Zespołowe zadanie programistyczne nr 2.

Tomografia stanu kwantowego kubitu (przypadek tzw. stanu czystego).

TERMIN I (regularny) : do 30.06.2020 (wtorek)
TERMIN II (uzupełniający bez konsekwencji za opóźnienie) : do 01.09.2020 (wtorek) godz. 24:00
TERMIN III (w przypadku uzyskania oceny niedostatecznej, np. na skutek niewykonania projektu we wcześniejszych terminach) : do 25.09.2020 (piątek) godz. 24:00.

I Treść zadania

0. Metodyka przygotowania do realiacji zadania

0.1. Należy zapoznać się z treścią wykładu zawartą w e-konspekcie, określanym dalej mianem tutorialu, pt.: : „FI_2019_2020_QI_Część_I.pdf" (zamieszczonego w chmurze PP).
Notacja [T-*liczba*] w dalszej części tekstu niniejszego opracowania oznacza odwołanie do slajdu tutorialu .
0.2. Załóż konto na platformie IBM Q https://quantum-computing.ibm.com/login [T-7] .
0.3 Wskazane jest samodzielne wykonanie przykładów zawartych w tutorialu.
0.4 Należy przeanalizować przykłady zawarte w pliku: FI_Pomiary_XYZ.ipynb (lub FI_Pomiary_XYZ.py , FI_Pomiary_XYZ.pdf). W tym celu należy wczytać plik FI_Pomiary_XYZ.ipynb do sytemu IBM Q.
0.5 Należy zapoznać się z dalsza częścią niniejszego opracowania.
0.6 Literatura: https://qiskit.org/textbook/preface.html .

1. Założenia zadania .

1.1. Dane są trzy liczby

```
In[1]:= k1 = Mod[A1 + A2 + A3 + A4, 2]; k2 = Mod[A1 + A2 + A3 + A4, 2^2]; k3 = Mod[A1 + A2 + A3 + A4, 2^3];
           modulo                          modulo                              modulo
```

gdzie liczby A1, A2, A3, A4 są zdefiniowane przez trzy ostatnie cyfry numerów albumów studentów wchodzących w skład zespołu.
Zauważmy, że k1 =0 lub 1, natomiast liczby k2 i k3 pozwalają zdefiniować kąty:

In[2]:= $\theta 0 = \frac{\pi}{k2};\ \phi 0 = \frac{2\pi}{k3};$

1.2. Zdefiniowany jest stan kubitu opisany przez wektor $|k1\rangle$;

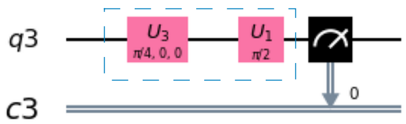
jeżeli k1=0 to $|0\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$, inaczej $|1\rangle = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$ [T-15].

In[3]:=
```
|0⟩ = {{1}, {0}}; |1⟩ = {{0}, {1}};
```

1.3. Zdefiniowane są operacje (bramki kwantowe) na stanach kubitu określone przez macierze u1 i u3.

In[4]:=
$$\text{u1}[\phi\_] = \begin{pmatrix} 1 & 0 \\ 0 & e^{i\,\phi 0} \end{pmatrix};\ \text{u3}[\theta\_, 0, 0] := \begin{pmatrix} \text{Cos}\left[\frac{\theta}{2}\right] & \text{Sin}\left[\frac{\theta}{2}\right] \\ -\text{Sin}\left[\frac{\theta}{2}\right] & \text{Cos}\left[\frac{\theta}{2}\right] \end{pmatrix}$$

Iloczyn tych macierzy odpowiada fragmentowi obwodu kwantowego, otoczonemu na rysunku prostokątem.



Rys. Obwód kwantowy przygotowujący stan kubitu

Działanie złożenia operacji u1.u3 na stan |0⟩ lub |1⟩ prezentują poniższe przykłady.

In[5]:=
```
(u1[ϕ0].u3[θ0, 0, 0].|0⟩) // MatrixForm
```
postać macierzy

Out[5]//MatrixForm=
$$\begin{pmatrix} \text{Cos}\left[\frac{\pi}{2\,\text{Mod}[A1+A2+A3+A4,4]}\right] \\ -e^{\frac{2\,i\,\pi}{\text{Mod}[A1+A2+A3+A4,8]}}\ \text{Sin}\left[\frac{\pi}{2\,\text{Mod}[A1+A2+A3+A4,4]}\right] \end{pmatrix}$$

In[6]:=
```
(u1[ϕ0].u3[θ0, 0, 0].|1⟩) // MatrixForm
```
postać macierzy

Out[6]//MatrixForm=
$$\begin{pmatrix} \text{Sin}\left[\frac{\pi}{2\,\text{Mod}[A1+A2+A3+A4,4]}\right] \\ e^{\frac{2\,i\,\pi}{\text{Mod}[A1+A2+A3+A4,8]}}\ \text{Cos}\left[\frac{\pi}{2\,\text{Mod}[A1+A2+A3+A4,4]}\right] \end{pmatrix}$$

2. Polecenia:

2.1 Dla parametrów k1, k2, k3 wyznacz teoretyczny wynik działania u1[ϕ0].u3[θ0,0,0].|k1⟩.

2.2 Korzystając z notatnika Jupyter , na platformie IBM Q [T-8] i języka Python 3 napisz program , który implementuje operację u1[ϕ0].u3[θ0,0,0].|k1⟩. Skorzystaj z pliku: FI_Pomiary_XYZ.ipynb (lub FI_Pomiary_XYZ.py , FI_Pomiary_XYZ.pdf) zamieszczonych w chmurze PP, który zawiera stosowny przykład (ostatni obwód kwantowy w pliku). Jeżeli k1=1 musisz korzystać również z bramki X. [T-28, T-29, T-32] .

2.3 Rozwiń ten program aby uwzględniał pomiary typu X , Y i Z [T-45, T-48, T-51] . Pomocnicze przykłady zamieszczono również w pliku: FI_Pomiary_XYZ.ipynb (lub FI_Pomiary_XYZ.py , FI_Pomiary_XYZ.pdf) .

2.4 Korzystając z napisanego programu wyznacz doświadczalne prawdopodobieństwa (*a posteriori*) uzyskania wyników 0 i 1 w pomiarach X, Y, Z.

2.5 Wyznacz na tej podstawie doświadczalny wektor Blocha. Wektor ten jest zdefiniowany w punktach: II/2/2.3 , III/1, IV/Krok 1 niniejszego opracowania.

2.3 Korzystając z wyznaczonego wektora Blocha i stosując algorytm opisany w punkcie IV

niniejszego opracowania wyznacz doświadczalny (*a posteriori*) stan kwantowy kubitu, który jest generowany w wyniku operacji u1[ϕ0].u3[θ0,0,0].|k1〉 .
2.6 Porównaj stan wyznaczony doświadczalnie (*a posteriori)* ze stanem wyznaczonym teoretycznie (*a priori*) zgodnie z opisem w punkcie I/1/1.3 lub II/2/2.3 niniejszego opracowania.
2.7 Każdy zespół powinien przesłać w formacie PDF:
a) ankietę z realizacji projektu, która powinna zawierać:
- skład zespołu realizującego projekt,
- wykaz prac wykonanych przez poszczególnych członków zespołu:
- procentowy udział w realizacji projektu poszczególnych członków zespołu
- wykaz przesłanych plików,
- wykaz zapożyczonych bibliotek (wskazać źródło) ,
- zrzuty ekranowe ilustrujące działanie programu, tzn. graficzne przedstawienie obwodów kwantowych wraz z graficznym przedstawieniem wyników pomiarów typu X, Y, Z;
- stan kwantowy wyznaczony *a posteriori* i *a priori*.
- zapis obliczeń wykonanych zgodnie procedurą zawartą w punkcie IV niniejszego opracowania (znamiennie można przesłać plik z adekwatnym fragmentem notatnika Mathematica zapisany w formacie PDF .
b) notatnik Jupyter zawierający ten kod zapisany w formacie PDF albo kod źródłowy w języku python.
3. Każdy z członków zespołu powinien obowiązkowo przesłać oświadczenie poprzez oficjalny mail o treści :
„Niniejszym oświadczam, że informacje zawarte w ankiecie dotyczącej **Zadania programistycznego nr 2** z fizyki dla informatyków , a przesłane przez upoważnionego przeze mnie reprezentanta zespołu, panią/pana *Imię i Nazwisko*, zostały uzgodnione w zespole o składzie :
- *Imię1 i Nazwisko*1,
- *Imię2 i Nazwisko2,*
- *Imię3 i Nazwisko3,*
- *Imię4 i Nazwisko4.*
i są zgodne ze stanem faktycznym.
Potwierdzam wyszczególnioną w ankiecie deklarację o moim osobistym udziale w realizacji **Zadania programistycznego nr 2**, który obejmował:
- *należy wyszczególnić te same informacje, które zawarto w ankiecie* .
Jestem świadom odpowiedzialności regulaminowej i prawnej za złożenie fałszywego oświadczenia.
Imię i Nazwisko
4. W komunikacji mailowej należy korzystać z oficjalnego konta studenckiego przydzielonego przez Politechnikę Poznańską.
5. W przypadku **Zadania programistycznego nr 2** standardowo nie przewiduję *colloquium.* Jednak w razie powziętych wątpliwości zastrzegam możliwość wskazania terminu takiego colloquium dla całego zespołu z prowadzącym zajęcia, w trybie audio-wideo.

In[7]:=

II Twierdzenie

1.Założenia twierdzenia:

1.1. Dane są wektory bazy przestrzeni stanów kwantowych kubitu:
$|0\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix}; |1\rangle = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$

oraz macierze $\sigma x = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$, $\sigma y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}$, $\sigma z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$.

1.2. Zdefiniowane są trzy pary wektorów :

a) $|0_z\rangle = |0\rangle$, $|1_z\rangle = |1\rangle$;

b) $|0_x\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$, $|1_x\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$;

c) $|0_y\rangle = \frac{1}{\sqrt{2}}(|0\rangle + i|1\rangle)$, $|1_y\rangle = \frac{1}{\sqrt{2}}(|0\rangle - i|1\rangle)$.

1.3. Ogólny stan kubitu określony jest przez wektor kolumnowy typu ket:
$|\psi_0\rangle = \mathrm{Cos}[\theta/2]\,|0\rangle + e^{i\phi}\,\mathrm{Sin}[\theta/2]\,|1\rangle$,

któremu odpowiada wektor typu bra : $\langle\psi_0| = |\psi_0\rangle^{\dagger} = \mathrm{Cos}[\theta/2]\,\langle 0| + e^{i\phi}\,\mathrm{Sin}[\theta/2]\,\langle 1|$.

1.4. Zdefiniowany jest wektor P=[Px,Py,Pz] taki, że
$Px = \langle\psi_0|\sigma x|\psi_0\rangle$, $Py = \langle\psi_0|\sigma y|\psi_0\rangle$, $Pz = \langle\psi_0|\sigma z|\psi_0\rangle$.

1.5. Zdefiniowana jest macierz σP=Px σx+ Py σy+Pz σz.

2. Tezy twierdzenia:

2.1. Dla wektora P zachodzi $|P|^2 = (Px)^2 + (Py)^2 + (Pz)^2 = 1$, tzn . wektor P jest wektorem jednostkowym, przy czym jego współrzędne są rzeczywiste.

2.2. Wektor , opisujący stan kubitu, jest wektorem własnym operatora (macierzy)
σP=Px σx+ Py σy+Pz σz , odpowiadającym wartości własnej λ=+1 tego operatora.

2.3. Współrzędne wektora Blocha P można obliczyć jako różnice prawdopodobieństw:

a) Px=px $[0_x]$ - px $[1_x]$, gdzie ;

b) Py=py $[0_y]$ - py $[1_y]$, gdzie ;

c) Pz=pz $[0_z]$ - pz $[1_z]$, gdzie .

Uwaga! Standardowa notacja iloczynu skalarnego w fizyce kwantowej zapisywana jest w postaci . W niniejszym opracowaniu korzystamy z notacji nawiązującej do notacji programu Mathematica .

3. Dowód :

Dowód przeprowadzimy, z wykorzystaniem programu Mathematica (*Stephen Wolfram, twórca programu Mathematica, jest fizykiem, https://www.stephenwolfram.com/, https://www.wolframphysics.org/*).

Krok 1. Definicja bazy przestrzeni wektorowej, wektora stanu oraz iloczynu skalarnego.

Zakładamy, że kąt polarny θ i kąt azymutalny ϕ, określające położenie stanu kubitu na sferze Blocha [T-19], są rzeczywiste. *Litery greckie wprowadzamy z klawiatury przez sekwencję znaków*:

σ - Esc s Esc , θ - Esc th Esc , ϕ - Esc f Esc , ψ - Esc ps Esc .

In[8]:=
```
$Assumptions = {θ ∈ Reals ∧ ϕ ∈ Reals}
```
domyślne założenia · liczby rzec··· · liczby rze

Out[8]= $\{\theta \in \mathbb{R}\ \&\&\ \phi \in \mathbb{R}\}$

Definiujemy wektory bazy i jako wektory kolumnowe. W programie Mathematica do zapisu symbolu lub stosujemy odpowiednie skróty: Esc ket Esc , Esc bra Esc . *Funkcja* **Matrix-Form** *programu Mathematica ukazuje jawną strukturę wektorów, macierzy.*

In[9]:=
```
(|0⟩ = {{1}, {0}}) // MatrixForm
```
postać macierzy

Out[9]//MatrixForm=
$$\begin{pmatrix} 1 \\ 0 \end{pmatrix}$$

In[10]:=
```
(|1⟩ = {{0}, {1}}) // MatrixForm
```
postać macierzy

Out[10]//MatrixForm=
$$\begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

Definiujemy operację sprzężenia † (hermitowskiego) , które przekształca wektor ket na wektor [T - 16]. W programie Mathematica używamy skrótu Esc ct Esc . *Zastosowanie funkcji* **Simplify[ ]** *pozwala uwzględnić założenia* $\theta \in \mathbb{R}$ *i* $\phi \in \mathbb{R}$.

In[11]:=
```
⟨i_| := Simplify[|i⟩†]
```
uprość

Definiujemy wektor ket $|\psi_0\rangle$ opisujący ogólny stan kwantowy kubitu oraz wektor dualny bra $\langle\psi_0|$, który odpowiada wektorowi ket [T - 16].

In[12]:=
```
(|ψ0⟩ = Cos[θ/2] |0⟩ + ⅇ^(ⅈ ϕ) Sin[θ/2] |1⟩) // MatrixForm
```
cosinus · sinus · postać macierzy

Out[12]//MatrixForm=
$$\begin{pmatrix} \cos\left[\frac{\theta}{2}\right] \\ e^{i\phi}\sin\left[\frac{\theta}{2}\right] \end{pmatrix}$$

In[13]:=
```
(⟨ψ0| = Simplify[|ψ0⟩†]) // MatrixForm
```
uprość · postać macierz

Out[13]//MatrixForm=
$$\begin{pmatrix} \cos\left[\frac{\theta}{2}\right] & e^{-i\phi}\sin\left[\frac{\theta}{2}\right] \end{pmatrix}$$

Definiujemy iloczyn skalarny . Symbol ten wprowadzamy za pomocą skrótu klawiszowego Esc braket Esc. *Kropka "." oznacza tu mnożenie wektora wierszowego i kolumnowego, zastosowanie ";" blokuje wyświetlanie wyniku przypisania ale nie blokuje przypisania.*

In[14]:=
```
⟨a_ | b_⟩ = ⟨a|.|b⟩;
```

Krok 2. Definicja macierzy związanych z pomiarami typu: X, Y , Z ; wektora P .

Dowód tezy 1. twierdzenia.

Po zdefiniowaniu wektorów bra i ket wprowadzamy tzw. macierze Pauliego: σx , σy, σz (oznaczane w tutorialu również jako X, Y, Z. Związane są one z trzema typami pomiarów realizowanych na kubicie (w dowolnej fizycznej implementacji) [T-45, 48, 51].

In[15]:=
```
σx = PauliMatrix[1]; σy = PauliMatrix[2]; σz = PauliMatrix[3];
```
macierze Pauliego macierze Pauliego macierze Pauliego

In[16]:=
```
{{σx, σy, σz}} // MatrixForm
```
postać macierzy

Out[16]//MatrixForm=

$$\left(\begin{pmatrix}0 & 1\\ 1 & 0\end{pmatrix}\ \begin{pmatrix}0 & -i\\ i & 0\end{pmatrix}\ \begin{pmatrix}1 & 0\\ 0 & -1\end{pmatrix}\right)$$

Wyznaczamy współrzędne Px= , Py= , Pz= wektora P={Px,Py,Pz}. Zauważmy, że w tym celu działamy macierzą σy na wektor kolumnowy (ket) , tzn.wykonujemy operację: . W wyniku otrzymujemy pewien wektor kolumnowy. Następnie mnożymy ten wynik przez wektor wierszowy (bra) i w wyniku otrzymujemy skalar (liczbę). *Kropka "." oznacza w programie Mathematica mnożenia macierzy. Polecenie* **FullSimplify[]** *kontroluje uproszczenia algebraiczne a polecenie* **Flatten[]** *redukuje liczbę nawiasów klamrowych niepotrzebnych w dalszych obliczeniach , tj. „spłaszcza" strukturę tablic. Stosując funkcję* **Flatten[]** *należy oczywiście postępować ostrożnie, żeby przypadkowo nie przekształcić np. macierzy w wektor.*

In[17]:=
```
Px = Flatten[FullSimplify[⟨ψ0|.σx.|ψ0⟩]]
```
spłaszcz uprość pełniej

Out[17]= {Cos[ϕ] Sin[θ]}

In[18]:=
```
Py = Flatten[FullSimplify[⟨ψ0|.σy.|ψ0⟩]]
```
spłaszcz uprość pełniej

Out[18]= {Sin[θ] Sin[ϕ]}

In[19]:=
```
Pz = Flatten[FullSimplify[⟨ψ0|.σz.|ψ0⟩]]
```
spłaszcz uprość pełniej

Out[19]= {Cos[θ]}

Stosujemy operację transpozycji ^{T} (skrót Esc tr Esc), żeby zapisać wektor P w postaci wektora wierszowego. Zauważamy, że wektor P:

In[20]:=
```
P = {Px, Py, Pz}ᵀ
```

Out[20]= {{Cos[ϕ] Sin[θ], Sin[θ] Sin[ϕ], Cos[θ]}}

jest wektorem o współrzędnych rzeczywistych.

Obliczamy iloczyn skalarny wektora P z samym sobą, tzn. wyznaczamy kwadrat długości tego wektora.

In[21]:=
```
P.Pᵀ
```

Out[21]= $\{\{\mathrm{Cos}[\theta]^2 + \mathrm{Cos}[\phi]^2\,\mathrm{Sin}[\theta]^2 + \mathrm{Sin}[\theta]^2\,\mathrm{Sin}[\phi]^2\}\}$

Upraszczamy otrzymany wynik

In[22]:=
```
nP = Simplify[Sqrt[P.Pᵀ]]
```
uprość pierwiastek kwa

Out[22]= {{1}}

otrzymując wartość kwadratu długości wektora P równą nP=1.
Zatem została udowodniona teza 1. twierdzenia.

Krok 3. Definicja macierzy σP oraz rozwiązanie jej zagadnienia własnego (tzn. wyznaczanie wartości własnych i wektorów własnych) .

Definiujemy wektor kolumnowy, którego współrzędnymi są macierze Pauliego.

```
In[23]:= σ = {{σx}, {σy}, {σz}}
Out[23]= {{{{0, 1}, {1, 0}}}, {{{0, -i}, {i, 0}}}, {{{1, 0}, {0, -1}}}}
```

Jawną strukturę “wektora” macierzowego σ możemy zbadać korzystając z funkcji **MatrixForm** .

```
In[24]:= σ // MatrixForm
              postać macierzy
```

Out[24]//MatrixForm=

$$\begin{pmatrix} \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \\ \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix} \\ \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \end{pmatrix}$$

Pamiętając, że wektor P jest wektorem wierszowym wyznaczamy iloczyn skalarny P.σ w jawnej postaci.

```
In[25]:= P.σ // MatrixForm
                postać macierzy
```

Out[25]//MatrixForm=

$$\begin{pmatrix} \begin{pmatrix} \mathrm{Cos}[\theta] & \mathrm{Cos}[\phi]\,\mathrm{Sin}[\theta] - i\,\mathrm{Sin}[\theta]\,\mathrm{Sin}[\phi] \\ \mathrm{Cos}[\phi]\,\mathrm{Sin}[\theta] + i\,\mathrm{Sin}[\theta]\,\mathrm{Sin}[\phi] & -\mathrm{Cos}[\theta] \end{pmatrix} \end{pmatrix}$$

Korzystając z polecenia **Flatten{□,2]** usuwamy zewnętrzne nawiasy zbędne w dalszych obliczeniach, przy czym parametr „2” określa poziom “spłaszczenia” wyrażenia.

```
In[26]:= Flatten[P.σ, 2] // MatrixForm
         spłaszcz          postać macierzy
```

Out[26]//MatrixForm=

$$\begin{pmatrix} \mathrm{Cos}[\theta] & \mathrm{Cos}[\phi]\,\mathrm{Sin}[\theta] - i\,\mathrm{Sin}[\theta]\,\mathrm{Sin}[\phi] \\ \mathrm{Cos}[\phi]\,\mathrm{Sin}[\theta] + i\,\mathrm{Sin}[\theta]\,\mathrm{Sin}[\phi] & -\mathrm{Cos}[\theta] \end{pmatrix}$$

Po rozpoznaniu struktury wyrażenia P.σ definiujemy macierz σn. Funkcja FullSimplify[] dokonuje upraszczających przekształceń algebraicznych, w tym trygonometrycznych i zespolonych. np.: zamienia postać trygonometryczną liczby zespolonej Cos[ϕ]+i Sin[ϕ] na wykładniczą $e^{i\,\phi}$.

```
In[27]:= (σP = FullSimplify[Flatten[P.σ, 2]]) // MatrixForm
              uprość pełniej  spłaszcz             postać macierzy
```

Out[27]//MatrixForm=

$$\begin{pmatrix} \mathrm{Cos}[\theta] & e^{-i\,\phi}\,\mathrm{Sin}[\theta] \\ e^{i\,\phi}\,\mathrm{Sin}[\theta] & -\mathrm{Cos}[\theta] \end{pmatrix}$$

W tym momencie możemy przejść do wyznaczenia wartości własnych i wektorów własnych macierzy σP. *W programie* Mathematica *wartości własne i wektory własne wyznaczamy za pomocą funkcji* **Eigensystem[] .**

```
In[28]:= Eigensystem[σP]
         wartości i wektory własne
```

Out[28]= $\{\{-1, 1\}, \{\{e^{-i\,\phi}\,(\mathrm{Cot}[\theta] - \mathrm{Csc}[\theta]), 1\}, \{e^{-i\,\phi}\,(\mathrm{Cot}[\theta] + \mathrm{Csc}[\theta]), 1\}\}\}$

W wyrażeniu powyżej lista {-1,1} oznacza wartości własne macierzy σP a kolejne dwie listy zawierają wektory własne , które które kolejno odpowiadają wartościom własnym -1 i 1 . Zauważmy, że funkcja **Csc[]** (nie jest cosinus) jest odwrotnością funkcji **Sin[] .** Wynik można uprościć korzystając z funkcji **TrigFactor[]** , która przekształca wyrażania trygonometryczne do postaci iloczynowej. Uproszczoną postać przypisujemy do zmiennej **ww** .

```
In[29]:= (ww = TrigFactor[Eigensystem[σP]]) // MatrixForm
           faktoryzuj fu··· wartości i wektory własne   postać macierzy
```

Out[29]//MatrixForm=

$$\begin{pmatrix} -1 & 1 \\ \{-e^{-i\phi}\,\mathrm{Tan}[\frac{\theta}{2}],\,1\} & \{e^{-i\phi}\,\mathrm{Cot}[\frac{\theta}{2}],\,1\} \end{pmatrix}$$

Z drugiego wiersza otrzymanego wyrażania (**ww**) wybieramy drugi wektor własny, który odpowiada wartości własnej równej 1 i przedstawiamy go w postaci kolumnowego wektora ket **|ψn⟩** . *W tym celu podajemy indeksy wyrażenia, które chcemy wybrać:* **[[*i,j,k*]] ,** *gdzie* ***i*** *wskazuje wiersz,* ***j*** *kolumnę a* ***k*** *konkretną składową wybieranego wektora własnego.*

```
In[30]:= (|ψn⟩ = {{ww[[2, 2, 1]]}, {ww[[2, 2, 2]]}}) // MatrixForm
                                                       postać macierzy
```

Out[30]//MatrixForm=

$$\begin{pmatrix} e^{-i\phi}\,\mathrm{Cot}[\frac{\theta}{2}] \\ 1 \end{pmatrix}$$

Konstruujemy również wierszowy wektor dualny bra.

```
In[31]:= ⟨ψn| = Simplify[|ψn⟩†]
                 uprość
```

Out[31]= $\{\{e^{i\phi}\,\mathrm{Cot}[\frac{\theta}{2}],\,1\}\}$

Krok 4. Normowanie wektora własnego i dowód tezy 2. twierdzenia .

Wektory własne wyznaczane przez program Mathematica na ogół nie są unormowane. Teoria kwantowa wymaga by wektor opisujący stan kwantowe kubitu był unormowany do jedności [T-18 i T-19], gdyż kwadraty modułów współrzędnych są prawdopodobieństwem znalezienia kubitu w określonym stanie [T-17, T-19, T-20]. W celu unormowania wektora stanu obliczamy iloczyn skalarny ⟨ψn|.|ψn⟩ , tj. wyznaczamy kwadrat długości wektora **norma2**. Wyrażanie **[[**1,1**]]** pozwala usunąć niepotrzebne nawiasy i "wyłuskać" z nich liczbę, która stanowi szukana wartość iloczynu skalarnego.

```
In[32]:= norma2 = TrigFactor[((⟨ψn|.|ψn⟩)[[1, 1]])]
                  faktoryzuj funkcje trygonometryczne
```

Out[32]= $\mathrm{Csc}[\frac{\theta}{2}]^2$

Następnie wyznaczamy czynnik normujący **n** przez który należy pomnożyć normowany wektor aby jego długość po unormowaniu była równa jedności. *Przypisanie puste* **n=.** *usuwa wszelkie przypisanie do n nieporządne przy korzystaniu z funkcji* **Solve[]** *służącej do algebraicznego rozwiązywania równań.*

```
In[33]:= n =.; r = Solve[norma2 == 1/n^2, n]
                   rozwiąż równanie
```

Out[33]= $\{\{n \to -\mathrm{Sin}[\frac{\theta}{2}]\},\,\{n \to \mathrm{Sin}[\frac{\theta}{2}]\}\}$

Czynnik normujący jest określany z dokładnością do czynnika fazowego $e^{i\delta}$, gdzie δ może być dowolne. Czynnik $e^{i\delta}$ nie wpływa na wartość $\langle\psi|.|\psi\rangle$, w szczególności $e^{i\delta}$ może przyjąc wartość 1 lub -1. Ponadto czynnik $e^{i\delta}$ nie wpływa również na wartość $|\langle\phi\,|\,.\,|\,\psi\rangle|^2$ [T-17 i notatki odręczne na slajdach T-45 i T-48], tzn. $|\langle\phi\,|\,.\,|\,\psi\rangle|^2 = \left|\langle\phi\,|\,.e^{i\delta}\,|\,\psi\rangle\right|^2$, bo $|e^{i\delta}|$=1. Zatem $e^{i\delta}$ nie wpływa na prawdopodobieństwo otrzymywania wyników pomiarów typu X, Y i Z. Możemy więc przyjąć,

```
In[34]:= n = (ⅇ^(ⅈ ϕ) n /. r[[2, 1]])
```

```
Out[34]= ⅇ^(ⅈ ϕ) Sin[θ/2]
```

że czynnik normujący jest równy $e^{i\phi}\,\mathrm{Sin}\left[\frac{\theta}{2}\right]$ i ostatecznie wyznaczamy unormowany wektor $|\psi\rangle$ odpowiadający wartości własnej równej 1.

```
In[35]:= |ψ⟩ = n |ψn⟩
```

```
Out[35]= {{Cos[θ/2]}, {ⅇ^(ⅈ ϕ) Sin[θ/2]}}
```

```
In[36]:= |ψ⟩ // MatrixForm
                postać macierzy
```

Out[36]//MatrixForm=

$$\begin{pmatrix} \mathrm{Cos}\left[\frac{\theta}{2}\right] \\ e^{i\phi}\,\mathrm{Sin}\left[\frac{\theta}{2}\right] \end{pmatrix}$$

<u>Wektor jest zatem równy wektorowi co kończy dowód tezy 2. twierdzenia.</u>

Krok 5. Prawdopodobieństwo wyników pomiaru typu X. Dowód tezy 3- a).

Zdefiniujmy wektory (oznaczane w tutorialu [T-45] jako i) poprzez :

```
In[37]:= |0x⟩ = 1/√2 (|0⟩ + |1⟩); |1x⟩ = 1/√2 (|0⟩ - |1⟩);
```

W jawnej postaci można je zapisać jako $\begin{pmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{pmatrix}$ i $\begin{pmatrix} \frac{1}{\sqrt{2}} \\ -\frac{1}{\sqrt{2}} \end{pmatrix}$.

Są one ortonormalnymi (unormowanymi i ortogonalnymi) wektorami własnymi macierzy σx (oznaczanej także jako X) , co można sprawdzić wykonując obliczenia σx. , σx. , oraz , , . Pomiar typu X [T-45] sprawdza prawdopodobieństwo znalezienia kubitu w stanach: , . Prawdopodobieństwo to wyznaczamy korzystając z iloczynu skalarnego: px$[i_x]$= = , gdzie i=0 albo 1 . W programie Mathematica można to wyrazić poprzez zdefiniowaną funkcję (poniżej) , która zawiera również operacje upraszczające wyrażanie.

```
In[38]:= px[i_x] := Flatten[Simplify[((⟨ix|.|ψ0⟩))* ⟨ix|.|ψ0⟩]][[1]]
                    spłaszcz uprość
```

Obliczając wyrażanie różnicę prawdopodobieństw:

In[39]:= FullSimplify[px[0_x] - px[1_x]]
uprość pełniej

Out[39]= Cos[ϕ] Sin[θ]

stwierdzamy, że wartość px[0_x] – px[1_x] jest równa składowej Px wektora Blocha P. Kończy to dowód tezy 3-a) twierdzenia.

Krok 6. Prawdopodobieństwo wyników pomiaru typu Y. Dowód tezy 3- b).

Zdefiniujmy wektory (oznaczane w tutorialu [T-48] jako i) poprzez :

In[40]:= $|0_y\rangle = \frac{1}{\sqrt{2}}(|0\rangle + i\,|1\rangle)$; $|1_y\rangle = \frac{1}{\sqrt{2}}(|0\rangle - i\,|1\rangle)$;

W jawnej postaci można je zapisać jako $\begin{pmatrix} \frac{1}{\sqrt{2}} \\ i\,\frac{1}{\sqrt{2}} \end{pmatrix}$ i $\begin{pmatrix} \frac{1}{\sqrt{2}} \\ -i\,\frac{1}{\sqrt{2}} \end{pmatrix}$.

Tym razem są to ortonormalne wektory własne macierzy σy (oznaczanej także jako Y) , co można sprawdzić wykonując obliczenia analogicznie jak dla przypadku pomiaru typu X. Pomiar typu X [T-48] sprawdza prawdopodobieństwo znalezienia kubitu w stanach: , . Prawdopodobieństwo to wyznaczamy korzystając z iloczynu skalarnego: py[i_y] =

= , gdzie i=0 albo 1 .

In[41]:= py[i_y] := Flatten[Simplify[(⟨i_y|.|ψ_0⟩)* ⟨i_y|.|ψ_0⟩]][[1]]
spłaszcz uprość

Również tym razem wyznaczamy różnicę prawdopodobieństw.

In[42]:= FullSimplify[py[0_y] - py[1_y]]
uprość pełniej

Out[42]= Sin[θ] Sin[ϕ]

Stwierdzamy, że wartość py[0_y] – py[1_y] jest równa składowej Py wektora Blocha P. Kończy to dowód tezy 3-b) twierdzenia.

Krok 7. Prawdopodobieństwo wyników pomiaru typu Z. Dowód tezy 3- c).

Zdefiniujmy wektory (oznaczane w tutorialu [T-15] jako $|0\rangle$ i $|1\rangle$) poprzez :

In[43]:= $|0_z\rangle = |0\rangle$; $|1_z\rangle = |1\rangle$;

wektory te stanowią tzw. bazę obliczeniową.

Tym razem są to ortonormalne wektory własne macierzy σz (oznaczanej także jako Z) , co można sprawdzić wykonując obliczenia analogicznie jak dla przypadku pomiaru typu X. Pomiar typu X [T-51] sprawdza prawdopodobieństwo znalezienia kubitu w stanach: $|0_z\rangle$, $|1_z\rangle$. Prawdopodobieństwo to wyznaczamy korzystając z iloczynu skalarnego: pz[i_z] = $|\langle i_z|.|\psi_0\rangle|^2$

= $(\langle i_z|.|\psi_0\rangle)^* \langle i_z|.|\psi_0\rangle$ (i=0 albo 1) .

In[44]:= pz[i_z] := Flatten[Simplify[(⟨i_z|.|ψ_0⟩)* ⟨i_z|.|ψ_0⟩]][[1]]
spłaszcz uprość

Wyznaczamy również różnicę prawdopodobieństw.

In[45]:=
```
FullSimplify[pz[0z] - pz[1z]]
```
uprość pełniej

Out[45]= $\mathrm{Cos}[\theta]$

Stwierdzamy zatem że wartość $py[0_z] - py[1_z]$ jest równa składowej Pz wektora Blocha P. Kończy to dowód tezy 3-c) twierdzenia.

quod erat demonstrandum

III Praktyczny wniosek z twierdzenia .

1. Zauważmy, że wartości prawdopodobieństw $px[0_x]$, $px[1_x]$; $py[0_y]$, $py[1_y]$ oraz $pz[0_z]$, $pz[1_z]$ w komputerze kwantowym wyznaczamy z doświadczalnie. Tym samy możemy wyznaczyć eksperymentalne wartości:
$Px=px[0_x] - px[1_x]$, $Py=py[0_y] - py[1_y]$, $Pz=pz[0_z] - pz[1_z]$.
2. Dysponując wektorem P wyznaczonym z doświadczenia (*a posteriori*) możemy skonstruować wektor σP=Px.σx+Py.σy+Pz.σz a następnie wyznaczyć wartości własne i wektory własne korzystając z polecenia **(ww=TrigFactor[Eigensystem[σP]])//MatrixForm .**
3. Wybieramy i normujemy wektor własny macierzy σn odpowiadający wartości własnej równej +1. Wektor tej wyznacza stan kubitu przed pomiarem

IV Przykład empirycznego wyznaczania stanu kwantowego kubitu (tomografii kubitu).

Krok 1. Na podstawie wyników pomiarów X, Y, Z zamieszczonych w tutorialu [T-45, T-47 i T-50] zapisujemy doświadczalne wartości prawdopodobieństw (*a posteriori*) :

In[46]:=
```
pxd[0x] = 0.49219;
pxd[1x] = 0.50781;
pyd[0y] = 1;
pyd[1y] = 0;
pzd[0z] = 0.47656;
pzd[1z] = 0.52344;
```

a następnie wyznaczamy doświadczalny (*a posteriori*) wektor Blocha Pd.

In[47]:=
```
Pxd = pxd[0x] - pxd[1x]; Pyd = pyd[0y] - pyd[1y]; Pzd = pzd[0z] - pzd[1z];
Pdn = {Pxd, Pyd, Pzd}
```

Out[48]= {-0.01562, 1, -0.04688}

Krok 2. Ze względu na błędy (szumy) komputera kwantowego powodowane przez dekoherncję (tj. wpływ otoczenia na stan kubitu) długość doświadczalnego wektora Pdn nie jest równa 1. Dla tego trzeba wprowadzić korektę, tzn. unormować wektor Pdn do Pd o długości już równej jeden.
Udowodniliśmy, że wektor Blocha P jest ma współrzędne rzeczywiste.
Zatem obliczamy kwadrat długości wektora Pdn (iloczyn skalarny P.P).

In[49]:=
```
nPd2 = Pdn.Pdn
```

Out[49]= 1.00244

Następnie mnożymy wektor Pdn przez czynnik $\frac{1}{\sqrt{nPd2}}$ i uzyskujemy wektor Blocha po korekcie unormowany do jedności.

In[50]:=
```
Pd = 1/Sqrt[nPd2] Pdn
```

Out[50]= {-0.015601, 0.998781, -0.0468229}

Krok 3. Definiujemy doświadczalną macierz σP.

In[51]:=
```
(σPd = Flatten[Pd.σ, 1]) // MatrixForm
```
spłaszcz — postać macierzy

Out[51]//MatrixForm=
$$\begin{pmatrix} -0.0468229 + 0.\, i & -0.015601 - 0.998781\, i \\ -0.015601 + 0.998781\, i & 0.0468229 + 0.\, i \end{pmatrix}$$

Wyznaczamy wartości własne i wektory własne macierzy σP.

In[52]:=
```
(wwd = Eigensystem[σPd]) // MatrixForm
```
wartości i wektory własne — postać macierzy

Out[52]//MatrixForm=
$$\begin{pmatrix} -1. & 1. \\ \{0.0112993 + 0.723384\, i,\ 0.690354 + 0.\, i\} & \{-0.010782 - 0.69027\, i,\ 0.723472 + 0.\, i\} \end{pmatrix}$$

Krok 4. „Wyłuskujemy” wektor własny odpowiadający wartości własnej +1.

In[53]:=
```
(|ψnd⟩ = {{wwd[[2, 2, 1]]}, {wwd[[2, 2, 2]]}}) // MatrixForm
```
postać macierzy

Out[53]//MatrixForm=
$$\begin{pmatrix} -0.010782 - 0.69027\, i \\ 0.723472 + 0.\, i \end{pmatrix}$$

Krok 5. Normujemy wektor własny odpowiadający wartości własnej +1. Korzystamy z funkcji **Chop[]** , żeby bardzo małe wartości, bliskie 0 przybliżyć do zera.

In[54]:=
```
normad2 = (Flatten[Chop[⟨ψnd|.|ψnd⟩]][[1]])
```
spłaszcz — zamiana bardzo małej liczby na zero

Out[54]= 1.

Krok 6. Wyznaczamy unormowany wektor stanu

In[55]:=
```
(|ψud⟩ = 1/Sqrt[normad2] |ψnd⟩) // MatrixForm
```
postać macierzy

Out[55]//MatrixForm=
$$\begin{pmatrix} -0.010782 - 0.69027\, i \\ 0.723472 + 0.\, i \end{pmatrix}$$

Krok 7. Wyłączamy czynnik fazowy $e^{i\phi_0}$ przed nawias [T-18], żeby wektor stanu kubitu przedstawić w postaci $|\psi_0\rangle = e^{i\phi_0}\left(|C0|\ |0\rangle + e^{i(\phi_1-\phi_0)}\ |C1|\ |1\rangle\right)$. W tym celu „Wyłuskujemy” współczynniki C0 i C1 , fazy ϕ_0 i ϕ_1 , obliczamy fazę względną ϕw=$\phi_1 - \phi_0$ oraz moduły mC0=|C0| i mC1=|C1|.

In[56]:=
```
C0 = (|ψud⟩[[1, 1]]);
C1 = (|ψud⟩[[2, 1]]);
ϕ0 = Arg[C0];  ϕ1 = Arg[C1];
     argument liczby ⋯  argument lic
mC0 = Abs[C0];
      wartość bezwzględna
mC1 = Abs[C1];
      wartość bezwzględna
ϕw = ϕ1 - ϕ0
```

Out[57]= 1.58642

Krok 8. Ostatecznie zapisujemy postać wyznaczonego wektora opisującego stan kwantowy kubitu pomijając ogólny czynnik fazowy $e^{i\,\phi_0}$ wyłączony przez nawias. *Korzystamy z funkcji* **Defer[]** *, która odracza przekształcenia pozostawiając pierwotną formę wyrażenia*.

In[58]:=
```
|ψd⟩ = Chop[mC0 |0⟩ + e^(Defer[i] ϕw) mC1 |1⟩]
       zamiana bardzo małej liczby na zero
```

Out[58]= $\{\{0.690354\}, \{0.723472\, e^{1.58642\, i}\}\}$

In[59]:=
```
(|ψd⟩) // MatrixForm
           postać macierzy
```

Out[59]//MatrixForm=

$$\begin{pmatrix} 0.690354 \\ 0.723472\, e^{1.58642\, i} \end{pmatrix}$$

In[60]:=
$$\begin{pmatrix} 0.690353941572614\text{`} \\ 0.7234717930611779\text{`}\, e^{1.586415056635388\text{`}\, i} \end{pmatrix} \text{ // N // MatrixForm}$$

pr⋯ postać macierzy

Out[60]//MatrixForm=

$$\begin{pmatrix} 0.690354 \\ -0.0112993 + 0.723384\, i \end{pmatrix}$$

Otrzymany wynik kończy procedurę wyznaczania doświadczalnego (*a posteriori*) wektora stanu.

Krok 9. Sprawdzenie - porównanie z wynikiem teoretycznym (*a priori*) transformacji stanu przedstawionej na wykładzie za pomocą bramek H i U1[$\frac{\pi}{2}$] .

W tym celu obliczamy wyrażanie:

$$U1[\tfrac{\pi}{2}].H.|0\rangle = \begin{pmatrix} 1 & 0 \\ 0 & e^{i\left(\frac{\pi}{2}\right)} \end{pmatrix} \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} .$$

In[61]:=
```
(1 / Sqrt[2]) {{1, 0}, {0, Exp[I Pi / 2]}}.{{1, 1}, {1, -1}}.{1, 0} // N // MatrixForm
      pierwiastek kwadratowy   fu⋯ ⋅ pi                               pr⋯  postać macierzy
```

Out[61]//MatrixForm=

$$\begin{pmatrix} 0.707107 \\ 0. + 0.707107\, i \end{pmatrix}$$

Wniosek : zauważamy względną różnicę między wynikiem *a posteriori* a przewidywaniem *a priori* rzędu 2% .