
99
100
101 plt.figure(3)
102 # plotting points as a scatter plot
103 y = [0,0,0,0,0,0,0,0,0,0,0,0,0,0,0]
104 plt.scatter(Y, y, label= "stars", color= "red",
105             marker= "*", s=30)
106 # x-axis label
107 plt.xlabel('PC1 - axis')
108 # frequency label
109 plt.ylabel('y - axis')
110 # plot title
111 plt.title('FINAL DATA')
112 # showing legend
113 plt.legend()
114
115 # function to show the plot
116 plt.show()
117

```

```

118
119 fig, axs = plt.subplots(2, 2)
120
121 axs[0, 0].scatter(x1, x2, label= "stars", color= "red",marker= "*", s=30)
122 axs[0, 0].set_ylabel='SQUARE FOOTAGE')
123 axs[0, 0].set_title('ORIGIN DATA')
124
125 axs[0, 1].scatter(xx1, xx2, label= "stars", color= "red",marker= "*", s=30)
126 axs[0, 1].set_xlabel='NUMBER OF BEDROOMS',ylabel='SQUARE FOOTAGE')
127 axs[0, 1].set_title('CENTERED DATA')
128
129 axs[1, 0].scatter(Y, y, label= "stars", color= "red",marker= "*", s=30)
130 axs[1, 0].set_xlabel='PC1 - axis',ylabel='y - axis')
131 axs[1, 0].set_title('FINAL DATA')
132
133 plt.show()
134

```

3. Quantum_Library.py:

Το αρχείο Quantum_Library.py, αποτελεί την βάση του κώδικα μου σε γλώσσα Python, για την υλοποίηση της κβαντικής βιβλιοθήκης. Τα αρχεία Bell_States_Demonstration.py, QPCA.py, το χρησιμοποιούν για να ‘τρέξουν’. Για αυτό και το αρχείο Quantum_Library.py, πρέπει να βρίσκεται στον ίδιο φάκελο με αυτά.

Εισαγωγή Βιβλιοθηκών:

```

4
5 import numpy as np
6 import random
7

```

Κλάση για τη δημιουργία μιας κβαντικής κατάστασης:

```

13
14 #####
15 # A class in python that creates a Quantum State
16 # A class is like a "blueprint" for creating objects
17 #####
18
19 # All classes have a function called __init__(), which is always executed when the class is being initiated
20 # __init__() function --> to assign values to object properties that are necessary to do when the object is being
   created
21 class create_Qstate:
22     def __init__(self, vector):
23         length = np.linalg.norm(vector)
24         if not abs(1 - length) < 0.00001:
25             raise ValueError('Error! Quantum states must be unit length!')
26         self.vector = np.array(vector)
27

```


Συνάρτηση που ελέγχει αν ένας τελεστής είναι μοναδιαίος:

```
7
8 # A function that checks if the operator is Unitary:  $U \cdot U^\dagger = U^\dagger \cdot U = I$ 
9 def check_if_unitary(U_matrix):
10     U_matrix_dagger = np.transpose(U_matrix).conjugate()
11     identity_matrix = np.eye(len(U_matrix))
12     return np.allclose(identity_matrix, np.matmul(U_matrix_dagger, U_matrix))
13
```

Κλάση για τη δημιουργία ενός κβαντικού τελεστή:

```
5 #####
6 # A class in python that creates a Quantum Operator
7 #####
8 class create_Qoperator:
9     def __init__(self, matrix):
10         if not check_if_unitary(matrix):
11             raise ValueError('Error! Quantum operators must be unitary!')
12         self.matrix = matrix
13
```

Συναρτήσεις στην κλάση create_Qstate:

Συνάρτηση που υπολογίζει τον αναστροφosuζυγή μιας κατάστασης:

```
33 # A function that calculates the dagger(transpose+conjugate) of the state
34 def dagger_state(state):
35     state_dagger = np.transpose(state.vector).conjugate()
36     return create_Qstate(state_dagger)
37
```

Συνάρτηση που υπολογίζει τον ανάστροφο μιας κατάστασης:

```
27
28 # A function that transposes the state
29 def transpose_state(state):
30     stateT = np.transpose(state.vector)
31     return create_Qstate(stateT)
32
```

Συνάρτηση που υπολογίζει το τανυστικό γινόμενο μεταξύ 2 κβαντικών καταστάσεων:

```
4 # A function that applies the tensor product between two Quantum States
5 def tensor_product_state(self, state):
6     new_vector = np.kron(self.vector, state.vector)
7     return create_Qstate(new_vector)
8
```

Συνάρτηση για την εκτύπωση μιας κβαντικής κατάστασης:

```
79
80 # A function for Quantum State representation
81 def __repr__(self):
82     return '\n<The Quantum State that is created is:\n {}>'.format(' '.join(map(str, self.vector)))
83
```

Συνάρτηση που μετράει κβαντικές καταστάσεις ενός qubit: (4096 επαναλήψεις)

```
37
38 # A function that measures 1-qubit Quantum States
39 def measurement_1_qubit(self,jobs=4096):
40     count_0 = 0
41     count_1 = 1
42
43     for i in range(0,jobs-6):
44         choices = range(len(self.vector))
45         weights = [abs(a)**2 for a in self.vector]
46         result = random.choices(choices, weights)[0]
47         if result==0:
48             count_0 +=1
49         elif result==1:
50             count_1 +=1
51
52     return (count_0/jobs)*100,(count_1/jobs)*100
53
```

Συνάρτηση που μετράει κβαντικές καταστάσεις 2 qubits: (4096 επαναλήψεις)

```
53
54 # A function that measures 2-qubit Quantum States
55 def measurement_2_qubit(self,jobs=4096):
56     count_0 = 0
57     count_1 = 1
58     count_2 = 2
59     count_3 = 3
60
61     for i in range(0,jobs-6):
62         choices = range(len(self.vector))
63         weights = [abs(a)**2 for a in self.vector]
64         result = random.choices(choices, weights)[0]
65         if result==0:
66             count_0 +=1
67         elif result==1:
68             count_1 +=1
69         elif result==2:
70             count_2 +=1
71         elif result==3:
72             count_3 +=1
73
74     return
75         (count_0/jobs)*100,(count_1/jobs)*100,(count_2/jobs)*100,
76         (count_3/jobs)*100
```

Συνάρτηση που μετράει κβαντικές καταστάσεις 3ων qubits: (4096 επαναλήψεις)

```

75
76 # A function that measures 3-qubit Quantum States
77 def measurement_3_qubit(self,jobs=4096):
78     count_0 = 0
79     count_1 = 1
80     count_2 = 2
81     count_3 = 3
82     count_4 = 4
83     count_5 = 5
84     count_6 = 6
85     count_7 = 7
86
87     for i in range(0,jobs-6):
88         choices = range(len(self.vector))
89         weights = [abs(a)**2 for a in self.vector]
90         result = random.choices(choices, weights)[0]
91         if result==0:
92             count_0 +=1
93         elif result==1:
94             count_1 +=1
95         elif result==2:
96             count_2 +=1
97         elif result==3:
98             count_3 +=1
99         elif result==4:
100             count_4 +=1
101         elif result==5:
102             count_5 +=1
103         elif result==6:
104             count_6 +=1
105         elif result==7:
106             count_7 +=1
107
108     return
        (count_0/jobs)*100,(count_1/jobs)*100,(count_2/jobs)*100,
        (count_3/jobs)*100,(count_4/jobs)*100,(count_5/jobs)*100,
        (count_6/jobs)*100,(count_7/jobs)*100

```

Συνάρτηση που μετράει κβαντικές καταστάσεις 4ων qubits: (4096 επαναλήψεις – ίδια λογική με παραπάνω)

```

109
110 # A function that measures 4-qubit Quantum States
111 def measurement_4_qubit(self,jobs=4096):
112     count_0 = 0

```

Συνάρτηση που μετράει κβαντικές καταστάσεις 5 qubits: (4096 επαναλήψεις – ίδια λογική με παραπάνω)

```

167
168 # A function that measures 5-qubit Quantum States
169 def measurement_5_qubit(self,jobs=4096):
170     count_0 = 0

```

Συναρτήσεις στην κλάση create_Qoperator:

Συνάρτηση που επενεργεί έναν κβαντικό τελεστή σε μια κβαντική κατάσταση:

```
94 # A matrix_vector multiplication function
95 def apply_operator(self, state):
96     new_vector = np.matmul(self.matrix, state.vector)
97     return create_Qstate(new_vector)
98
```

Συνάρτηση που υπολογίζει το τανυστικό γινόμενο ανάμεσα σε 2 κβαντικούς τελεστές:

```
99 # A function that applies the tensor product between two Quantum Operators
100 def tensor_product_operator(self, operation):
101     new_matrix = np.kron(self.matrix, operation.matrix)
102     return create_Qoperator(new_matrix)
103
```

Συνάρτηση για την εκτύπωση ενός κβαντικού τελεστή:

```
104 # A function for Quantum Operator representation
105 def __repr__(self):
106     return '\n<The Quantum Operator that is created is:\n {}>'.format(str(self.matrix))
107
```

Η δημιουργία των κβαντικών καταστάσεων ή των κβαντικών τελεστών και οι κλήσεις των συναρτήσεων τους πρέπει να πραγματοποιηθούν σε ένα άλλο αρχείο python name_of_file.py, το οποίο πρέπει να είναι στον ίδιο φάκελο με το Quantum_Library.py.

Το αρχείο name_of_file.py πρέπει να έχει τις παρακάτω εισαγωγές βιβλιοθηκών:

```
1 import numpy as np
2 import math
3 import cmath
4 import random
5 import matplotlib
6 import matplotlib.pyplot as plt
7
8 from Quantum_Library import create_Qstate, create_Qoperator
```

Το αρχείο name_of_file.py πρέπει να έχει τις παρακάτω συναρτήσεις:

Συνάρτηση που υπολογίζει το τανυστικό γινόμενο ανάμεσα σε πολλαπλές καταστάσεις:
(χρησιμοποιεί την συνάρτηση tensor_product_state της κλάσης create_Qstate)

```

9
10 # A function that produces tensor product between multiple states
11 def multi_tensor_states(*args):
12     value = create_Qstate([[1.0]])
13     for i in args:
14         value = value.tensor_product_state(i)
15     return value
16

```

Συνάρτηση που υπολογίζει το τανυστικό γινόμενο ανάμεσα σε πολλαπλούς τελεστές:
(χρησιμοποιεί την συνάρτηση tensor_product_operator της κλάσης create_Qoperator)

```

17 # A function that produces tensor product between multiple operators
18 def multi_tensor_operators(*args):
19     value = create_Qoperator([[1.0]])
20     for i in args:
21         value = value.tensor_product_operator(i)
22     return value
23

```

Μερικά παραδείγματα για το πως μπορούμε να δημιουργήσουμε κβαντικές καταστάσεις ή κβαντικούς τελεστές και πως καλούμε τις συναρτήσεις: (αυτά είναι γραμμένα στο αρχείο name_of_file.py)

κβαντικές καταστάσεις(quantum states):

```

27
28 # State |0>
29 zero = create_Qstate([1, 0])
30
31 # State |1>
32 one = create_Qstate([0, 1])
33

```

κβαντικοί τελεστές(quantum operators):

```

79 # Hadamard Gate
80 H = create_Qoperator([
81
82     [1/2**0.5, 1/2**0.5],
83     [1/2**0.5, -1/2**0.5],
84 ])
85

```

```
310
311 # Toffoli-CCNOT Gate
312 Toffoli = create_Qoperator([
313
314         [1,0,0,0,0,0,0,0],
315         [0,1,0,0,0,0,0,0],
316         [0,0,1,0,0,0,0,0],
317         [0,0,0,1,0,0,0,0],
318         [0,0,0,0,1,0,0,0],
319         [0,0,0,0,0,1,0,0],
320         [0,0,0,0,0,0,1,0],
321         [0,0,0,0,0,0,0,1],
322 ])
323
```

Συναρτήσεις(παραδείγματα κλήσεων):

1. **multi_tensor_states:**

Generally →

stateT = multi_tensor_states(state[1],state[2],state[3],...,state[n])[raw vector]

state = stateT.transpose_state() [column vector]

$|00\rangle \rightarrow$ zero2T = multi_tensor_states(zero,zero) [raw vector]

zero2 = zero2T.transpose_state() [column vector]

$|000\rangle \rightarrow$ zero3T = multi_tensor_states(zero,zero,zero) [raw vector]

zero3 = zero3T.transpose_state() [column vector]

$|0000\rangle \rightarrow$ zero4T = multi_tensor_states(zero,zero,zero,zero) [raw vector]

zero4 = zero4T.transpose_state() [column vector]

$|00000\rangle \rightarrow$ zero5T = multi_tensor_states(zero,zero,zero,zero,zero) [raw vector]

zero5 = zero5T.transpose_state() [column vector]

2. multi_tensor_operators:

Generally→

`operator = multi_tensor_operators(operator[1],operator[2],operator[3],...,operator[n])`

`H*H → H_H = multi_tensor_operators(H,H)`

`H*H*H → H_H_H = multi_tensor_operators(H,H,H)`

Συναρτήσεις κβαντικών καταστάσεων:

1.**dagger_state** → `psi_state_dagger = psi_state.dagger_state()`

2.**tranpose_state** → `psi_state_transpose = psi_state.transpose_state()`

3.**__repr__** → `prin = psi_state.__repr__()`

`print(prin)`

4.**measurement_1_qubit** → `measure_1 = 1_qubit_state.measurement_1_qubit()`

5. **measurement_2_qubit** → `measure_2 = 2_qubit_state.measurement_2_qubit()`

6. **measurement_3_qubit** → `measure_3 = 3_qubit_state.measurement_3_qubit()`

7. **measurement_4_qubit** → `measure_4 = 4_qubit_state.measurement_4_qubit()`

8. **measurement_5_qubit** → `measure_5 = 5_qubit_state.measurement_5_qubit()`

Συναρτήσεις κβαντικών τελεστών:

1.**apply_operator**: `(H*H*H)|000> → psi_state = H_H_H.apply_operator(zero3)`

2.**__repr__** → `prin = operator.__repr__()`

`print(prin)`

4. Bell_States_Demonstration.py:

```

1  import numpy as np
2  import math
3  import cmath
4  import random
5  import matplotlib
6  import matplotlib.pyplot as plt
7
8  from Quantum_Library import create_Qstate, create_Qoperator
9
10 # A function that produces tensor product between multiple states
11 def multi_tensor_states(*args):
12     value = create_Qstate([[1.0]])
13     for i in args:
14         value = value.tensor_product_state(i)
15     return value
16
17 # A function that produces tensor product between multiple operators
18 def multi_tensor_operators(*args):
19     value = create_Qoperator([[1.0]])
20     for i in args:
21         value = value.tensor_product_operator(i)
22     return value
23
24
25 #####
26 # Quantum States #
27 #####
28
29 # State |0>
30 zero = create_Qstate([1, 0])
31
32 # State |1>
33 one = create_Qstate([0, 1])
34
35 #####
36 # Quantum Gates #
37 #####
38
39 #####
40 # 1-Qubit Gates # 2x2
41 #####
42
43 # Identity Gate
44 I = create_Qoperator([
45     [1,0],
46     [0,1],
47 ])
48
49
50 # Hadamard Gate
51 H = create_Qoperator([
52     [1/2**0.5, 1/2**0.5],
53     [1/2**0.5, -1/2**0.5],
54 ])
55
56
57 #####
58 # 2-Qubit Gates # 4x4
59 #####
60
61 # CNOT-CX Gate
62 CNOT = create_Qoperator([
63     [1,0,0,0],
64     [0,1,0,0],
65     [0,0,0,1],
66     [0,0,1,0],
67 ])
68
69
70

```



```

74
75 print("\nImplementation of Bell State b00 = (1/sqrt(2))(|00>+|11>)\n")
76
77 zero2T = multi_tensor_states(zero,zero) #dianisma grammis
78 zero2 = zero2T.transpose_state() #dianisma sthlhs
79 prin1 = zero2.__repr__()
80 print("\nThe tensor product between |0> and |0> is the Quantum State:\n",prin1)
81
82 H_I = multi_tensor_operators(H,I)
83 prin2 = H_I.__repr__()
84 print("\nThe tensor product between Hadamard and Identity Gate is the Quantum
      Operator:\n",prin2)
85
86 psi = H_I.apply_operator(zero2)
87 prin3 = psi.__repr__()
88 print("\nThe (H_I)|00> is the Quantum State:\n",prin3)
89
90 b00 = CNOT.apply_operator(psi)
91 prin4 = b00.__repr__()
92 print("\nThe Bell State b00 is the Quantum State:\n",prin4)
93
94 measure_1 = b00.measurement_2_qubit()
95 print("\nThe measurement of b00 ball state is:\n",measure_1)
96
97 names = ['|00>', '|01>', '|10>', '|11>']
98
99 plt.figure(1)
100 plt.bar(names,measure_1,width=0.2,color='red')
101 # x-axis label
102 plt.xlabel('Quantum States')
103 # y-axis label
104 plt.ylabel('Probability Distribution %')
105 # plot title
106 plt.title('Bell State b00')
107

```

```

109
110 print("\nImplementation of Bell State b01 = (1/sqrt(2))(|01>+|10>)\n")
111
112 zero_oneT = multi_tensor_states(zero,one) #dianisma grammis
113 zero_one = zero_oneT.transpose_state() #dianisma sthlhs
114 prin5 = zero_one.__repr__()
115 print("\nThe tensor product between |0> and |1> is the Quantum State:\n",prin5)
116
117 print("\nThe tensor product between Hadamard and Identity Gate is the Quantum
      Operator:\n",prin2)
118
119 psi_1 = H_I.apply_operator(zero_one)
120 prin6 = psi_1.__repr__()
121 print("\nThe (H_I)|01> is the Quantum State:\n",prin6)
122
123 b01 = CNOT.apply_operator(psi_1)
124 prin7 = b01.__repr__()
125 print("\nThe Bell State b01 is the Quantum State:\n",prin7)
126
127
128 measure_2 = b01.measurement_2_qubit()
129 print("\nThe measurement of b01 ball state is:\n",measure_2)
130
131 plt.figure(2)
132 plt.bar(names,measure_2,width=0.2,color='red')
133 # x-axis label
134 plt.xlabel('Quantum States')
135 # y-axis label
136 plt.ylabel('Probability Distribution %')
137 # plot title
138 plt.title('Bell State b01')
139

```

ΚΒΑΝΤΙΚΗ ΜΗΧΑΝΙΚΗ ΜΑΘΗΣΗ, ΕΦΑΡΜΟΓΕΣ ΚΑΙ ΥΛΟΠΟΙΗΣΗ ΣΕ ΚΒΑΝΤΙΚΟ HARDWARE

```
141
142 print("\nImplementation of Bell State b10 = (1/sqrt(2))(|00>-|11>)\n")
143
144 one_zeroT = multi_tensor_states(one,zero) #dianisma grammis
145 one_zero = one_zeroT.transpose_state() #dianisma sthlhs
146 prin8 = one_zero.__repr__()
147 print("\nThe tensor product between |1> and |0> is the Quantum State:\n",prin8)
148
149 print("\nThe tensor product between Hadamard and Identity Gate is the Quantum
      Operator:\n",prin2)
150
151 psi_2 = H_I.apply_operator(one_zero)
152 prin9 = psi_2.__repr__()
153 print("\nThe (H_I)|10> is the Quantum State:\n",prin9)
154
155 b10 = CNOT.apply_operator(psi_2)
156 prin10 = b10.__repr__()
157 print("\nThe Bell State b10 is the Quantum State:\n",prin10)
158
159
160 measure_3 = b10.measurement_2_qubit()
161 print("\nThe measurement of b10 ball state is:\n",measure_3)
162
163 plt.figure(3)
164 plt.bar(names,measure_3,width=0.2,color='red')
165 # x-axis label
166 plt.xlabel('Quantum States')
167 # y-axis label
168 plt.ylabel('Probability Distribution %')
169 # plot title
170 plt.title('Bell State b10')
171
```

```
73
74 print("\nImplementation of Bell State b11 = (1/sqrt(2))(|01>-|10>)\n")
75
76 one_oneT = multi_tensor_states(one,one) #dianisma grammis
77 one_one = one_oneT.transpose_state() #dianisma sthlhs
78 prin11 = one_one.__repr__()
79 print("\nThe tensor product between |1> and |1> is the Quantum State:\n",prin11)
80
81 print("\nThe tensor product between Hadamard and Identity Gate is the Quantum
      Operator:\n",prin2)
82
83 psi_3 = H_I.apply_operator(one_one)
84 prin12 = psi_3.__repr__()
85 print("\nThe (H_I)|11> is the Quantum State:\n",prin12)
86
87 b11 = CNOT.apply_operator(psi_3)
88 prin13 = b11.__repr__()
89 print("\nThe Bell State b11 is the Quantum State:\n",prin13)
90
91 measure_4 = b11.measurement_2_qubit()
92 print("\nThe measurement of b11 ball state is:\n",measure_4)
93
94 plt.figure(4)
95 plt.bar(names,measure_4,width=0.2,color='red')
96 # x-axis label
97 plt.xlabel('Quantum States')
98 # y-axis label
99 plt.ylabel('Probability Distribution %')
100 # plot title
101 plt.title('Bell State b11')
102
103
```

```

203
204 plt.show()
205
206 fig, axs = plt.subplots(2, 2)
207
208 axs[0, 0].bar(names,measure_1,width=0.2,color='red')
209 axs[0, 0].set_ylabel='Probability Distribution %')
210 axs[0, 0].set_title('Bell State b00')
211
212 axs[0, 1].bar(names,measure_2,width=0.2,color='green')
213 axs[0, 1].set_title('Bell State b01')
214
215 axs[1, 0].bar(names,measure_3,width=0.2,color='orange')
216 axs[1, 0].set_xlabel='Bell State b10', ylabel='Probability Distribution %')
217
218 axs[1, 1].bar(names,measure_4,width=0.2,color='yellow')
219 axs[1, 1].set_xlabel='Bell State b11')
220
221 plt.show()
222
223

```

5. QPCA.py:

```

1 import numpy as np
2 import math
3 import cmath
4 import random
5 import matplotlib
6 import matplotlib.pyplot as plt
7
8 from Quantum_Library import create_Qstate, create_Qoperator
9
10 # A function that produces tensor product between multiple states
11 def multi_tensor_states(*args):
12     value = create_Qstate([[1.0]])
13     for i in args:
14         value = value.tensor_product_state(i)
15     return value
16
17 # A function that produces tensor product between multiple operators
18 def multi_tensor_operators(*args):
19     value = create_Qoperator([[1.0]])
20     for i in args:
21         value = value.tensor_product_operator(i)
22     return value
23
24 # Dhmiourgia arxikis katastashs kai pylwn
25
26 # State |0000>
27 zero = create_Qstate([1,0])
28 zero5T = multi_tensor_states(zero,zero,zero,zero,zero) #dianysma grammis
29 zero5 = zero5T.transpose_state() #dianysma sthlhs
30
31
32 # Identity Gate
33 I = create_Qoperator([
34
35         [1,0],
36         [0,1],
37 ])
38

```

ΚΒΑΝΤΙΚΗ ΜΗΧΑΝΙΚΗ ΜΑΘΗΣΗ, ΕΦΑΡΜΟΓΕΣ ΚΑΙ ΥΛΟΠΟΙΗΣΗ ΣΕ ΚΒΑΝΤΙΚΟ HARDWARE

```
39 # Hadamard Gate
40 H = create_Qoperator([
41
42         [1/2**0.5, 1/2**0.5],
43         [1/2**0.5, -1/2**0.5],
44 ])
45
46 # CNOT-CX Gate
47 CNOT = create_Qoperator([
48
49         [1,0,0,0],
50         [0,1,0,0],
51         [0,0,0,1],
52         [0,0,1,0],
53 ])
54
55 # reverse CNOT-CX Gate
56 CNOT_rev = create_Qoperator([
57
58         [1,0,0,0],
59         [0,0,0,1],
60         [0,0,1,0],
61         [0,1,0,0],
62 ])
63
64 # T Gate ||  $R_\phi(\phi=\pi/4)$  Gate &&  $e^{i\pi/4} = \cos(\pi/4) + i\sin(\pi/4)$ 
65 T = create_Qoperator([
66
67         [1,0],
68         [0,math.cos(math.pi/4)+1j*math.sin(math.pi/4)],
69 ])
70
71 # T_dagger Gate ||  $R_\phi(\phi=\pi/4)$  Gate &&  $e^{-i\pi/4} = \cos(\pi/4) - i\sin(\pi/4)$ 
72 T_dagger = create_Qoperator([
73
74         [1,0],
75         [0,math.cos(math.pi/4)-1j*math.sin(math.pi/4)],
76 ])
77
78 # UA Physical Gate &&  $e^{i\phi} = \cos(\phi) + i\sin(\phi)$ 
79 theta = 0.3787
80 phi = 1.1775
81 lamda = 1.68
82 UA = create_Qoperator([
83
84         [math.cos(theta/2),-1*math.sin(theta/2)*math.cos(lamda)-1j*math.sin(theta/2)*math.sin(lamda)],
85         [math.sin(theta/2)*math.cos(phi)+1j*math.sin(theta/2)*math.sin(phi),math.cos(theta/2)*math.cos(lamda+phi)+1j*math.cos(theta/2)*math
            .sin(lamda+phi)],
86 ])
87
88 # UB Physical Gate &&  $e^{i\phi} = \cos(\phi) + i\sin(\phi)$ 
89 theta = 1.1775
90 phi = -0.735
91 lamda = 0.3787
92 UB = create_Qoperator([
93
94         [math.cos(theta/2),-1*math.sin(theta/2)*math.cos(lamda)-1j*math.sin(theta/2)*math.sin(lamda)],
95         [math.sin(theta/2)*math.cos(phi)+1j*math.sin(theta/2)*math.sin(phi),math.cos(theta/2)*math.cos(lamda+phi)+1j*math.cos(theta/2)*math
            .sin(lamda+phi)],
96 ])
97
98 # UA' Physical Gate &&  $e^{i\phi} = \cos(\phi) + i\sin(\phi)$ 
99 theta = 1.68
100 phi = 0
101 lamda = 0
102 UA_ton = create_Qoperator([
103
104         [math.cos(theta/2),-1*math.sin(theta/2)*math.cos(lamda)-1j*math.sin(theta/2)*math.sin(lamda)],
105         [math.sin(theta/2)*math.cos(phi)+1j*math.sin(theta/2)*math.sin(phi),math.cos(theta/2)*math.cos(lamda+phi)+1j*math.cos(theta/2)*math
            .sin(lamda+phi)],
106 ])
107
```

ΚΕΦΑΛΑΙΟ 8: ΠΑΡΑΡΤΗΜΑΤΑ

```

108 # Toffoli-CCNOT Gate
109 Toffoli = create_Qoperator([
110     [1,0,0,0,0,0,0,0],
111     [0,1,0,0,0,0,0,0],
112     [0,0,1,0,0,0,0,0],
113     [0,0,0,1,0,0,0,0],
114     [0,0,0,0,1,0,0,0],
115     [0,0,0,0,0,1,0,0],
116     [0,0,0,0,0,0,1,0],
117     [0,0,0,0,0,0,0,1],
118 ])
119
120 # reverse Toffoli-CCNOT Gate
121 Toffoli_rev = create_Qoperator([
122     [1,0,0,0,0,0,0,0],
123     [0,1,0,0,0,0,0,0],
124     [0,0,1,0,0,0,0,0],
125     [0,0,0,1,0,0,0,0],
126     [0,0,0,0,1,0,0,0],
127     [0,0,0,0,0,1,0,0],
128     [0,0,0,0,0,0,1,0],
129     [0,0,0,0,0,0,0,1],
130 ])
131
132 # Fredkin-CSWAP Gate
133 CSWAP = create_Qoperator([
134     [1,0,0,0,0,0,0,0],
135     [0,1,0,0,0,0,0,0],
136     [0,0,1,0,0,0,0,0],
137     [0,0,0,1,0,0,0,0],
138     [0,0,0,0,1,0,0,0],
139     [0,0,0,0,0,1,0,0],
140     [0,0,0,0,0,0,1,0],
141     [0,0,0,0,0,0,0,1],
142 ])
143
144
145
146
149
150 # Dhmiourgia tensor products
151
152 # a-->(IxIxUAxUAxI)
153 I_I_UA_UA_I = multi_tensor_operators(I,I,UA,UA,I)
154
155 # b-->(IxCNOTxIxI)
156 I_CNOT_I_I = multi_tensor_operators(I,CNOT,I,I)
157
158 # c-->(CNOTxIxIxI)
159 CNOT_I_I_I = multi_tensor_operators(CNOT,I,I,I)
160
161 # d-->(CNOTxIxCNOTreverse)
162 CNOT_I_CNOTreverse = multi_tensor_operators(CNOT,I,CNOT_rev)
163
164 # e-->(UBxIxUA'xUA'xUB)
165 UB_I_UA_ton_UA_ton_UB = multi_tensor_operators(UB,I,UA_ton,UA_ton,UB)
166
167 # f-->(IxHxIxIxI)
168 I_H_I_I_I = multi_tensor_operators(I,H,I,I,I)
169
170
171 ### Shmeio 2 tropwn #####
172
173 # 1 tropos
174
175 # g-->(IxIxCNOTxI)
176 I_I_CNOT_I = multi_tensor_operators(I,I,CNOT,I)
177
178 # h-->(IxToffolireversexI)
179 I_Toffolireverse_I = multi_tensor_operators(I,Toffoli_rev,I)
180
181

```

ΚΒΑΝΤΙΚΗ ΜΗΧΑΝΙΚΗ ΜΑΘΗΣΗ, ΕΦΑΡΜΟΓΕΣ ΚΑΙ ΥΛΟΠΟΙΗΣΗ ΣΕ ΚΒΑΝΤΙΚΟ HARDWARE

```
181
182 # 2 tropos
183
184 # k-->(IxCSWAPxI)
185 #I_CSWAP_I = multi_tensor_operators(I,CSWAP,I)
186
187 #####
188
189
190 # QPCA Cicuit Implementation
191
192
193 # (IxIxUAxUAxI)|0000> = |ψ1>
194 psi_1 = I_I_UA_UA_I.apply_operator(zero5)
195
196 # (IxCNOTxIxI)|ψ1> = |ψ2>
197 psi_2 = I_CNOT_I_I.apply_operator(psi_1)
198
199 # (CNOTxIxIxI)|ψ2> = |ψ3>
200 psi_3 = CNOT_I_I_I.apply_operator(psi_2)
201
202 # (IxCNOTxIxI)|ψ3> = |ψ4>
203 psi_4 = I_CNOT_I_I.apply_operator(psi_3)
204
205 # (CNOTxIxCNOTreverse)|ψ4> = |ψ5>
206 psi_5 = CNOT_I_CNOTreverse.apply_operator(psi_4)
207
208 # (UBxIxUA'xUA'xUB)|ψ5> = |ψ6>
209 psi_6 = UB_I_UA_ton_UA_ton_UB.apply_operator(psi_5)
210
211 # (IxHxIxIxI)|ψ6> = |ψ7>
212 psi_7 = I_H_I_I_I.apply_operator(psi_6)
213
214
215
216
217
218
219
220
221
222
223
224
225 ##### 2 tropos
226
227 # (IxCSWAPxI)|ψ7> = |ψ8>
228 #psi_8 = I_CSWAP_I.apply_operator(psi_7)
229
230 # (IxHxIxIxI)|ψ8> = |ψ9>
231 #psi_9 = I_H_I_I_I.apply_operator(psi_8)
232
233 #####
234
235 # (IxHxIxIxI)|ψ10> = |ψ11>
236 psi_11 = I_H_I_I_I.apply_operator(psi_10)
237
238 print("\nFinal State:\n",psi_11)
239
240 measure = psi_11.measurement_5_qubit()
241 print("\nThe measurement of Final State is:\n",measure)
242
243
```

ΚΕΦΑΛΑΙΟ 8: ΠΑΡΑΡΤΗΜΑΤΑ

```
243
244 #names_5 = ['|00000>', '|00001>', '|00010>', '|00011>', '|00100>', '|00101>', '|00110>', '|00111>', '|01000>', '|01001>', '|01010>', '|01011>',
'|01100>', '|01101>', '|01110>', '|01111>', '|10000>', '|10001>', '|10010>', '|10011>', '|10100>', '|10101>', '|10110>', '|10111>',
'|11000>', '|11001>', '|11010>', '|11011>', '|11100>', '|11101>', '|11110>', '|11111>']
245 names_5 = ['|0>', '|1>', '|2>', '|3>', '|4>', '|5>', '|6>', '|7>', '|8>', '|9>', '|10>', '|11>', '|12>', '|13>', '|14>', '|15>', '|16>', '|17>',
'|18>', '|19>', '|20>', '|21>', '|22>', '|23>', '|24>', '|25>', '|26>', '|27>', '|28>', '|29>', '|30>', '|31>']
246
247
248 plt.figure(1)
249 plt.bar(names_5,measure,width=0.2,color='red')
250 # x-axis label
251 plt.xlabel('Quantum States(Decimal[0-31]-Binary[2^5=32])')
252 # y-axis label
253 plt.ylabel('Probability Distribution %')
254 # plot title
255 plt.title('QPCA')
256
257
258
259 plt.show()
260
```

ΠΑΡΑΡΤΗΜΑ 2: ΔΗΜΙΟΥΡΓΙΑ ΠΡΟΣΟΜΟΙΩΤΗ ΓΙΑ ΑΝΑΛΥΣΗ ΤΟΥ ΚΒΑΝΤΙΚΟΥ ΚΥΚΛΩΜΑΤΟΣ ΔΗΜΙΟΥΡΓΙΑΣ BELL ΚΑΤΑΣΤΑΣΕΩΝ

Παρακάτω παρουσιάζονται τα αποτελέσματα της υλοποίησης του κβαντικού κυκλώματος των Bell States μέσω δικού μου κώδικα σε γλώσσα Python, που υπάρχει παραπάνω στο Παράρτημα 1 με όνομα Bell States Demonstration.py.

Σε αυτό το σημείο, θα παρουσιαστούν τα αποτελέσματα της υλοποίησης του κβαντικού κυκλώματος για τη δημιουργία των καταστάσεων Bell, μέσω δικού μου κώδικα σε γλώσσα Python. Τη θεωρία των καταστάσεων Bell την είδαμε παραπάνω και πλέον γνωρίζουμε ότι αποτελούν κβαντικά εναγκαλισμένες καταστάσεις.

Η δημιουργία και τα αποτελέσματα της κατάστασης Bell $b00 = \frac{|00\rangle + |11\rangle}{\sqrt{2}}$ φαίνονται στη παρακάτω εικόνα:

```
Implementation of Bell State b00 = (1/sqrt(2))*(|00>+|11>)

The tensor product between |0> and |0> is the Quantum State:

<The Quantum State that is created is:
[1.], [0.], [0.], [0.]>

The tensor product between Hadamard and Identity Gate is the Quantum Operator:

<The Quantum Operator that is created is:
[[ 0.70710678  0.          0.70710678  0.
   0.          0.70710678  0.          0.70710678]
 [ 0.          0.70710678  0.          0.70710678]
 [ 0.70710678  0.         -0.70710678 -0.
   0.          0.70710678 -0.         -0.70710678]]>

The (H_I)|00> is the Quantum State:

<The Quantum State that is created is:
[0.70710678], [0.], [0.70710678], [0.]>

The Bell State b00 is the Quantum State:

<The Quantum State that is created is:
[0.70710678], [0.], [0.], [0.70710678]>

The measurement of b00 ball state is:
(49.365234375, 0.0244140625, 0.048828125, 50.5615234375)
```


ΚΕΦΑΛΑΙΟ 8: ΠΑΡΑΡΤΗΜΑΤΑ

Η δημιουργία και τα αποτελέσματα της κατάστασης Bell $b01 = \frac{|01\rangle + |10\rangle}{\sqrt{2}}$ είναι:

```
Implementation of Bell State b01 = (1/sqrt(2))(|01>+|10>)

The tensor product between |0> and |1> is the Quantum State:
<The Quantum State that is created is:
[0.], [1.], [0.], [0.]>

The tensor product between Hadamard and Identity Gate is the Quantum Operator:
<The Quantum Operator that is created is:
[[ 0.70710678  0.          0.70710678  0.          ]
 [ 0.          0.70710678  0.          0.70710678]
 [ 0.70710678  0.         -0.70710678 -0.          ]
 [ 0.          0.70710678 -0.          -0.70710678]]>

The (H_I)|01> is the Quantum State:
<The Quantum State that is created is:
[0.], [0.70710678], [0.], [0.70710678]>

The Bell State b01 is the Quantum State:
<The Quantum State that is created is:
[0.], [0.70710678], [0.70710678], [0.]>

The measurement of b01 ball state is:
(0.0, 51.0498046875, 48.876953125, 0.0732421875)
```

Η δημιουργία και τα αποτελέσματα της κατάστασης Bell $b10 = \frac{|00\rangle - |11\rangle}{\sqrt{2}}$ είναι:

```
Implementation of Bell State b10 = (1/sqrt(2))(|00>-|11>)

The tensor product between |1> and |0> is the Quantum State:
<The Quantum State that is created is:
[0.], [0.], [1.], [0.]>

The tensor product between Hadamard and Identity Gate is the Quantum Operator:
<The Quantum Operator that is created is:
[[ 0.70710678  0.          0.70710678  0.          ]
 [ 0.          0.70710678  0.          0.70710678]
 [ 0.70710678  0.         -0.70710678 -0.          ]
 [ 0.          0.70710678 -0.          -0.70710678]]>

The (H_I)|10> is the Quantum State:
<The Quantum State that is created is:
[0.70710678], [0.], [-0.70710678], [0.]>

The Bell State b10 is the Quantum State:
<The Quantum State that is created is:
[0.70710678], [0.], [0.], [-0.70710678]>

The measurement of b10 ball state is:
(49.853515625, 0.0244140625, 0.048828125, 50.0732421875)
```

Η δημιουργία και τα αποτελέσματα της κατάστασης Bell $b11 = \frac{|01\rangle - |10\rangle}{\sqrt{2}}$ είναι:

```
Implementation of Bell State b11 = (1/sqrt(2))(|01>-|10>)

The tensor product between |1> and |1> is the Quantum State:
<The Quantum State that is created is:
[0.], [0.], [0.], [1.]>

The tensor product between Hadamard and Identity Gate is the Quantum Operator:
<The Quantum Operator that is created is:
[[ 0.70710678  0.          0.70710678  0.          ]
 [ 0.          0.70710678  0.          0.70710678]
 [ 0.70710678  0.         -0.70710678 -0.          ]
 [ 0.          0.70710678 -0.          -0.70710678]]>

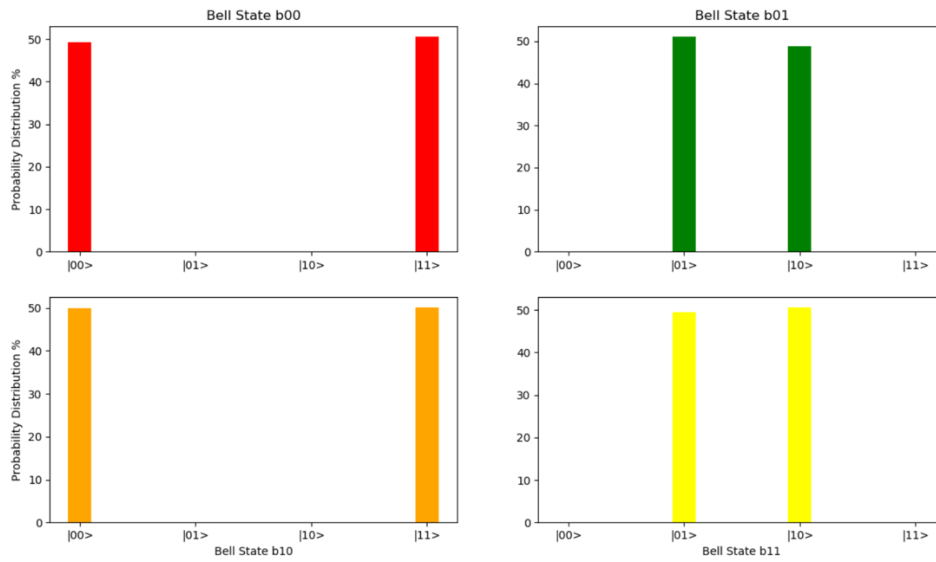
The (H_I)|11> is the Quantum State:
<The Quantum State that is created is:
[0.], [0.70710678], [0.], [-0.70710678]>

The Bell State b11 is the Quantum State:
<The Quantum State that is created is:
[0.], [0.70710678], [-0.70710678], [0.]>

The measurement of b11 ball state is:
(0.0, 49.3896484375, 50.537109375, 0.0732421875)
```

Τα παραπάνω είναι τα αποτελέσματα των πράξεων μέχρι και την μέτρηση για κάθε μία κατάσταση Bell.

Το παρακάτω διάγραμμα μας οπτικοποιεί την μέτρηση κάθε κατάστασης Bell. Παρατηρούμε αυτό που ήδη ξέραμε, ότι δηλαδή, κάθε κατάσταση Bell μπορεί να βρίσκεται σε δύο καταστάσεις με ποσοστό περίπου 50% στη καθεμία.



- Η b00 μπορεί να είναι 50% $|00\rangle$ και 50% $|11\rangle$.
- Η b01 μπορεί να είναι 50% $|01\rangle$ και 50% $|10\rangle$.
- Η b10 μπορεί να είναι 50% $|00\rangle$ και 50% $|11\rangle$.
- Η b11 μπορεί να είναι 50% $|01\rangle$ και 50% $|10\rangle$.

ΠΑΡΑΡΤΗΜΑ 3: ΚΩΔΙΚΑΣ ΓΙΑ ΤΗΝ ΠΛΑΤΦΟΡΜΑ QISKIT ΤΗΣ IBM

Σε αυτό το παράρτημα περιέχεται ο κώδικας για την πλατφόρμα qiskit της IBM με την υλοποίηση του QPCA, με αρχικά δεδομένα αριθμού δωματίων και τετραγωνικά μέτρα δωματίων διαφόρων σπιτιών, δηλαδή 2 διανυσμάτων των 15 στοιχείων το καθένα, ο οποίος μπορεί να τρέξει στο jupyter notebook και παράγει το κβαντικό κύκλωμα του QPCA καθώς και τα διαγράμματα στο 5^ο κεφάλαιο συγκεκριμένα στην ενότητα 5.2.

```

1  |from qiskit import *
2
3  qr = QuantumRegister(5)
4  cr = ClassicalRegister(1)
5  #cr = ClassicalRegister(5)
6
7  QPCA_circuit = QuantumCircuit(qr,cr)
8
9  %matplotlib inline
10
11
12  QPCA_circuit.u3(0.3787, 1.1775, 1.68, qr[1])
13  QPCA_circuit.u3(0.3787, 1.1775, 1.68, qr[2])
14  QPCA_circuit.barrier()
15  QPCA_circuit.cx(qr[1], qr[0])
16  QPCA_circuit.cx(qr[2], qr[4])
17  QPCA_circuit.barrier()
18  QPCA_circuit.u3(1.1775, -0.735, 0.3787, qr[0])
19  QPCA_circuit.u3(1.68, 0, 0, qr[1])
20  QPCA_circuit.u3(1.68, 0, 0, qr[2])
21  QPCA_circuit.u3(1.1775, -0.735, 0.3787, qr[4])
22  QPCA_circuit.barrier()
23  QPCA_circuit.h(qr[3])
24  QPCA_circuit.barrier()
25  QPCA_circuit.h(qr[1])
26  QPCA_circuit.h(qr[2])
27  QPCA_circuit.cx(qr[2], qr[1])
28  QPCA_circuit.h(qr[1])
29  QPCA_circuit.h(qr[2])
30  QPCA_circuit.h(qr[1])
31  QPCA_circuit.cx(qr[2], qr[1])
32  QPCA_circuit.tdg(qr[1])
33  QPCA_circuit.cx(qr[2], qr[1])
34  QPCA_circuit.cx(qr[3], qr[2])
35  QPCA_circuit.cx(qr[2], qr[1])
36  QPCA_circuit.cx(qr[3], qr[2])
37  QPCA_circuit.barrier(qr[1])
38  QPCA_circuit.t(qr[1])
39  QPCA_circuit.cx(qr[2], qr[1])
40  QPCA_circuit.tdg(qr[1])
41  QPCA_circuit.cx(qr[2], qr[1])
42  QPCA_circuit.cx(qr[3], qr[2])
43  QPCA_circuit.cx(qr[2], qr[1])
44  QPCA_circuit.cx(qr[3], qr[2])
45  QPCA_circuit.barrier(qr[1])

```

```
46 QPCA_circuit.t(qr[1])
47 QPCA_circuit.t(qr[2])
48 QPCA_circuit.h(qr[1])
49 QPCA_circuit.cx(qr[3], qr[2])
50 QPCA_circuit.barrier(qr[1])
51 QPCA_circuit.tdg(qr[2])
52 QPCA_circuit.t(qr[3])
53 QPCA_circuit.barrier(qr[1])
54 QPCA_circuit.cx(qr[3], qr[2])
55 QPCA_circuit.h(qr[1])
56 QPCA_circuit.h(qr[2])
57 QPCA_circuit.barrier(qr[3])
58 QPCA_circuit.cx(qr[2], qr[1])
59 QPCA_circuit.barrier(qr[3])
60 QPCA_circuit.h(qr[1])
61 QPCA_circuit.h(qr[2])
62 QPCA_circuit.h(qr[3])
63
64 QPCA_circuit.measure(qr[3], cr[0])
65 #QPCA_circuit.measure(qr[0], cr[0])
66 #QPCA_circuit.measure(qr[1], cr[1])
67 #QPCA_circuit.measure(qr[2], cr[2])
68 #QPCA_circuit.measure(qr[3], cr[3])
69 #QPCA_circuit.measure(qr[4], cr[4])
70
```

```
70
71 #simulation
72 simulator = Aer.get_backend('qasm_simulator')
73 result = execute(QPCA_circuit, backend = simulator, shots = 4096).result()
74 from qiskit.tools.visualization import plot_histogram
75 plot_histogram(result.get_counts(QPCA_circuit))
76
77 #QPCA_circuit.draw()
78 #QPCA_circuit.draw(output='mpl')
79
80 #connect account - run in real hardware
81 #IBMQ.load_account()
82 #provider = IBMQ.get_provider('ibm-q')
83 #qcomp = provider.get_backend('ibmq_london')
84 #job = execute(QPCA_circuit, backend=qcomp, shots = 4096)
85 #from qiskit.tools.monitor import job_monitor
86 #job_monitor(job)
87 #result = job.result()
88 #plot_histogram(result.get_counts(QPCA_circuit))
89
```

ΠΑΡΑΡΤΗΜΑ 4: ΤΑΝΥΣΤΙΚΟ ΓΙΝΟΜΕΝΟ ΠΙΝΑΚΩΝ

Εδώ θα αναλύσουμε το τανυστικό γινόμενο πινάκων. Για να περιγράψουμε 2 qubits, χρειαζόμαστε έναν νέο διανυσματικό χώρο Hilbert, ο οποίος εάν ονομάσουμε H_A το χώρο του qubit A και H_B το χώρο του qubit B, ορίζεται ως το τανυστικό γινόμενο:

$$H_{AB} = H_A \otimes H_B$$

Για γραμμικές απεικονίσεις, το τανυστικό γινόμενο δύο πινάκων ταυτίζεται με το γινόμενο Kronecker αυτών. Έστω δύο πίνακες $A = \begin{pmatrix} \alpha_{11} & \alpha_{12} \\ \alpha_{21} & \alpha_{22} \end{pmatrix}$ και $B = \begin{pmatrix} \beta_{11} & \beta_{12} \\ \beta_{21} & \beta_{22} \end{pmatrix}$. Μετά την πράξη του τανυστικού γινομένου ο πίνακας έχει την εξής μορφή:

$$A \otimes B = \begin{pmatrix} \alpha_{11}B & \alpha_{12}B \\ \alpha_{21}B & \alpha_{22}B \end{pmatrix} = \begin{pmatrix} \alpha_{11}\beta_{11} & \alpha_{11}\beta_{12} & \alpha_{12}\beta_{11} & \alpha_{12}\beta_{12} \\ \alpha_{11}\beta_{21} & \alpha_{11}\beta_{22} & \alpha_{12}\beta_{21} & \alpha_{12}\beta_{22} \\ \alpha_{21}\beta_{11} & \alpha_{21}\beta_{12} & \alpha_{22}\beta_{11} & \alpha_{22}\beta_{12} \\ \alpha_{21}\beta_{21} & \alpha_{21}\beta_{22} & \alpha_{22}\beta_{21} & \alpha_{22}\beta_{22} \end{pmatrix}$$

Αντισταθμίζουμε από το παραπάνω, αλλά και από την ενότητα 2.3 που περιγράψαμε τον κβαντικό καταχωρητή και το τανυστικό γινόμενο ανάμεσα σε κβαντικές καταστάσεις, ότι ένα qubit μπορεί να περιγραφεί από ένα διάνυσμα στήλης και ότι μπορούμε να δράσουμε πάνω του έναν τελεστή, που είναι πίνακας μεγέθους 2×2 , αν θέλουμε να μετασχηματίσουμε την κατάσταση του. Αντιστοίχως για δύο qubits μπορούμε να δράσουμε τελεστές μεγέθους 4×4 , για τρία qubits μπορούμε να δράσουμε τελεστές μεγέθους 8×8 κ.ο.κ.

Στη γενική περίπτωση, λοιπόν, ορίζουμε έναν σύνθετο χώρο Hilbert διαστάσεων $N = 2^n$, όπου n είναι το πλήθος των qubits, καθώς και το τανυστικό γινόμενο του χώρου Hilbert για τα qubits, ως:

$$H = \bigotimes_{i=1}^n H_i$$

Αν υποθέσουμε ότι οι μετασχηματισμοί που μπορεί να υποστεί ένα σύστημα σε αυτό το χώρο, συμβολίζεται με P_k (οποιαδήποτε πύλη), τα στοιχεία που ανήκουν σε αυτόν το χώρο ορίζονται ως:

$$P_k = \bigotimes_{j=1}^n P_{k_j}$$

Για το τανυστικό γινόμενο δύο τελεστών, ισχύουν οι παρακάτω ιδιότητες:

- Είναι γραμμικό.
- Αν οι τελεστές A και B είναι ερμιτιανοί, τότε ο τελεστής $A \otimes B$ είναι ερμιτιανός.
- Αν οι τελεστές A και B είναι προβολικοί, τότε ο τελεστής $A \otimes B$ είναι προβολικός.
- Αν οι τελεστές A και B είναι μοναδιαίοι, τότε ο τελεστής $A \otimes B$ είναι μοναδιαίος.
- Αν οι τελεστές A και B είναι θετικοί, τότε ο τελεστής $A \otimes B$ είναι θετικός.

Στην ενότητα 2.4, που αναφέρεται στον κβαντικό εναγκαλισμό, μάθαμε τι σημαίνει ο όρος εναγκαλισμένα σωματίδια και αναφερθήκαμε στο πότε μια κατάσταση είναι διαχωρίσιμη. Εδώ τώρα, μπορούμε να ορίσουμε ότι:

‘Όταν λέμε ότι ένα σύστημα βρίσκεται σε κατάσταση γινομένου ή σε διαχωρίσιμη κατάσταση, αυτό σημαίνει ότι μπορεί να γραφεί ως τανυστικό γινόμενο των καταστάσεων των επιμέρους συστημάτων.’