AKADEMIA GÓRNICZO-HUTNICZA IM. STANISŁAWA STASZICA W KRAKOWIE

WYDZIAŁ FIZYKI I INFORMATYKI STOSOWANEJ

KATEDRA ODDZIAŁYWAŃ I DETEKCJI CZĄSTEK

Projekt dyplomowy

Generator Monte Carlo wiązki terapeutycznej w oparciu o charakterystykę przestrzeni fazowej IAEA

Monte Carlo therapeutic beam generator based on the IAEA phase space characteristic

Autor: Dawid Justyna

Kierunek studiów: Fizyka Medyczna

Opiekun pracy: dr inż. Bartłomiej Rachwał

Kraków, 2023

Chciałbym złożyć najszczersze podziękowania
opiekunowi projektu dr inż. Bartłomiejowi Rachwałowi.
Bardzo dziękuję za przekazaną wiedzę, poświęcony czas,
a także cenne wskazówki dotyczące pisania projektu.

Spis treści

1 Wprowadzenie

W dzisiejszych czasach radioterapia pełni ważną rolę w leczeniu nowotworów. Jej głównym celem jest napromieniowanie komórek nowotworowych odpowiednią dawką przekazaną przez promieniowanie jonizujące. Istnieje wiele metod stosowanych w radioterapii, obecnie najczęściej wykorzystywanymi są (1):

- **Intensity-modulated radiotherapy (IMRT)** - radioterapia z możliwością modulacji intensywności promieniowania
- **Volumetric Modulated Arc Therapy (VMAT)** - wielołukowa technika dynamiczna, która poprzez obrót ramienia (gantry) moduluje wiązkę, regulując jej moc w trakcie obrotu
- **Brachyterapia** - umieszczanie źródła w samym guzie lub w jego okolicy

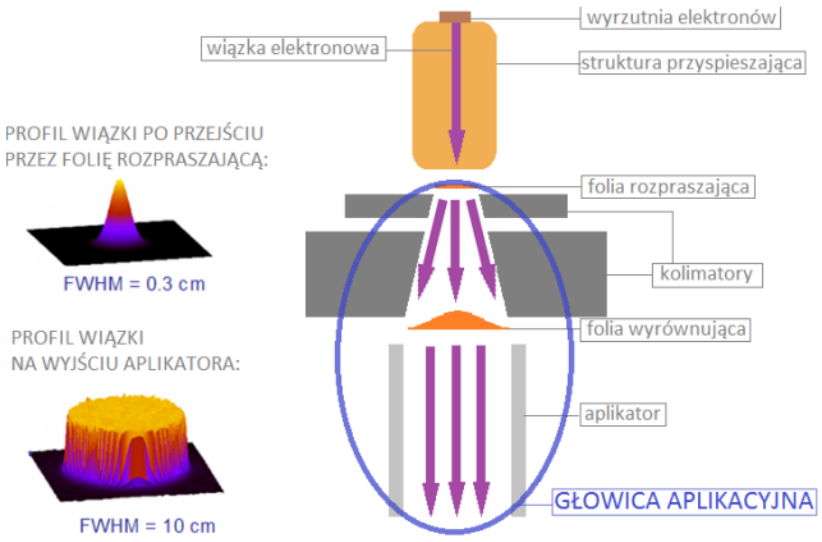
Każda z nich ma zalety i wady. Różnią się między innymi czasem trwania terapii, a także zastosowaniem w zależności od rodzaju nowotworu. Łączy je natomiast jedna ważna cecha, napromieniowanie jak największej części guza, przy jak najmniejszym obciążeniu dla sąsiadujących, zdrowych komórek. Jedną z głównych zasad stosowanych w radioterapii jest zasada - **ALARA (As Low As Reasonably Acievable)**, która zakłada, że dawki promieniowania należy utrzymywać na jak najniższym poziomie, o ile jest to możliwe ze względów bezpieczeństwa i skuteczności leczenia. Wraz z rozwojem technologii zaczęły pojawiać się modele przeprowadzające symulacje radioterapii. Są one często wykorzystywane w testowaniu różnych rozwiązań. Za pomocą modeli fizycznych, matematycznych i analitycznych można symulować zjawiska podczas napromieniowania pacjenta takie jak: rozkład dawki, skład wiązki promieniowania, kierunek propagacji poza polem naświetlania. Ciężko sobie wyobrazić prowadzenie jakichkolwiek badań, gdzie rozwiązanie byłoby testowane bezpośrednio na pacjentach, zamiast na napisanych wcześniej algorytmach.

Celem tego projektu było stworzenie modelu, który odtwarza statystykę fotonów do utworzenia wiązki terapeutycznej na podstawie istniejącej przestrzeni fazowej w formacie IAEA, umożliwiający prowadzenie symulacji rozkładu dawki na fantomach. Do otrzymania wyników przeprowadzone były symulacje Monte Carlo oraz transformacja Imana-Conovera. Następnie poprawność wygenerowanej wiązki sprawdzona była w programie PRIMO.

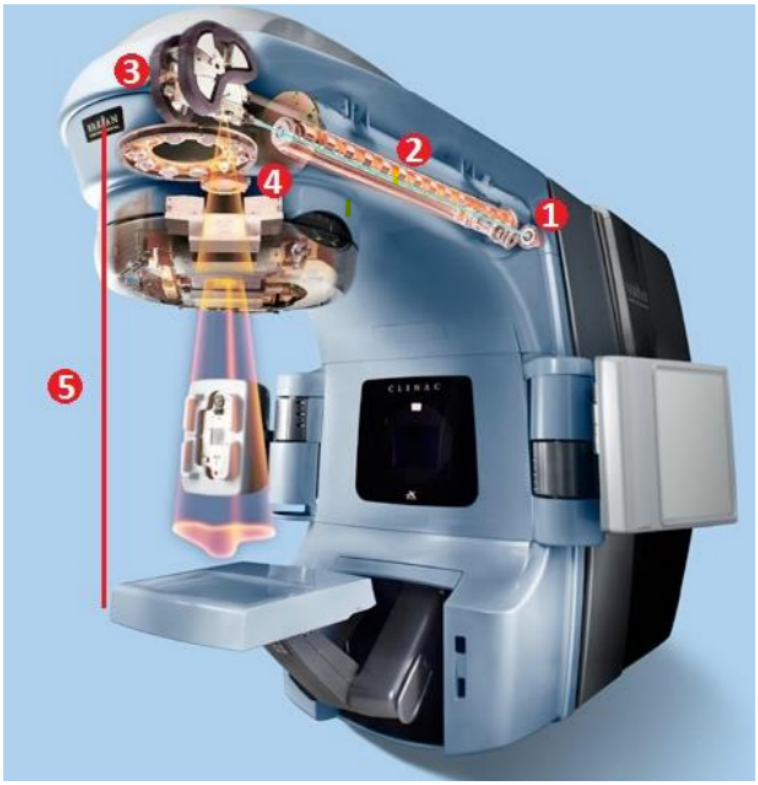
2 Wstęp teoretyczny

2.1 Akcelerator i jego budowa

Akcelerator to urządzenie, które służy do zwiększenia prędkości naładowanych cząstek do wartości zbliżonych do prędkości światła. Można wyróżnić dwa rodzaje akceleratorów: liniowe i kołowe. Te pierwsze wykorzystują pole elektryczne do przyspieszania cząstek i poruszają się one po torze prostoliniowym. Akcelerator kołowy wykorzystuje pole magnetyczne magnesów trwałych do zakrzywiania toru cząstek, a przyspieszane są one podobnie jak wcześniejszy polem elektrycznym. Obecnie w medycynie częściej spotyka się akceleratory liniowe. Po przyspieszeniu cząstek są one zakrzywiane i trafiają w układ dozymetryczny, gdzie wiązka jest dopasowywana do konkretnego pacjenta. Najpierw jest odpowiednio rozpraszana i kolimowana, a następnie formowana za pomocą filtra spłaszczającego (folia wyrównująca). (2)



Rysunek 1: Spłaszczanie wiązki [2]



Rysunek 2: Przykład medycznego akceleratora firmy Varian. 1 – źródło elektronów, 2 – struktura przyspieszająca, 3 – zakrzywianie wiązki, 4 – układ dozymetryczny, 5 – układ formowania wiązki terapeutycznej [2]

2.2 Przestrzeń fazowa akceleratora

Opisane wyżej powstawanie wiązki i jej parametry są kluczowe w radioterapii i to właśnie one odgrywają ważna rolę w symulacjach komputerowych. Międzynardowa Agencja Energii Atomowej (IAEA, International Atomic Energy Agency) opracowała i udostępniła niezależne orpogramowanie do zapisywania i odczytywania przestrzeni fazowych w formacie **.phsp**. Opisują one zbiór reprezentatywnych pseudocząstek powstających ze źródła radioterapii wraz z ich właściwościami. Charakteryzują się one następującymi parametrami (3):

- **X** - pierwsza współrzędna położenia (cm)
- **Y** - druga współrzędna położenia (cm)
- **Z** - trzecia współrzędna położenia (cm)
- **U** - cosinus pierwszego kierunku

- **V** - cosinus drugiego kierunku
- **E** - energia kinetyczna (MeV)
- **Particle_type** - typ cząsteczki: foton, elektron, pozyton, proton lub neutron

2.3 Metoda Monte Carlo

Metoda Monte Carlo to sposób na symulowanie skomplikowanych procesów, takich jak obliczanie całek czy generowanie losowych wartości według określonego rozkładu prawdopodobieństwa. Zamiast konkretnego algorytmu, opiera się ona na probabilistycznych właściwościach danego rozkładu. Niestety, ma ona kilka ograniczeń - jest stosowana tylko dla skończonej liczby prób i jej wynik jest jedynie statystycznym odzwierciedleniem prawdopodobieństwa, co oznacza, że nie jest dokładny. (4)

2.4 Korelacja i kowariancja

W statystyce istotne są takie pojęcia, jak kowariancja i korelacja.
Kowariancja jest miarą wspólnych zmian dwóch zmiennych losowych X i Y. Można ją wyrazić za pomocą następującego wzoru

$$cov_{X,Y} = \frac{\sum_{i=1}^{n}(X_i - \bar{X})(Y_i - \bar{Y})}{n-1} \tag{1}$$

gdzie:
$\bar{X}, \bar{Y}$ - średnie zmiennych X i Y
X_i, Y_i - wartość zmiennych X i Y
n - liczba wartości danych

Kowariancja informuje o relacjach między zmiennymi, nie wiadomo jednak jak silne one są. W tym celu oblicza się tak zwany współczynnik korelacji Pearsona.

$$cor_{X,Y} = \frac{cov_{X,Y}}{\sigma_x * \sigma_y} \tag{2}$$

gdzie:

σ_x, σ_y - odchylenia standardowe zmiennych X i Y

Wartości współczynnika korelacji mogą być z zakresu (-1,1).
Dla:

$$\begin{cases} cor_{X,Y} & = & 0 \ \ brak \ \ korelacji \\ cor_{X,Y} & > & 0 \ \ korelacja \ \ dodatnia \\ cor_{X,Y} & < & 0 \ \ korelacja \ \ ujemna \end{cases}$$

Im wartość jest bliżej 1 lub -1 tym korelacja jest silniejsza. W praktyce znając odchylenie standardowe zmiennej łatwo uzyskać z kowariancji korelację i na odwrót, dlatego też często pojęcia te są stosowane wymiennie.

Jeśli mamy do czynienia z więcej niż dwoma zmiennymi, można posługiwać się macierzą korelacji. Jest ona symetryczna, której elementami są współczynniki korelacji poszczególnych par zmiennych losowych. Dla zbioru X = ($x_1, x_2, ..., x_n$):

$$\begin{bmatrix} 1 & cor_{x_1,x_2} & \cdots & cor_{x_1,x_n} \\ cor_{x_2,x_1} & 1 & \cdots & cor_{x_2,x_n} \\ \vdots & \vdots & \ddots & \vdots \\ cor_{x_n,x_1} & cor_{x_n,x_2} & \cdots & 1 \end{bmatrix}$$

Podobnie wygląda macierz kowariancji. Na diagonali występują jedynki, a pozostałę komórki są wypełnione wartościami między poszczególnymi kowariancjami dwóch zmiennych. (5)

2.5 Rangowanie

Rangowanie danych jest to operacja, która przypisuje poszczególnym wartościom ze zbioru ich miejsce w szeregu tzn. ich dokładna wartość nie jest ważna. Istotne jest natomiast ile wartości jest mniejszych, a ile większych. Na tej podstawie przydziela się tak zwane rangi. Wyróżnić można kilka metod rangowania:

- **Prosta (ordinary)** - nadaje się rangi zgodnie z ich kolejnością występowania po posortowaniu. W przypadku takich samych wartości, przypisywanee są dwie kolejne liczby całkowite.

Wartości	Rangi
3.2	2
4.5	4
1.7	1
3.2	3

Tabela 1: Nadawanie rang metodą prostą

- **Ułamkowa (fractional)** - podobnie jak poprzednio nadawane są rangi zgodnie z kolejnością wystąpień. Jednak przy wystąpieniu takich samych wartości przypisywana jest im identyczne rangi, a ich wartość oblicza się ze średniej arytmetycznej tych rang z metody prostej.

Wartości	Rangi
3.2	2.5
4.5	4
1.7	1
3.2	2.5

Tabela 2: Nadawanie rang metodą ułamkową

- **Procentowa (percentage)** - w tym przypadku znajduje się maksymalną wartość i nadaje jej rangę 1, a dla pozostałych danych oblicza, jaką część stanowią maksymalnej wartości i taka ranga zostaje przypisana.

Wartości	Rangi
3	0.4
7.5	1
1.5	0.2
3	0.4

Tabela 3: Nadawanie rang metodą procentową

To tylko przykładowe metody rangowania. Każda z nich ma swoje zalety i ograniczenia. Warto wybierać je w zależności od konkretnych potrzeb. (6)

2.6 Rozkład Cholesky'ego

W matematyce można się spotkać z pojęciem faktoryzacji, czyli rozkładem na czynniki np. wielomianów czy macierzy. Tymi drugimi zajął się francuski matematyk Andre-Louis Cholesky. Opracował on metodę, która każdą symetryczną, dodatnie określoną macierz X, można rozłożyć na iloczyn dwóch macierzy:

$$X = L * L^T \tag{3}$$

gdzie:
L - macierz dolna trójkątna z rzeczywistymi i dodatnimi przekątnymi
L^T - transpozycja macierzy L

Dodatnia macierz trójkątna nazywana transformacją lub macierzą Choleskiego można przedstawić w następujący sposób:

$$\begin{bmatrix} L_{11} & 0 & \cdots & 0 \\ L_{21} & L_{22} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ L_{n1} & L_{n2} & \cdots & L_{nn} \end{bmatrix}$$

Komórki tej macierzy oblicza się używając odpowiedniego algorytmu (7):

$$\begin{cases} L_{i,j} = \sqrt{X_{i,j} - \sum_{k=1}^{j-1} L_{i,k}^2}, & dla \;\; i = j \\ L_{i,j} = \frac{1}{L_{i,j}}(X_{i,j} - \sum_{k=1}^{j-1} L_{i,k} * L_{j,k}), & dla \;\; i \neq j \end{cases}$$

gdzie:

i - numer wiersza

j - numer kolumny

$j = 1, 2, .., n$

$i = j, j + 1, ..., n$

Macierz Choleskiego może być używana do korelowania zmiennych. Jeśli mamy zestaw nieskorelowanych zmiennych i oczekiwaną macierz kowariancji, można wykonać następujące procedurę (8):

1. Należy wziąć nieskorelowane dane, zapisane w macierzy X, gdzie każda kolumna odpowiada jednej zmiennej.

2. Wykorzystując transformację Choleskiego wyznaczyć macierz trójkątną z macierzy kowariancji.

3. Obliczyć nowy zestaw danych za pomocą Y = X*L.

Otrzymana macierz Y, zawiera taką samą liczbę zmiennych i ich wartości co dane wejściowe X, a nowy zestaw danych jest odpowiednio skorelowany. Transformacja Choleskiego może być również używana do dekorelowania zmiennych. W tym przypadku algorytm postępowania jest podobny, jednak na wejściu przyjmowane są dane skorelowane, a zamiast macierzy L brana jest jej odwrotność L^{-1}. Dzięki tej metodzie można nie tylko nadawać określoną korelację zmiennym niezależnym, ale także zmieniać istniejącą korelację. Najpierw wykonuje się algorytm dekorelacji, a następnie na dane nakłada się oczekiwaną macierz kowariancji. Transformację Choleskiego można też wykorzystywać w innych celach, m.in do rozwiązywania układów równań, a także do pisania algorytmów uczenia maszynowego.

2.7 Transformacja Imana-Conovera

Korelacja zmiennych za pomocą macierzy Choleskiego pozwala uzyskać oczekiwane zależności między zmiennymi, ale algorytm ten zmienia również wartości wejściowe, co może być niepożądanym skutkiem. W 1982 roku amerykańscy matematycy Ronald L. Iman i William Jay Conover opracowali algorytm, który pozwala na korelację wejściowych danych bez zmiany ich wartości, jedynie zmieniając kolejność ich wystąpień. Algorytm oparty jest o następujące etapy (9):

1. Bierzemy nieskorelowane dane, zapisane w postaci macierzy X. W przypadku, gdy dane są w jakiś sposób skorelowane, dekorelujemy je np. stosując odwrotną macierz Choleskiego.

2. Nadajemy rangi, najlepiej gdyby one się nie powtarzały, zapewni nam to np. metoda prostą (ordinary).

3. Następnie normalizujemy wartości rang, to znaczy przesuwa się odpowiednio wektor rang i dzieli przez taką wartość, aby otrzymać rozkład normalny o średniej równej 0 i odchyleniu standardowym równym 1. (N)

4. Ustalamy macierz korelacji jaką chcemy otrzymać i obliczamy z niej macierz Choleskiego. (P)

5. Obliczamy macierz korelacji z danych wejściowych, powinny być one nieskorelowane, zatem powinniśmy otrzymać macierz jednostkową, a następnie obliczamy z niej macierz Choleskiego Q.

6. Obliczamy macierz nowych skorelowanych danych Y za pomocą wzoru:

$$Y = N * (P * Q^{-1})^T \tag{4}$$

7. Nadajemy rangi nowym danym z macierzy Y, także metodą prostą.

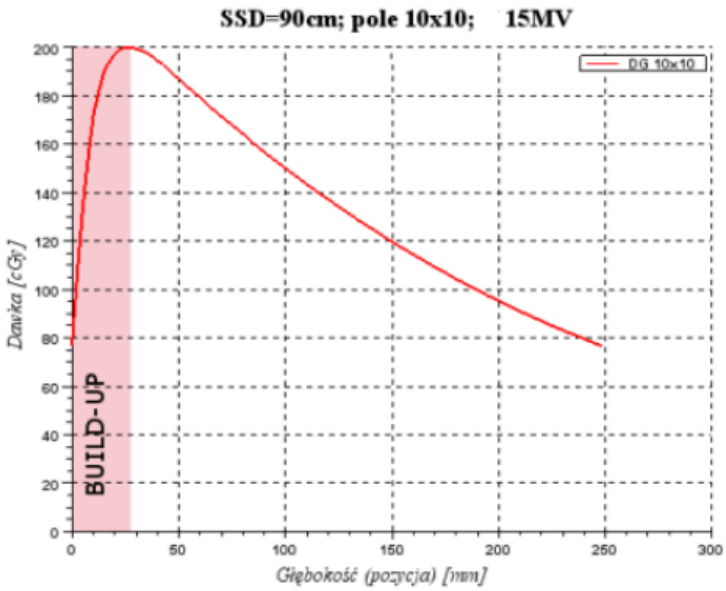
8. Ostatnim zadaniem jest permutacja danych wybranej zmiennej w taki sposób, że bierzemy rangę dla kolejnych wartości z macierzy Y. następnie odszukujemy ją w danych wejściowych i odczytujemy jaka wartość kryła się pod tą rangą i przypisujemy ją do kolejnych wartości zmiennych.

Po wykonaniu tych kroków otrzymane są nowe, skorelowane zmienne zapisane w macierzy Y. Zostają również zachowane rozkłady statystyczne, ponieważ wejściowe wartości nie zostały zmodyfikowane, a jedynie zamienione kolejnością wystąpień. Jest to znaczna przewaga nad korelacją za pomocą macierzy Choleskiego. (10)

2.8 Charakterystyka wiązek fotonowych w radioterapii

Do celów walidacji uzyskiwanych wyników, w projekcie wykorzystano informację na temat rozkładów dawki w fantomie. Opisuje się je następującymi parametrami:

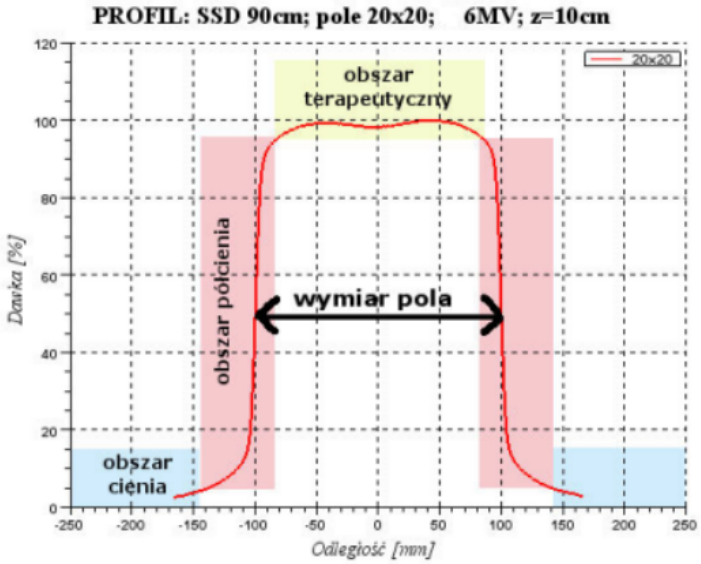
1. **Profil głębokościowy** - opisuje on jaką dawkę otrzymuje pacjent (fantom) wraz z wnikaniem wiązki wgłąb ciała



Rysunek 3: Przykład rozkładu dawki w profilu głębokościowym [11]

Na początku można zauważyć tzw. obszar narastania dawki. Trwa on aż do osiągnięcia maksimum. Przeważnie to właśnie tam zlokalizowany jest nowotwór. Kształt rozkładu dawki jest zawsze taki sam. Można go regulować np. zmieniając miejsce wystąpienia maksima różnymi technikami m.in. zmianą energii promieniowania.

2. **Profil poprzeczny** - opisuje rozkład dawki w płaszczyźnie prostopadłej do wiązki



Rysunek 4: Przykład rozkładu dawki w profilu poprzecznym [11]

Podobnie jak w profilu głębokościowym kształt jest zawsze podobny. Można wyróżnić poszczególne obszary: terapeutyczny (strefa gdzie dawka jest w granicy 90%-100% dawki względnej), półcienia (dawka między 20%-80%). Wymiar pola określa się od miejsca gdzie dawka osiąga poziom 50%. (11)

3 Wykorzystane technologie

3.1 Python

Projekt inżynierski został napisany w języku Python w środowisku Jupyter, co przynosi wiele korzyści, zwłaszcza przy pracy z większymi zestawami danych. Notatnik Jupyter umożliwia łatwą eksplorację, czyszczenie i przekształcanie danych, a także ich wizualizację. Jest to bardzo wygodne, ponieważ pozwala na wykonywanie pojedynczych fragmentów kodu źródłowego, co znacznie przyspiesza działanie programu. Jupyter umożliwia również pisanie kodu w chmurze, co pozwala kilku osobom na pracę nad jednym notatnikiem jednocześnie za pomocą narzędzia Google Colab (12). Python posiada wiele bibliotek przydatnych w analizie danych, a niektóre z nich zostały wykorzystane w tym projekcie:

- **NumPy** - jest to pakiet wykorzystywany do obliczeń naukowych. Zapewnia wiele operacji na macierzach m.in. transpozycje, dodawanie, mnożenie przez skalar, wektor, czy też mnożenie macierzy z inną macierzą. Po za tym umożliwia obliczanie macierzy odwrotnej, korelacji czy kowariancji. (13)
- **Pandas** - biblioteka wykorzystywana do analizy danych. W łatwy sposób można wyszukiwać w zbiorze danych parametry takie jak: minimum, maksimum, średnia, odchylenie standardowe itp. Pozwala również na selekcjonowanie, czy usuwanie danych. (14)
- **Seaborn** - biblioteka do wizualizacji danych np. do tworzenia wykresów, histogramów. (15)
- **SciPy** - biblioteka zawierająca funkcje statystyczne. W projekcie wykorzystana została jedna z nich do rangowania danych. (16)

3.2 ROOT

ROOT jest frameworkiem stworzonym w CERN. Stosowany głównie do analizy danych dla badań nad fizyką wysokich energii. Zapewnia integrację między innymi językami programowania takimi jak: Python, C++, czy R. W projekcie

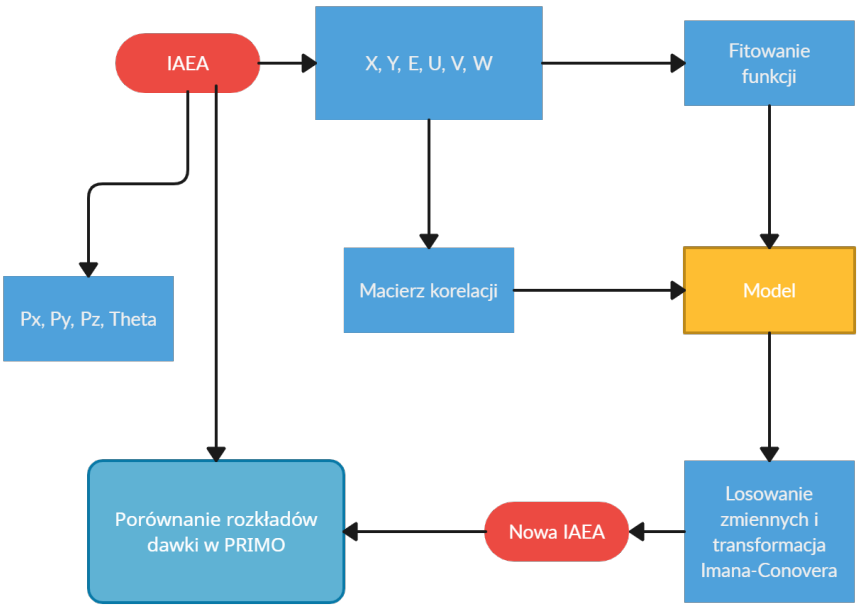
głównym zastosowaniem ROOT'a jest fitowanie funkcji prawdopodobieństwa do histogramów, a także wizualizacja ich. (17)

3.3 Primo

Primo to oprogramowanie komputerowe, które symuluje rozkłady dawek w fantomach lub z tomografii komputerowej w formacie DICOM. Posiada graficzny interfejs, który pozwala użytkownikowi na ustawianie poszczególnych parametrów takich jak: liczba cząstek, wymiary fantomu i jego materiał, a także na wybranie modelu akceleratora, na którym ma być przeprowadzana symulacja. Dostępne są akceleratory firmy Varian. (18)

4 Implementacja

Do realizacji projektu wykorzystana została przestrzeń fazowa, która zawierała historię cząstek w polu promieniowania 50mm x 50mm, energię średnią 0.05 keV i energię maksymalną 6.4 keV. Skrócony przebieg analizy pokazuję rysunek 5



Rysunek 5: Graf przedstawiający schemat postępowania

4.1 Eksploracja przestrzeni fazowych

Sprawdzono jak wygląda zestaw początkowy zestaw danych z wczytanej przestrzeni fazowej w Colabie.

	EvtId	X	Y	Z	E	Ekin	Px	Py	Pz	U	V	W	Theta	Particle
0	4	-13.8144	-25.5078	354.6	1.5559	1.5559	-0.0840	-0.1591	1.5454	-0.0540	-0.1023	0.9933	6.6443	gamma
1	5	-10.0695	-10.8014	354.6	2.6824	2.6824	-0.1146	-0.1096	2.6777	-0.0427	-0.0409	0.9983	3.3914	gamma
2	4	21.2358	-2.6387	354.6	0.5110	0.5110	0.3346	-0.2952	0.2491	0.6548	-0.5777	0.4874	60.8600	gamma
3	10	39.0925	-10.2528	354.6	0.6135	0.6135	0.0916	-0.0248	0.6061	0.1493	-0.0404	0.9880	8.9008	gamma
4	6	17.7997	-11.2071	354.6	2.1663	2.1663	0.1485	-0.0976	2.1590	0.0686	-0.0451	0.9966	4.7088	gamma
5	8	36.0283	3.8610	354.6	1.5647	1.5647	0.2230	0.0232	1.5486	0.1425	0.0148	0.9897	8.2423	gamma
6	10	-47.8256	-4.0063	354.6	1.0305	1.0305	-0.1860	-0.0172	1.0134	-0.1805	-0.0167	0.9834	10.4505	gamma
7	9	15.7277	-33.0124	354.6	1.2245	1.2245	0.0691	-0.1638	1.2115	0.0565	-0.1338	0.9894	8.3545	gamma

Rysunek 6: Fragment danych wejściowych

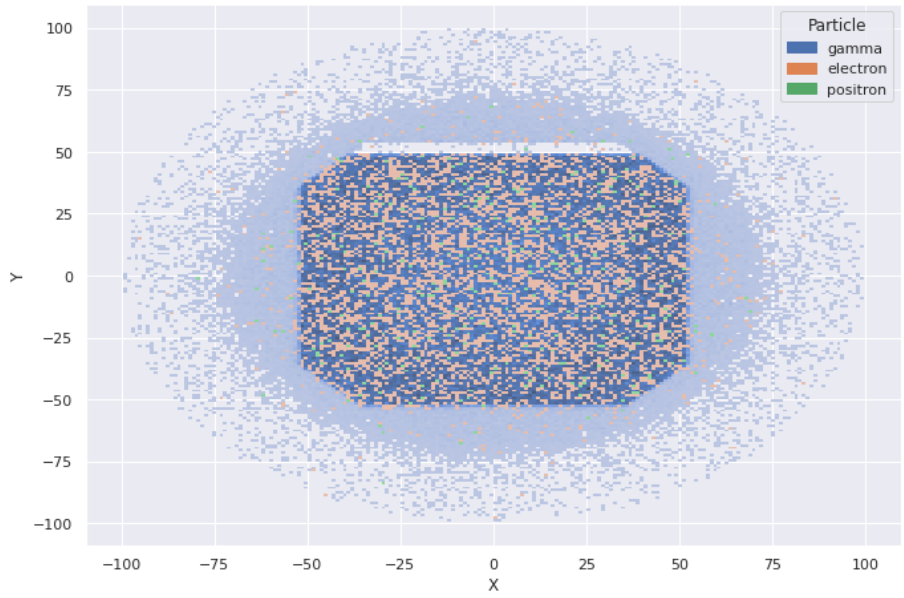
Powyższy fragment przedstawia zmienne charakteryzujące fotony w zapisa-

nej statystyce. Niektóre z nich (X, Y, Z, E, U, V, W, Particle) były wcześniej opisane zgodnie z dokumentacją IAEA. Pozostałe wyznaczane za pomocą tych wcześniejszych, istotne z punktu widzenia wykonywania symulacji Monte Carlo mają następujące oznaczenia:

- **EvtId** - nr eventu(historii) z której pochodzi dany foton
- **Ekin** - energia kinetyczna (Mev)
- **Px** - wartość pędu w kierunku wektora położenia X ($\frac{MeV}{c}$)
- **Py** - wartość pędu w kierunku wektora położenia Y ($\frac{MeV}{c}$)
- **Pz** - wartość pędu w kierunku wektora położenia Z ($\frac{MeV}{c}$)
- **Theta** - kąt zawarty między kierunkiem pędu cząstki i osią wiązki (osią Z)

4.1.1 Filtracja danych wejściowych

Do stworzenia modelu należało odfiltrować niepożądane dane.



Rysunek 7: Występowanie poszczególnych rodzajów cząstek w przestrzeni fazowej

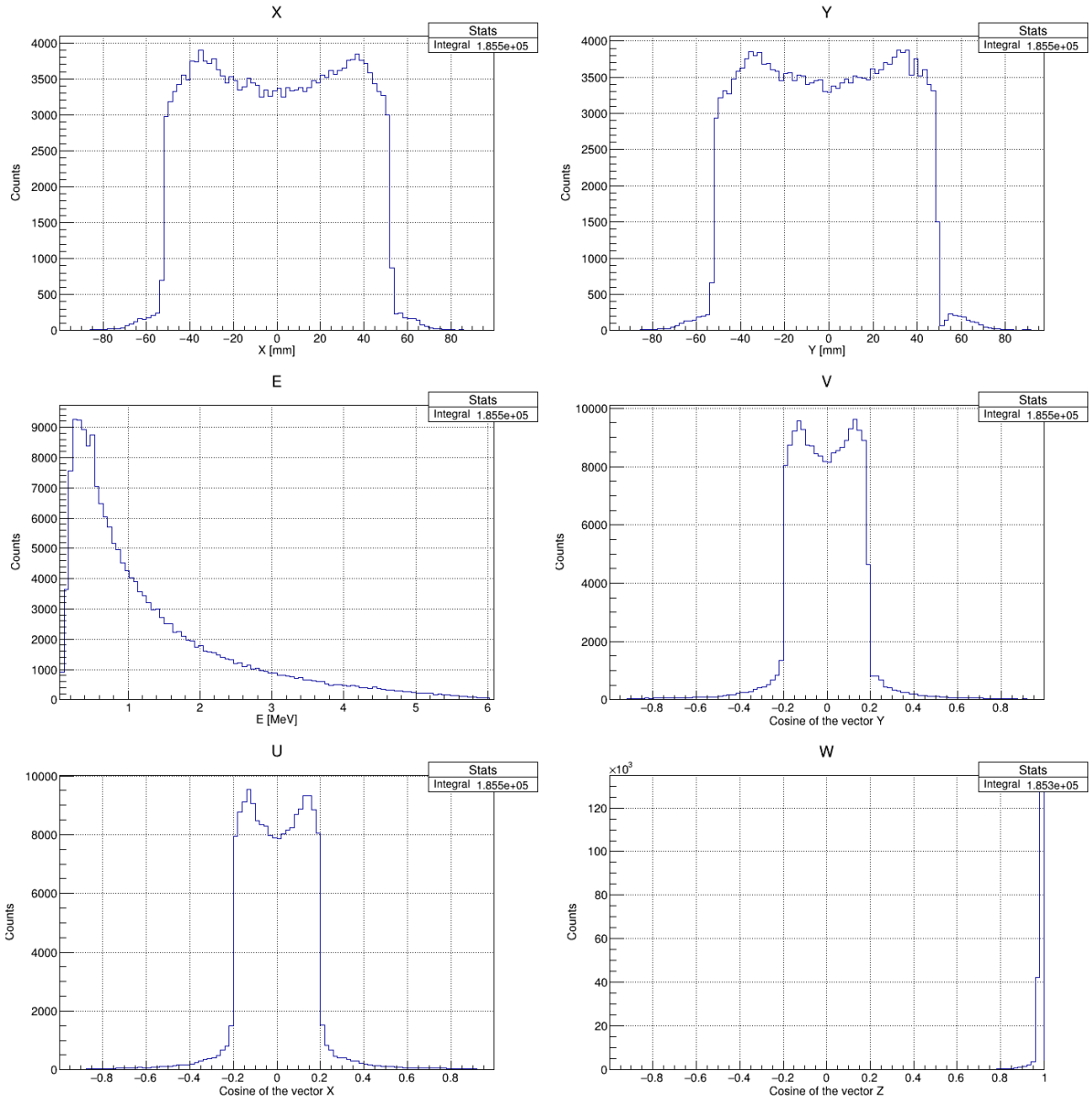
Pokazano wyżej rozkład rys. 7 położenia cząstek dla współrzędnej X i Y z wyszczególnieniem na na ich rodzaj (foton, elektron, pozyton). Jak widać fotony obejmują przeważającą część przestrzeni fazowej, dokładnie 99.46%, elek-

trony 0.5%, pozytony 0.03%. Do dalszej analizy została przeprowadzona wstępna filtracja danych. Usunięte zostały elektrony i pozytony, a także aby uzyskać zbiór niezależnych statystycznie fotonów, wybrano po jednej wartości z każdego eventu EvtID (w wyjściowej przestrzeni fazowej cząstki o takiej samej wartości EvtId pochodzą z tej samej historii, dlatego mogą być skorelowane).

```
data = data[data["Particle"]=="gamma"]
data = data.drop_duplicates(subset=["EvtId"])
```

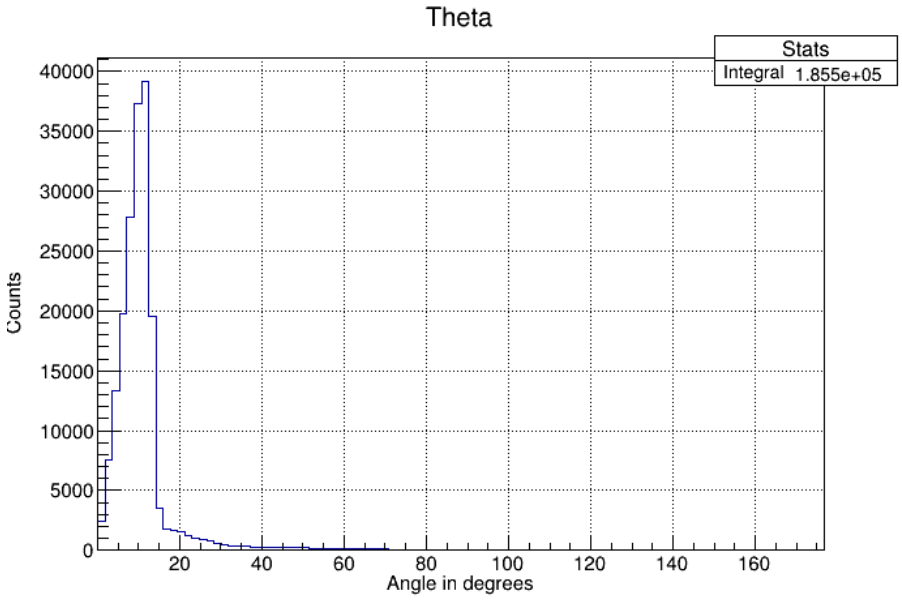
Listing 1: Preselekcja danych: redukcja wariancji i usuwanie cząstek innych niż gamma

Po wykonaniu operacji z listingu 1 została ok. połowa danych wyjściowych, głównie za sprawą usuwania fotonów pochodzącego z tego samego eventu. Sprawdzono jak wyglądają histogramy zmiennych, które wykorzystano do stworzenia modelu: X, Y, E, W, U, V.



Rysunek 8: Rozkłady wejściowe dla fotonów, przy statystyce 185500 i liczbie binów 100

Jak widać na rysunku 8, niektóre fotony ulegają rozproszeniu, gdzie np. pojedyncze wartości składowej X lub Y są w granicach 80 mm. Nie mają one istotnego wpływu na analizę, a poszerzały zakres histogramów. Oczyszczenie ich umożliwiło dokładniejsze fitowanie funkcji statystycznych do rozkładów. W tym celu wykorzystano zmienną Theta rys 9, która nie bierze udziału w tworzeniu modelu, ale w nadawaniu ograniczeń.



Rysunek 9: Zmienna Theta w rozkładzie wyjściowym. Statystyka 185500, liczba binów 100

Po analizie zdecydowano się na ograniczenia jak w Listingu 2

```
data = data[data["Theta"]<15]
data = data[data["X"].between(-55,55)]
data = data[data["Y"].between(-55,55)]
```

Listing 2: Preselekcja danych

Po tej selekcji zostało usunięte ok. 10% fotonów.

4.1.2 Fitowanie funkcji statystycznych do rozkładów

Do oczyszczonych danych za pomocą biblioteki ROOT dofitowano funkcje opisujące poszczególne rozkłady parametryczne zmiennych. Wykorzystano następujące funkcje lub ich połączenia: wielomiany, rozkład Gaussa i Landau. Przed fitowaniem ustalono dwa parametry, liczbę fotonów, a także liczbą przedziałów w histogramie. W pierwszym przypadku po obserwacji ustalono, że dla 2 milionów danych wejściowych, po odfiltrowaniu zostanie 800 tysięcy fotonów i była to wystarczająca liczba do tworzenia modelu. Do ustalania liczby przedziałów próbowano korzystać z wzoru:

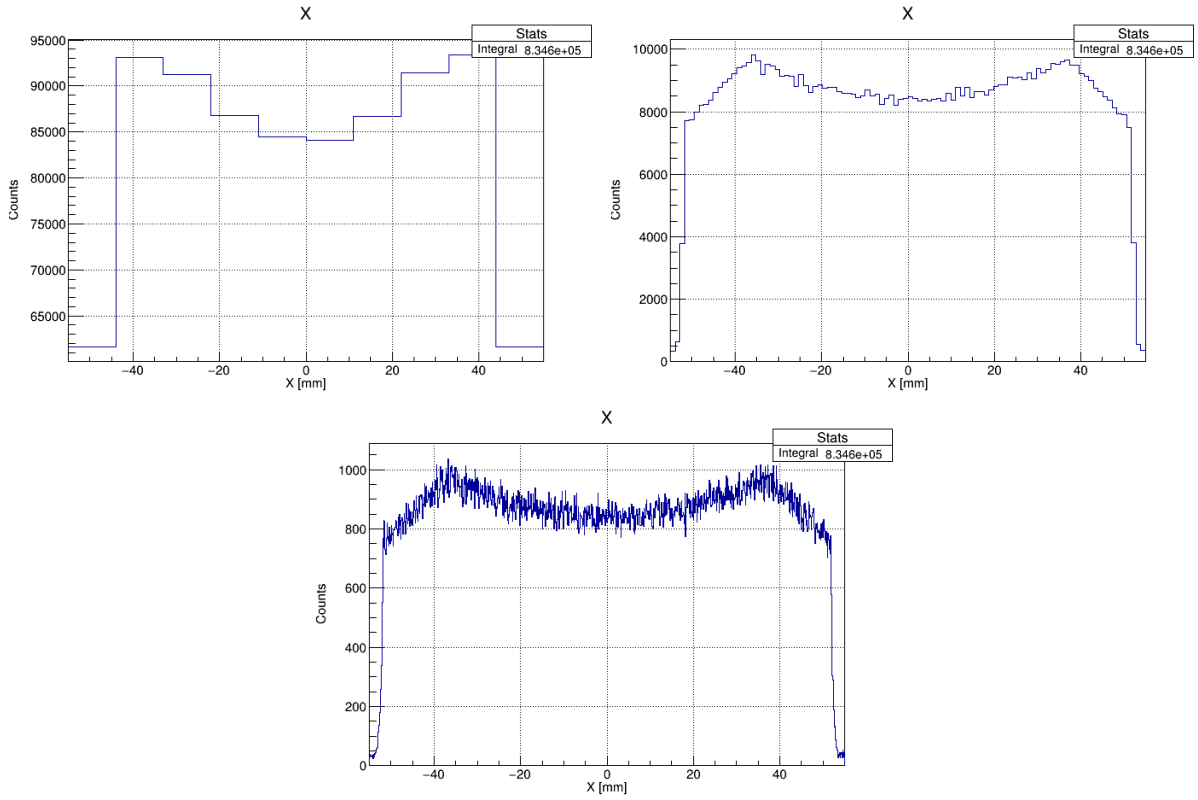
$$k \approx \sqrt{N} \tag{5}$$

gdzie:

k - liczba przedziałów

N - liczba danych

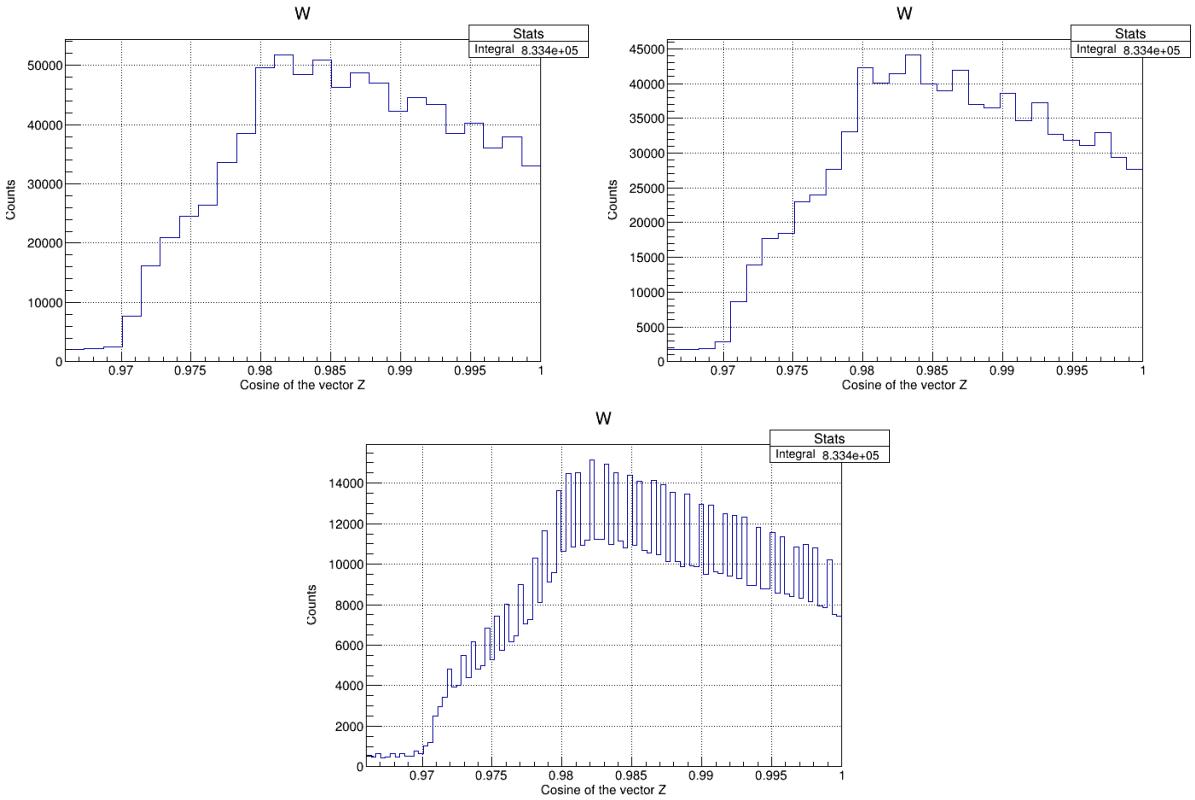
W tym przypadku było by to około 890 przedziałów. Nie jest to jednak wartość ściśle określona, dlatego przetestowano kilka wariantów m.in. dla 10, 100 i 1000 przedziałów.



Rysunek 10: Sprawdzenie jak zachowują się histogramy dla odpowiednio 10, 100, 1000 binów

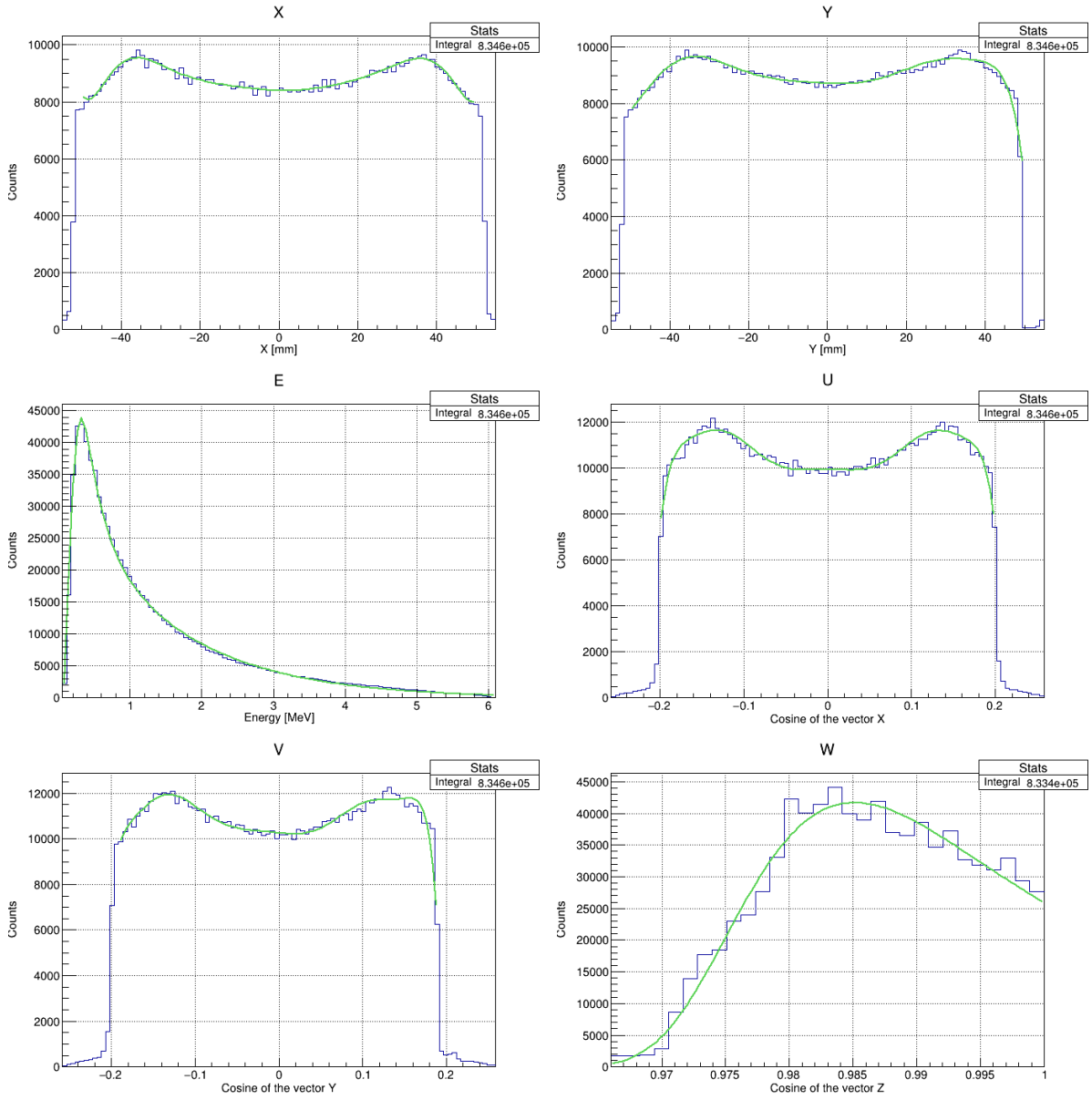
Jak widać na rysunku 10 dla 100 binów histogram jest najczytelniejszy. Nie zawiera pojedynczych skoków wartości, jak to jest w przypadku 1000 przedziałów. Przy 10 binach pomijane są niektóre szczegóły, takie jak np. fragmenty narastania między 40 a 50 mm. Zdecydowano się zatem na rysowanie histogramów dla 100 binów. W przypadku pozostałych zmiennych obserwacje były podob-

ne. Wyjątek stanowiła zmienna W, gdzie histogramy prezentowały się jak na rysunku 11:



Rysunek 11: Sprawdzenie jak zachowują się histogramy dla zmiennej W odpowiednio 25, 30, 100 binów

W tym przypadku dla 100 binów można zaobserwować duże różnicę między sąsiadującymi przedziałami. Przyczyną może być bardzo mała rozbieżność między minimalna a maksymalną wartością na histogramie. Jest to różnica około 0.04. W przypadku 100 binów, szerokość przedziałów wynosi 0.0004 co odpowiada rzędem wielkości do dokładności pomiaru zmiennej W. Do tworzenia modelu wybrano 30 binów.



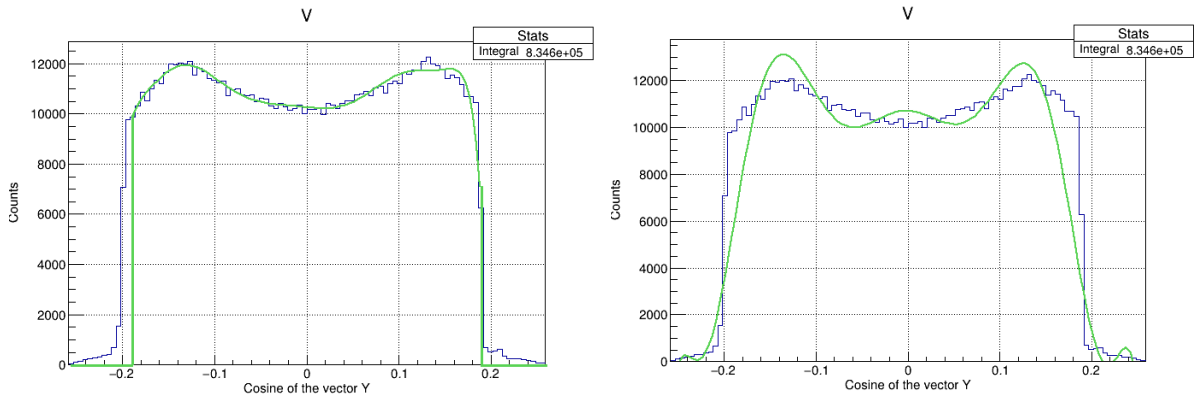
Rysunek 12: Fitowanie rozkładów

Dofitowane funkcje zostały zapisane jako obiekty ROOT. Domyślnie są one dopasowane do liczby zliczeń, jednak można je łatwy sposób normalizować i wtedy dają informację na temat gęstości prawdopodobieństwa w danym punkcie. Zmienne fitowane były następującymi funkcjami lub sumami funkcji:

- **X** - wielomian 10 stopnia
- **Y** - wielomian 10 stopnia
- **W** - Landau + Gaus + Gaus

- **V** - wielomian 10 stopnia
- **U** - wielomian 10 stopnia
- **E** - Landau + Gaus + Gaus

Na niektórych wykresach (X, Y, U, V) można zauważyć, że są fragmenty histogramu, gdzie nie ma dofitowanej funkcji. Jest to spowodowane tym, że są tam gwałtowne spadki częstości wystąpień i funkcje analityczne nie były sobie w stanie z tym poradzić. Zdecydowano się na ograniczenie dziedziny funkcji fitującej w tych przypadkach. Na rysunku 13 porównano dwa rozkłady ze różnymi dziedzinami.



Rysunek 13: Porównanie tej samej funkcji fitującej dla różnych dziedzin na przykładzie zmiennej V

Jak widać na rysunku 13 po lewej stronie jest histogram z ograniczoną dziedziną funkcji. Wszytsko co jest po za nim, może wystąpić z prawdopodobieństwem równym 0. Po prawej natomiast funkcja fituje na całej dziedzinie zmiennej V. Zdecydowanie lepszy efekt przynosi pierwsza z nich, zwłaszcza w miejscach, gdzie liczba zliczeń jest największa.

4.2 Model

Wcześniej opisane wyniki zostały zapisane w modelu posiadającym informację na temat fotonów, które będą odtwarzane. W jego zawartości znajdują się fitowane funkcję jako obiekty TH1 z biblioteki ROOT. Model przechowuje rów-

nież informację na temat macierzy korelacji zmiennych zapisanych jako obiekt TMatrixD, również z biblioteki ROOT.

```
def create_model(function, correlate_matrix):
  n = len(correlate_matrix)
  myfile = TFile.Open("model.root","RECREATE")
  myfile.WriteObject(function[0], "X")
  myfile.WriteObject(function[1], "Y")
  myfile.WriteObject(function[2], "E")
  myfile.WriteObject(function[3], "W")
  myfile.WriteObject(function[4], "V")
  myfile.WriteObject(function[5], "U")
  correlated = TMatrixD(n)
  for i in range(n):
    for j in range(n):
      correlated[i][j] = correlate_matrix[i][j]
  myfile.WriteObject(correlated, "correlate")
```

Listing 3: Tworzenie modelu

Funkcja tworząca model przyjmuje dwa argumenty. Pierwszym z nich jest lista funkcji fitujących rozkłady statystyczne. Drugim argumentem jest macierz korelacji zmiennych, które będą wykorzystywane w modelu, czyli dokładnie tych samych, do których były fitowane funkcje.

Funkcja przedstawiona na listingu 4 odczytuje informacje z nowego modelu i zwraca macierz korelacji w obiekcie DataFrame z biblioteki pandas. Drugą wartością zwracaną przez funkcję jest tablica zawierająca wszystkie fitowane rozkłady.

```
def read_model(model_name = "model.root"):
  model = TFile.Open(model_name)
  model_X = model.X
  model_Y = model.Y
  model_W = model.W
  model_U = model.U
  model_V = model.V
  model_E = model.E
  model_correlate = model.correlate
  cols =  ["X", "Y", "E", "W" ,"U", "V"]
  n = len(cols)
  cor_mat = [[ 0 for _ in range(n)] for _ in range(n)]
  for i in range(n):
    for j in range(n):
      cor_mat[i][j] =  model_correlate[i][j]
  cor_mat = pd.DataFrame(data  = cor_mat , columns=cols)
  return cor_mat, [model_X, model_Y, model_W,
                   model_U, model_V, model_E]
```

Listing 4: Wczytywanie modelu

4.3 Generowanie fotonów

W listingu 5 opisana jest procedura losowania nowych parametrów fotonów. Wykorzystywana jest w niej funkcja z biblioteki root GetRandom(), która znając rozkłady gęstości prawdopodobieństwa losuje wartości z osi X danego rozkładu. Szansa na wylosowanie danego punktu jest tym większa im wyższa była wartość dla niego w fitowanej funkcji. Przyjmowany jest argument, który pozwala zdecydować ile cząstek ma być wygenerowanych. Dla każdego fotonu osobno losowany jest każdy z parametrów (X, Y, E, W, U, V) niezależnie od siebie.

```
def generate_photons(fit_function, stats):
  X, Y, E, W, V, U = [], [],[], [], [], []
  cols =  ["X", "Y","E","U","V", "W"]
  n = len(cols)
  gRandom.SetSeed()
  for i in range(stats):
    X.append(fit_function[0].GetRandom())
    Y.append(fit_function[1].GetRandom())
    E.append(fit_function[2].GetRandom())
    U.append(fit_function[5].GetRandom())
    V.append(fit_function[4].GetRandom())
    W.append(fit_function[3].GetRandom())
  rand_photons = np.vstack((X, Y, E, U, V, W))
  rand_photons = pd.DataFrame(data  = rand_photons.T,
                              columns=cols)
  return rand_photons
```

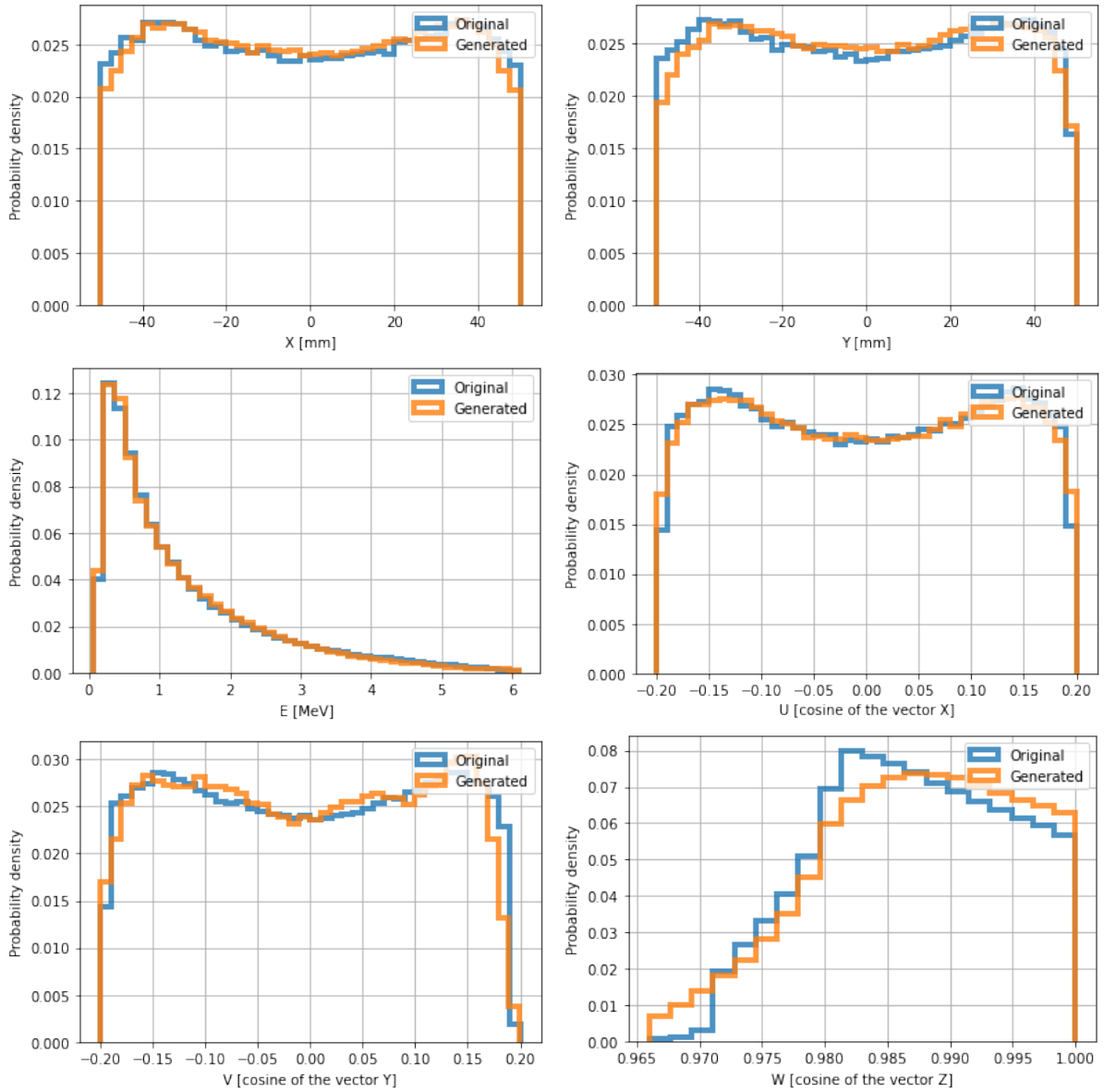
Listing 5: Generowanie nowych niezależnych fotonów

Sprawdzono jak wyglądają poszczególne zmienne względem zmiennych wejściowych, a także macierz korelacji.

	X	Y	E	U	V	W
X	1.0000	-0.0000	0.0007	-0.0009	-0.0029	-0.0010
Y	-0.0000	1.0000	-0.0008	0.0017	0.0023	-0.0003
E	0.0007	-0.0008	1.0000	0.0016	0.0035	0.0000
U	-0.0009	0.0017	0.0016	1.0000	-0.0012	-0.0003
V	-0.0029	0.0023	0.0035	-0.0012	1.0000	-0.0005
W	-0.0010	-0.0003	0.0000	-0.0003	-0.0005	1.0000

Rysunek 14: Macierz korelacji dla nowo wygenerowanych fotonów

Jak widać na rysunku 14, korelacja między poszczególnymi zmiennymi jest bliska 0 zgodnie z oczekiwaniami, ponieważ każdy parametr fotony był losowany niezależnie od pozostałych.



Rysunek 15: Porównanie wylosowanych parametrów fotonów z wejściowymi

Jak widać na rysunku 15 histogramy przypominają kształtem rozkłady wyjściowe. Kolejną częścią projektu było odwzorowanie korelacji między poszczególnymi parametrami fotonów. Do tego wykorzystany został algorytm Imana-Conovera którego implementacja zawarta jest w listingu nr 6 (19), (20)

```
def imanConover(data, corrmat,
                        cols = ["X", "Y","E","U","V", "W"]):
    X = data.to_numpy()
    X_rank = np.vstack([rankdata(datai, method="ordinal")
                                for datai in X.T]).T
    size = len(X_rank)
    X_rank_score = X_rank / (size + 1)
    X_rank_score = norm(0, 1).ppf(X_rank_score)
    P = np.linalg.cholesky(corrmat)
    corr_mat_X_rank = np.corrcoef(X_rank_score, rowvar=0)
    Q = np.linalg.cholesky(corr_mat_X_rank)
    Y = np.dot(P, np.linalg.inv(Q))
    new_data = np.dot(X_rank_score, Y.T)
    new_data_rank = np.vstack([rankdata(datai,
                                method="ordinal")
                                for datai in new_data.T]).T
    new = new_data_rank.copy()
    new = np.float64(new)
    for i in range(len(cols)):
      X = sorted(X, key=lambda x: x[i], reverse = False)
      new = sorted(new, key=lambda x: x[i], reverse = False)
      for j in range(size):
          new[j][i] = X[j][i]
    new = pd.DataFrame(data  = new , columns=cols)
    indexes = list(new.index)
    random.shuffle(indexes)
    new_shuffled = new.loc[indexes]
    new_shuffled = new_shuffled.reset_index()
    new_shuffled = new_shuffled[cols]
    return new_shuffled
```

Listing 6: Algorytm Imana-Conovera

Algorytm został zaimplementowany zgodnie z krokami opisanymi w rozdziale drugim w podpunkcie "Transformacja Imana-Conovera". Jako argumenty funkcja przyjmuje dane do skorelowania, a także oczekiwaną macierz korelacji. Dane wejściowe są nieskorelowane co pokazane zostało na rysunku 14, dlatego nie trzeba wykonywać części transformacji z dekorelacją zmiennych. Macierz korelacji natomiast zapisana była w modelu i stamtąd jest pobierana do funkcji. Kolejne linijki kodu odpowiadają poszczególnym krokom: Nadanie rang, normalizacja rozkładu o średniej 0 i odchyleniu standardowym 1, transformacja Choleskiego dla macierzy korelacji, obliczenie nowej macierzy ze wzoru

z punktu 6 i przypisanie im rang. Następnie pozostało odpowiednio permutować zmienne, aby otrzymać oczekiwane korelacje. Do tego celu wykorzystano algorytm sortowania. Mając dane wejściowe (data) oraz oczekiwaną wartość rangowania (new), sortując po odpowiedniej kolumnie zmiennej new i data, w łatwy sposób można przypisywać oryginalne wartości do odpowiedniej rangi. Dzięki temu otrzymany został zestaw danych z oryginalnymi wartościami, ale z odpowiednio nadanymi korelacjami między zmiennymi. Po przypisaniu wartości, ostatnia kolumna była posortowana rosnąco, co mogłoby być problematyczne, gdyby do analizy brano np. połowę danych. Z tego powodu po całej transformacji przemieszano jeszcze kolejność występowania wierszy za pomocą funkcji shuffle(). Następnie sprawdzono poprawność algorytmu porównując macierz korelacji dla danych oryginalnych

	X	Y	E	U	V	W
X	1.0000	-0.0006	0.0018	0.9818	-0.0006	-0.0004
Y	-0.0006	1.0000	-0.0013	-0.0005	0.9815	0.0138
E	0.0018	-0.0013	1.0000	0.0016	-0.0013	0.1396
U	0.9818	-0.0005	0.0016	1.0000	-0.0003	-0.0001
V	-0.0006	0.9815	-0.0013	-0.0003	1.0000	0.0139
W	-0.0004	0.0138	0.1396	-0.0001	0.0139	1.0000

Rysunek 16: Oryginalna macierz korelacji dla zmiennych z modelu

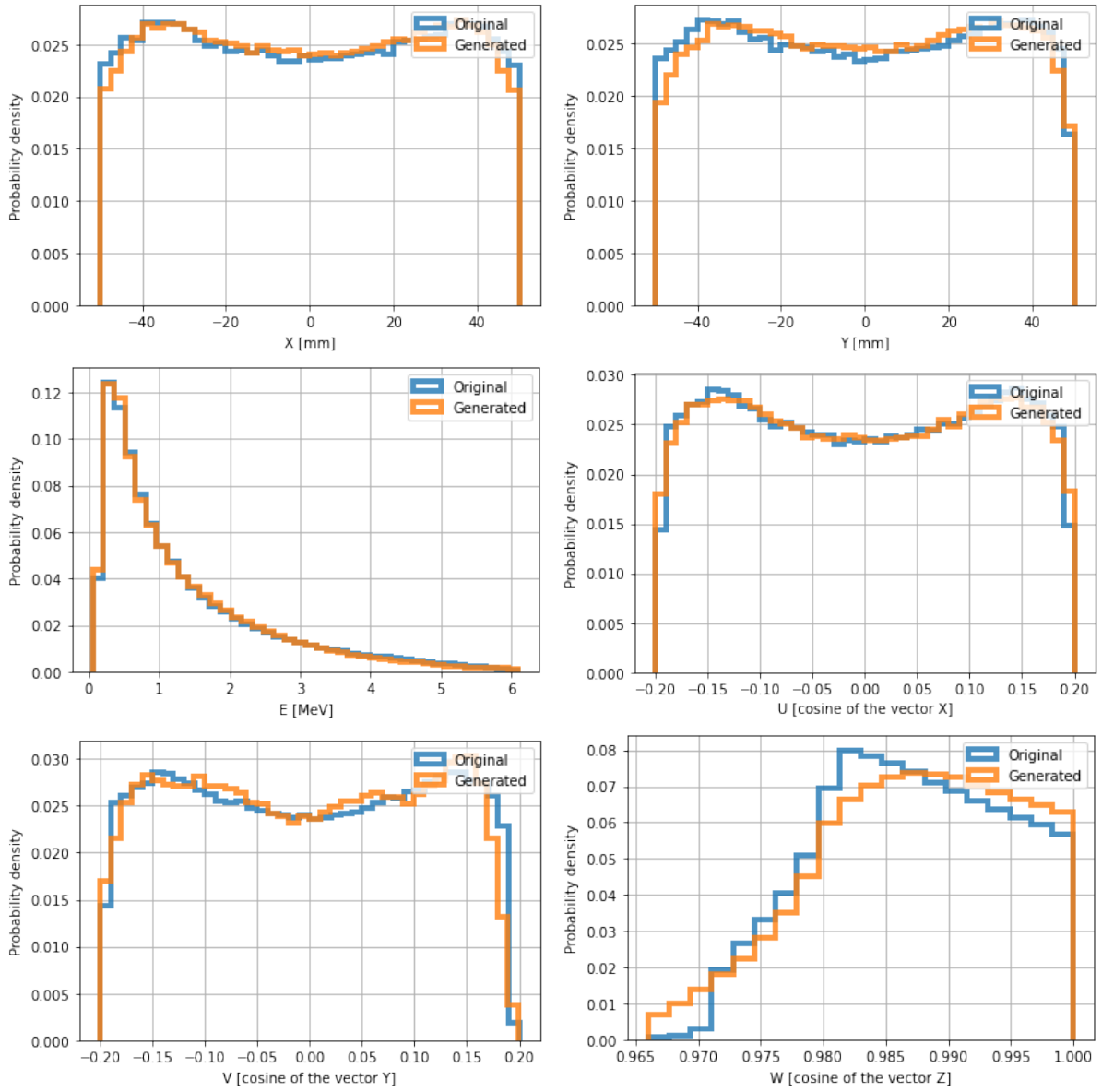
	X	Y	E	U	V	W
X	1.0000	-0.0004	-0.0001	0.9791	-0.0005	-0.0010
Y	-0.0004	1.0000	0.0011	-0.0005	0.9779	0.0473
E	-0.0001	0.0011	1.0000	0.0001	0.0014	0.1069
U	0.9791	-0.0005	0.0001	1.0000	-0.0005	-0.0009
V	-0.0005	0.9779	0.0014	-0.0005	1.0000	0.0459
W	-0.0010	0.0473	0.1069	-0.0009	0.0459	1.0000

Rysunek 17: Macierz korelacji po transformacji Imana-Conovera

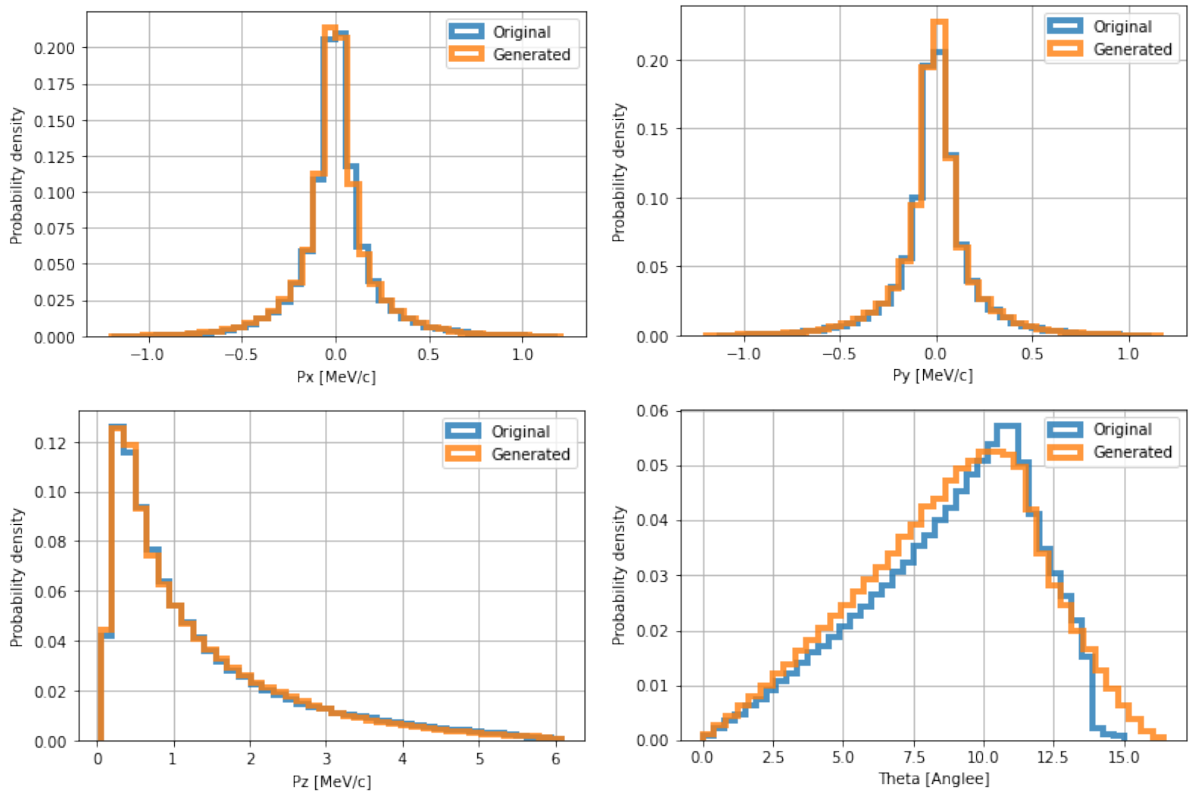
Jak widać na rysunkach 16 i 17 algorytm Imana-Conovera matematycznie poprawnie odtworzył macierz korelacji.

5 Walidacja

Porównano rozkłady otrzymane po fitowaniu i transformacji Imana-Conovera z rozkładami wejściowymi. Sprawdzono zarówno zmienne, które brały udział w tworzeniu modelu jak i pozostałe. Przy tych pierwszych rozkłady nie powinny się w ogóle różnić od tych po fitowaniu, ponieważ tak działa algorytm Imana-Conovera. Istotne jest natomiast jak odtworzone będą pozostałe zmienne. To właśnie ich porównanie jest istotne przy sprawdzeniu poprawności działania modelu i transformacji.



Rysunek 18: Porównanie rozkładów wyjściowych z wejściowymi



Rysunek 19: Porównanie rozkładów wyjściowych z wejściowymi

Jak widać rozkłady wyjściowe z duża dokładnością odpowiadają rozkładom wejściowym. Jedyne widoczne odchylenia są przy rozkładzie zmiennej W i Theta

Na rysunku 21 można zobaczyć, że macierz korelacji jest zbliżona do macierzy z rysunku 20. Niektóre korelacje, na poziomie niższym od 0.2 nie zostały odtworzone. Wszystkie silniejsze relacje, które pojawiły się w oryginalnych danych przestrzeni fazowej, znalazły się również w tych wygenerowanych przez model.

	X	Y	E	Px	Py	Pz	U	V	W	Theta
X	1.0000	-0.0021	0.0035	0.8785	-0.0010	0.0035	0.9816	-0.0021	-0.0017	0.0017
Y	-0.0021	1.0000	-0.0009	-0.0008	0.8782	-0.0008	-0.0020	0.9810	0.0129	-0.0129
E	0.0035	-0.0009	1.0000	0.0024	-0.0050	1.0000	0.0033	-0.0010	0.1413	-0.1413
Px	0.8785	-0.0008	0.0024	1.0000	-0.0006	0.0024	0.8906	-0.0009	-0.0007	0.0007
Py	-0.0010	0.8782	-0.0050	-0.0006	1.0000	-0.0049	-0.0010	0.8909	0.0093	-0.0093
Pz	0.0035	-0.0008	1.0000	0.0024	-0.0049	1.0000	0.0033	-0.0009	0.1490	-0.1490
U	0.9816	-0.0020	0.0033	0.8906	-0.0010	0.0033	1.0000	-0.0022	-0.0015	0.0015
V	-0.0021	0.9810	-0.0010	-0.0009	0.8909	-0.0009	-0.0022	1.0000	0.0129	-0.0129
W	-0.0017	0.0129	0.1413	-0.0007	0.0093	0.1490	-0.0015	0.0129	1.0000	-1.0000
Theta	0.0017	-0.0129	-0.1413	0.0007	-0.0093	-0.1490	0.0015	-0.0129	-1.0000	1.0000

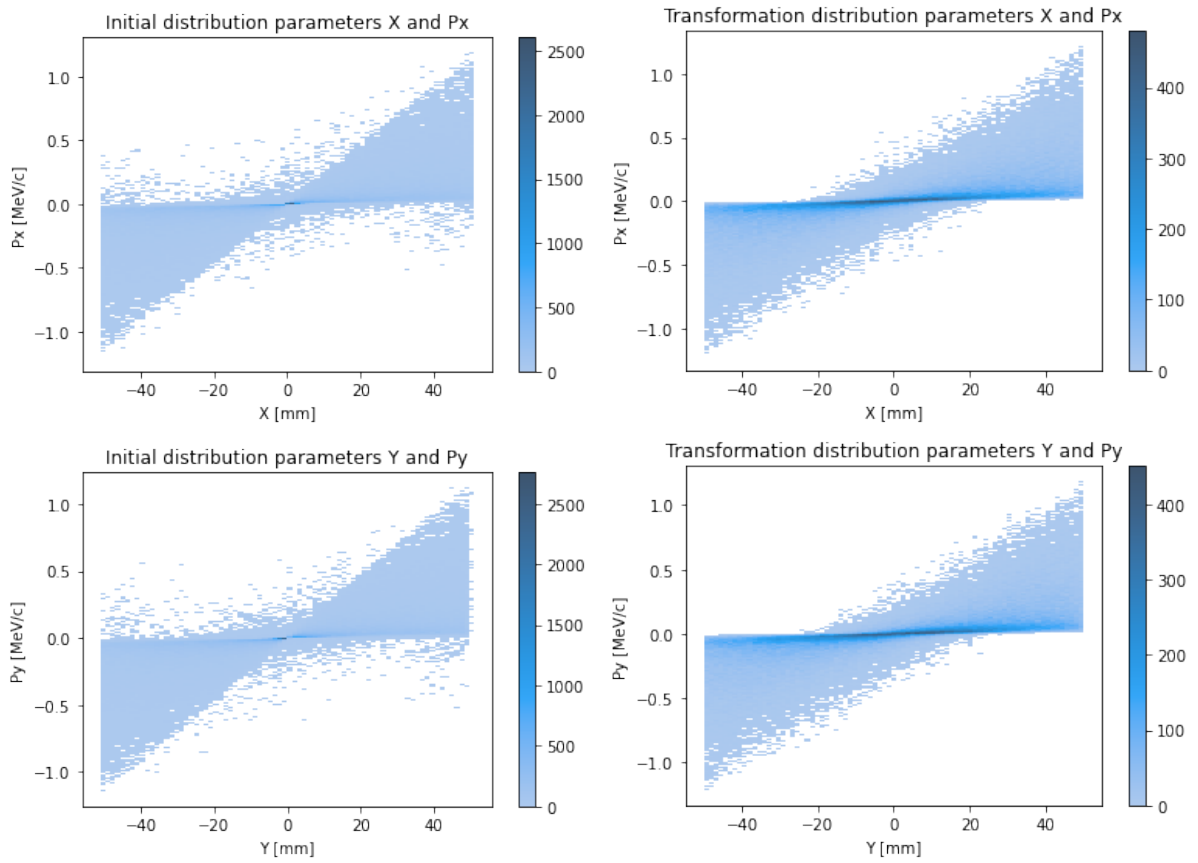
Rysunek 20: Oryginalna macierz korelacji

	X	Y	E	Px	Py	Pz	U	V	W	Theta
X	1.0000	-0.0004	-0.0001	0.8820	-0.0001	-0.0001	0.9791	-0.0005	0.0008	-0.0008
Y	-0.0004	1.0000	0.0011	-0.0004	0.8805	0.0017	-0.0005	0.9779	0.0613	-0.0613
E	-0.0001	0.0011	1.0000	-0.0006	-0.0181	1.0000	0.0001	0.0014	0.0014	-0.0014
Px	0.8820	-0.0004	-0.0006	1.0000	-0.0006	-0.0006	0.9025	-0.0005	0.0005	-0.0005
Py	-0.0001	0.8805	-0.0181	-0.0006	1.0000	-0.0176	-0.0001	0.9022	0.0487	-0.0487
Pz	-0.0001	0.0017	1.0000	-0.0006	-0.0176	1.0000	0.0001	0.0021	0.0098	-0.0098
U	0.9791	-0.0005	0.0001	0.9025	-0.0001	0.0001	1.0000	-0.0005	0.0008	-0.0008
V	-0.0005	0.9779	0.0014	-0.0005	0.9022	0.0021	-0.0005	1.0000	0.0631	-0.0631
W	0.0008	0.0613	0.0014	0.0005	0.0487	0.0098	0.0008	0.0631	1.0000	-1.0000
Theta	-0.0008	-0.0613	-0.0014	-0.0005	-0.0487	-0.0098	-0.0008	-0.0631	-1.0000	1.0000

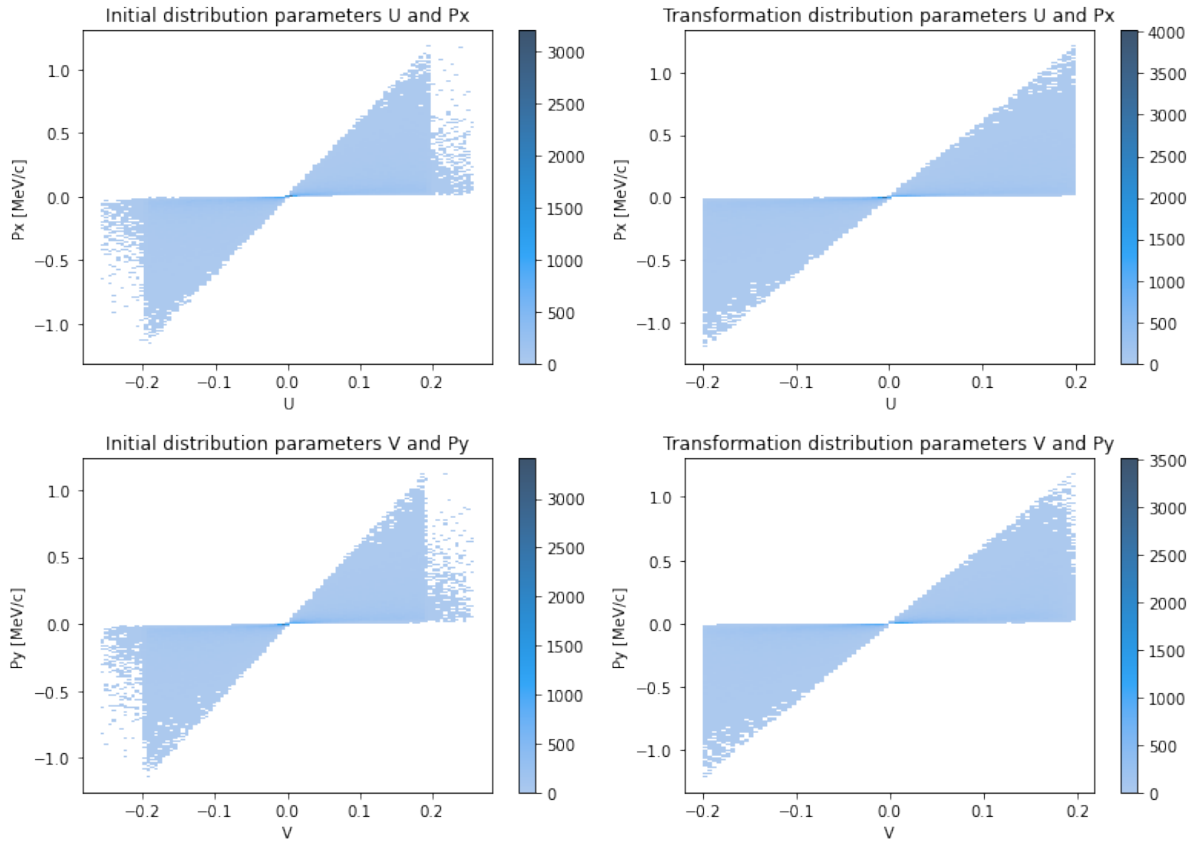
Rysunek 21: Macierz korelacji po transformacji

5.1 Porównanie poszczególnych zmiennych między sobą

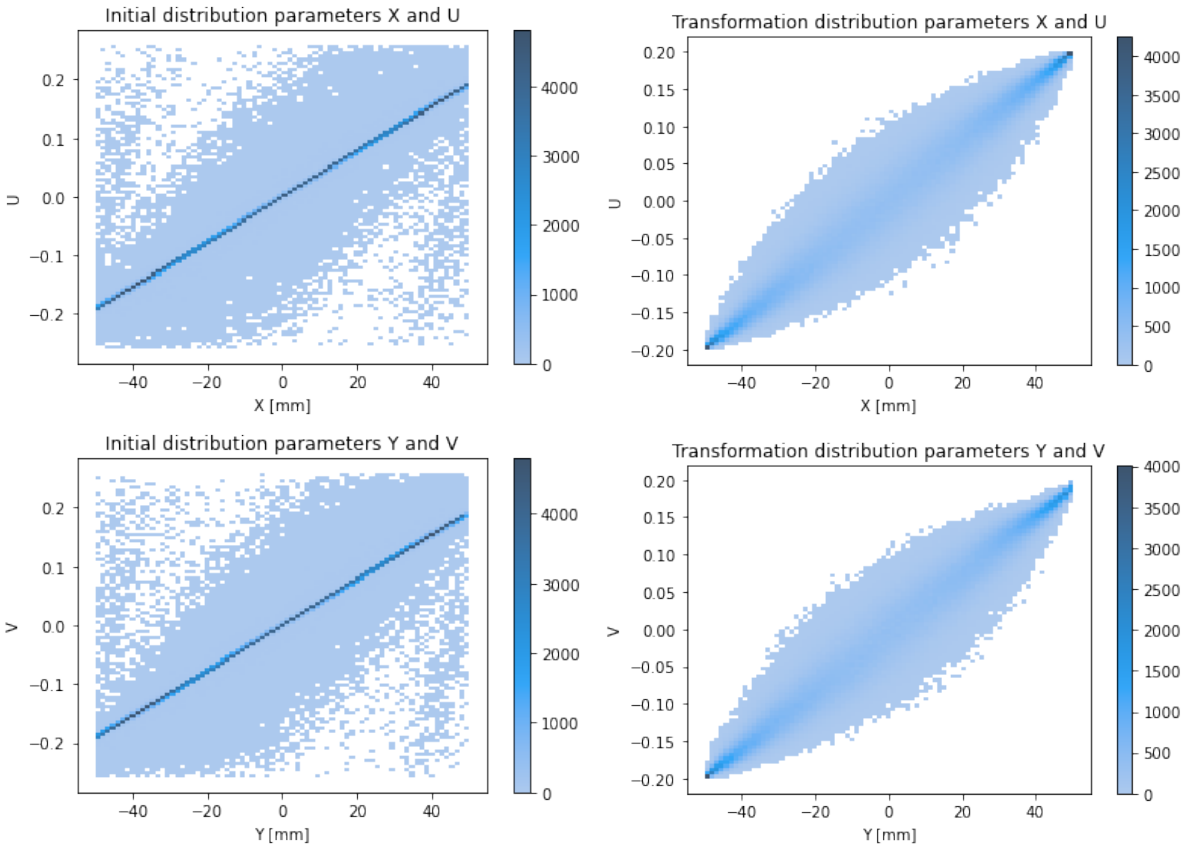
W kolejnym kroku sprawdzono jak odtworzone zostały relacje między poszczególnymi zmiennymi. Sprawdzono parametry, które były silnie ze sobą skorelowane, a także niektóre zależności, których nie widać było na macierzy korelacji.



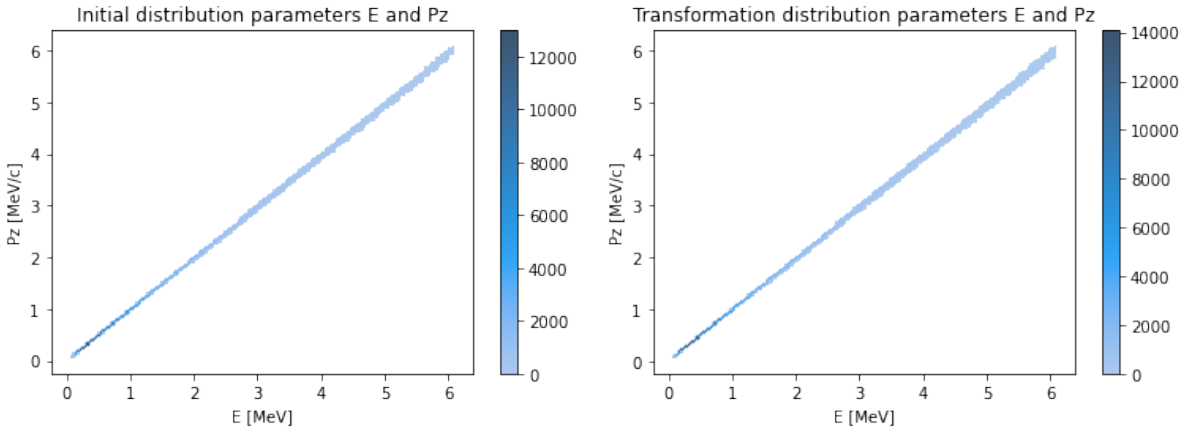
Rysunek 22: Porównanie zależności kierunków z wartościami pędu w tym kierunku



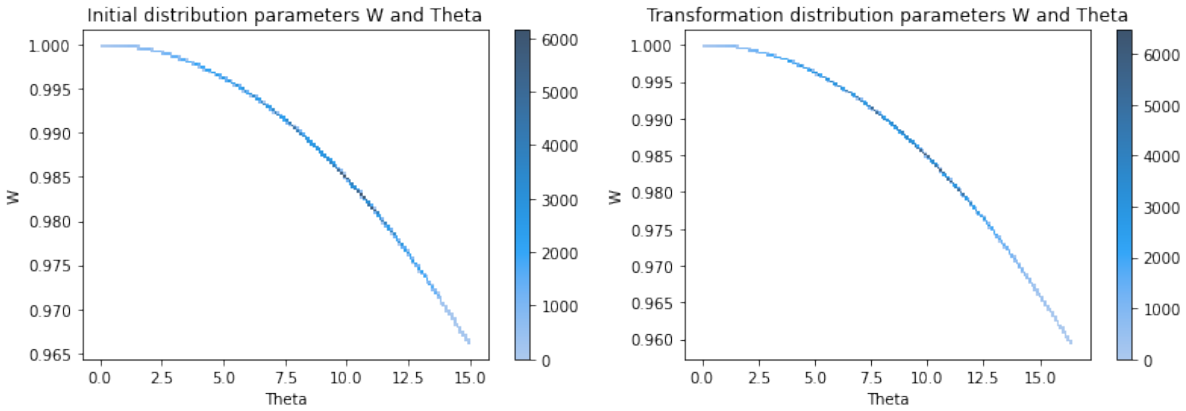
Rysunek 23: Porównanie zależności pędów kierunkowych z cosinusami kierunkowymi



Rysunek 24: Porównanie zależności kierunków z cosinusami odchyleń w tych kierunkach



Rysunek 25: Porównanie zależności energii z pędem w kierunku składowej Z



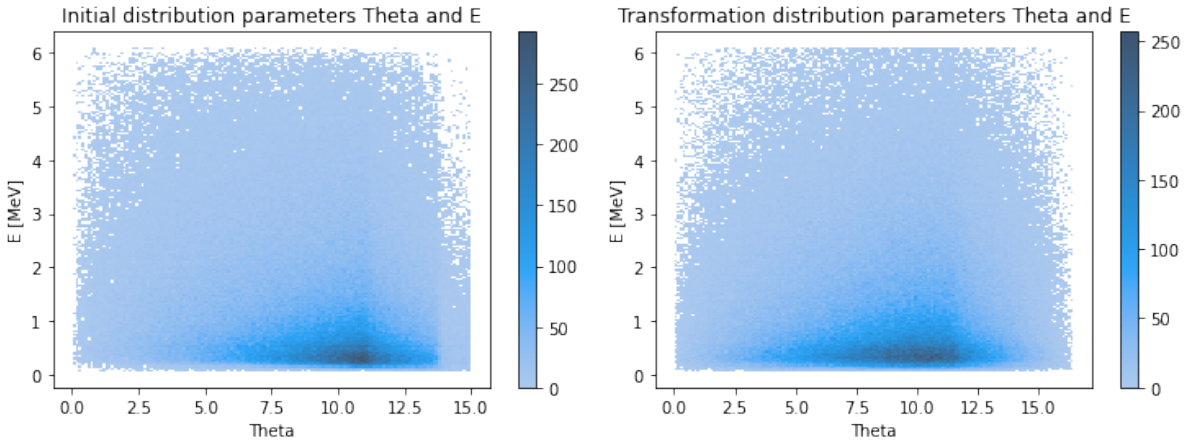
Rysunek 26: Porównanie zależności cosinusa w kierunku składowej Z z kątem Theta

Sprawdzone zostały gdzie korelacje były wysokie takie jak:

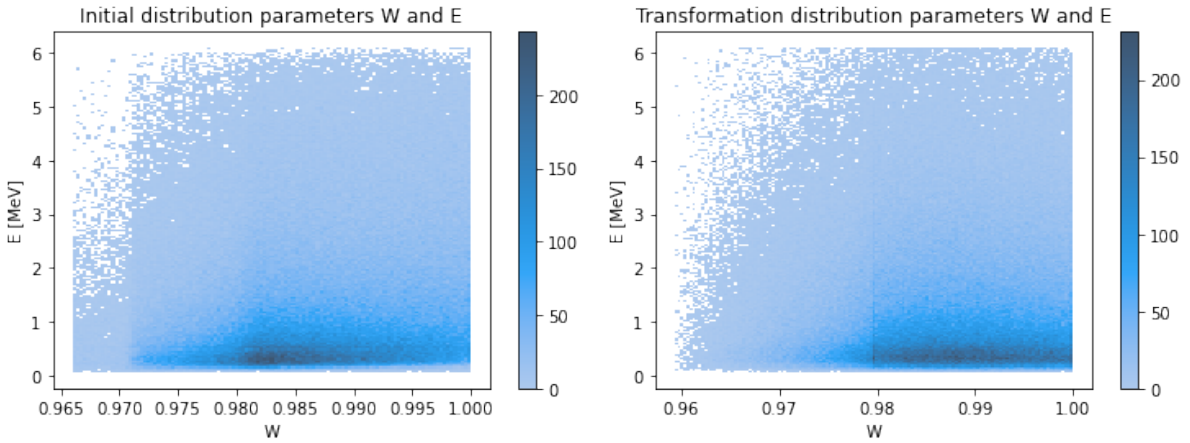
- X-Px, Y-Py (rys. 22)
- X-U, Y-V (rys. 24)
- Px-U, Py-V (rys. 23)
- E-Pz (rys. 25)
- W-Theta (rys. 26)

W każdej z relacji wypisanych wyżej, zależności zostały odtworzone w poprawny sposób, zwłaszcza dla wartości gdzie korelacje były bliskie 1. W niektórych przypadkach można zauważyć minimalne różnice. Wynikają one z tego, że algorytm Imana-Conovera dopasowuje korelacje liniowo, co nie zawsze musi mieć miejsce.

Następnie sprawdzono korelacje, które wykryte były w danych wejściowych, a nie pojawiły się po transformacji.



Rysunek 27: Porównanie zależności energii z kątem Theta



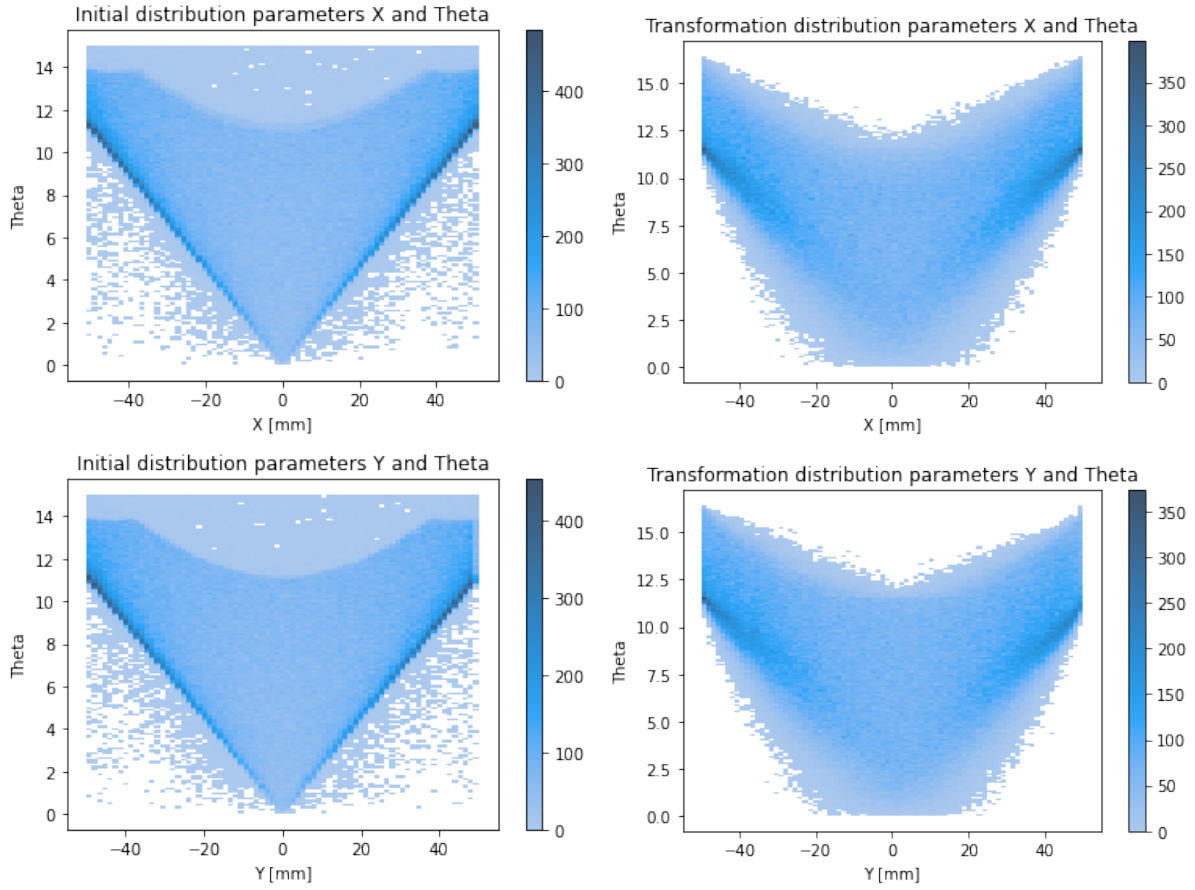
Rysunek 28: Porównanie zależności energii z cosinusem składowej kierunku Z

Sprawdzono następujące pary:

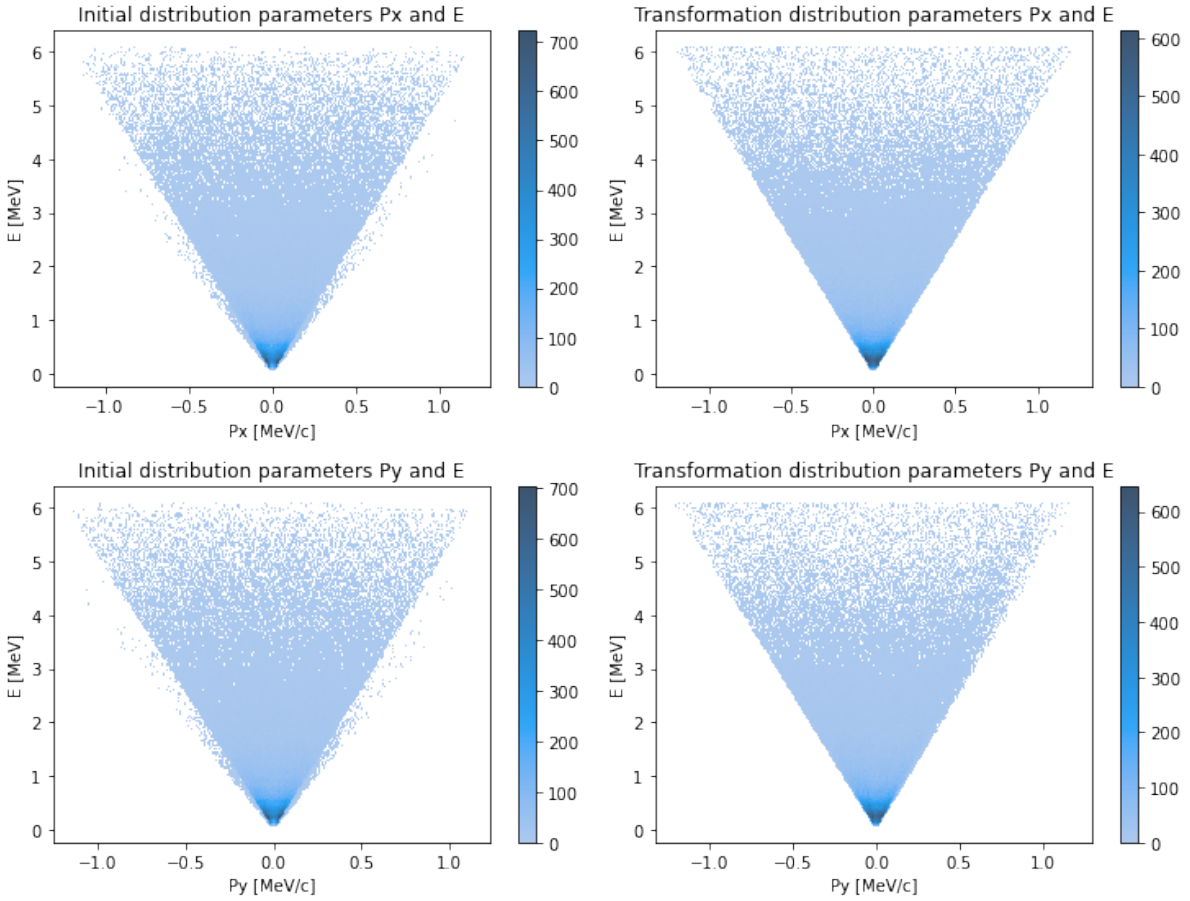
- E-Theta (rys. 27)
- E-W (rys. 28)

Nie zauważono specjalnych różnic między oryginalnymi danymi a tymi po transformacji. Wynika to z tego, że dane korelacje były bardzo niskie. Sprawdzono je jednak, żeby zweryfikować czy różnicę na pewno nie wpłyną na jakość odtwarzania fotonów.

Ostatnimi porównaniami jakie wykonano, były korelacje, których matematycznie nie było widać, jednak były wyraźnie widoczne podczas analizy.



Rysunek 29: Porównanie odchyleń z kątem Theta



Rysunek 30: Porównanie składowych pędów z energią

Można wyróżnić następujące przypadki:

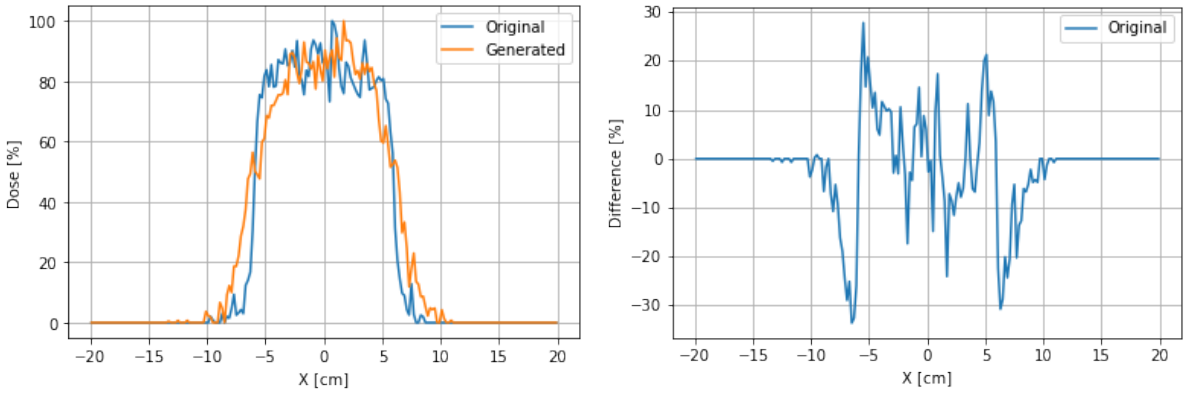
- X-Theta, Y-Theta (rys. 29)
- Px-E, Py-E (rys. 30)

Jak widać na przykładzie rysunków 29 i 30 relacje są znacznie widoczne to znaczy im większe wychylenie od osi X lub Y tym kąt Theta większy. Korelacja nie była w stanie tego zauważyć, ze względu na symetrię rozkładów. Kąt rośnie wraz z wartością bezwzględną wychylenia, jednak może być ono ujemne, dlatego ostatecznie korelacja jest w okolicach 0. Mimo to model poprawnie odtworzył zmienne i ich zależności między sobą.

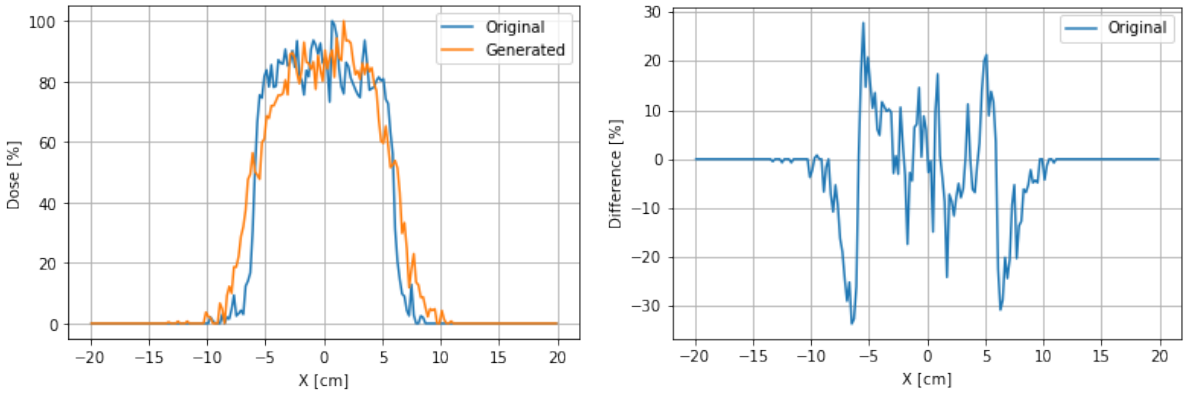
5.2 Porównanie rozkładu dawki w PRIMO

W poprzednich punktach sprawdzano poprawność modelu pod względem rozkładów statystycznych rozważanych zmiennych. Tym razem cała przestrzeń

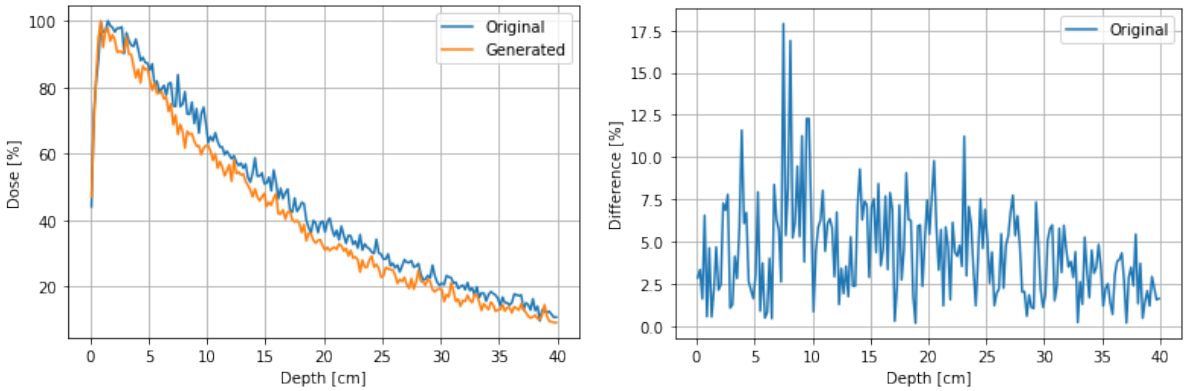
fazowa została poddana testowi. Sprawdzono, czy symulacje dotyczące rozkładu dawek w radioterapii są zgodne dla przestrzeni fazowej oryginalnej i wygenerowanej. Do weryfikacji użyto symulatora PRIMO, a konkretnie akceleratora Varian Clinac 2100. Symulacja przeprowadzana została na fantomie wodnym o wymiarach **40cm x 40cm x 40cm**. Było stałe pole promieniowania o wymiarach **5cm x 5cm**. Liczba cząstek biorąca udział w symulacji wyniosła **6000000**. Sprawdzono profile poprzeczne i głębokościowe. Otrzymano następujące wyniki:



Rysunek 31: Porównanie rozkładów dawki w profilu poprzecznym X i ich różnica



Rysunek 32: Porównanie rozkładów dawki w profilu poprzecznym X i ich różnica



Rysunek 33: Porównanie rozkładów dawki w profilu głębokościowym i ich różnica

W obu profilach poprzecznych na rysunkach 31 i 32 widać pokrycie się rozkładu oryginalnego z wygenerowanym na prawie całym wykresie. Jedynymi różnicami są obszary narastania dawki. W wygenerowanych rozkładach jest on nieco dłuższy to znaczy, zaczyna narastać wcześniej, a osiąga maksimum później. Dla profilu głębokościowego pokrycie się rozkładów jest zadowalające. Różnice jakie można zobaczyć biorą się prawdopodobnie ze statystyki jaka była wykorzystana do symulacji, a która była ograniczona brakiem odpowiedniego interfejsu do zapisu wygenerowanej statystyki fotonów bezpośrednio do formatu IAEA (formatem pośredniczącym był csv).

6 Podsumowanie

Celem projektu było stworzenie narzędzi umożliwiających generowanie wiązki fotonów, a następnie jej weryfikacja w programie PRIMO. Zaimplementowany został model, który posiada informacje o wybranych parametrach przestrzeni fazowej i ich wzajemnej korelacji. Następnie poddano go transformacji Imana-Conovera. Porównano wejściową i wygenerowaną przestrzeń fazową. Rozkłady i korelację zostały poprawnie odtworzone, również dla zmiennych, które nie tworzyły modelu. Pozwala to stwierdzić, że zarówno właściwości matematyczne jak i fizyczne zostały spełnione. Najważniejszą weryfikacją dla przestrzeni fazowej była symulacja rozkładu dawki w PRIMO. W tym przypadku wyniki również były zadowalające, zwłaszcza dla profilu głębokościowego. Dla profili poprzecznych powstały małe rozbieżności w obszarze narastania dawki. Dzięki poprawnemu odtworzeniu przestrzeni fazowej, można w łatwy i szybki sposób tworzyć nowe fotony do symulacji. Jest to ważna część projektu, ponieważ generowanie przestrzeni fazowych pochłania dużo mocy obliczeniowej, którą można zaoszczędzić korzystając ze stworzonego modelu. Sam algorytm Imana-Conovera zaimplementowany w projekcie może być wykorzystywany nie tylko do przestrzeni fazowych, ale także do jakichkolwiek danych, którym chce się nadać określoną korelację, nie zmieniając przy tym ich wartości. Cele projektu zostały spełnione. Model można wykorzystywać do symulacji jak np. we wspomnianym rozkładzie dawki w radioterapii, ale i nie tylko. Wyniki są obiecujące, a projekt może być w przyszłości rozszerzany o dalsze cele lub z jeszcze większą dokładnością odtworzyć przestrzenie fazowe.

Bibliografia

(1) M. TEOH, C. H. CLARK, K WOOD, S WHITAKER, A NISBET. Volumetric modulated arc therapy: a review of current literature and clinical use in practice (2011r.)

(2) E. OPONOWICZ. Medyczne akceleratory elektronów (2013r.)

(3) https://www-nds.iaea.org/phsp/phsp.htmlx

(4) N. MATUSZAK. Metoda Monte Carlo i jej zastosowanie w radioterapii (2019r.)

(5) https://mfiles.pl/pl/index.php/Kowariancja

(6) https://predictivesolutions.pl/rangowanie-danych-metody-i-przyklady

(7) https://www.youtube.com/watch?v=4SWMzENcgSE

(8) https://blogs.sas.com/content/iml/2012/02/08/use-the-cholesky-transformation-to-correlate-and-uncorrelate-variables.html

(9) STEPHEN J. MILDENHALL. Correlation and Aggregate Loss Distributions With An Emphasis On The Iman-Conover Method (2005r.)

(10) R. IMAN, W. CONOVER. A distribution-free approach to inducing rank correlation among input variables (1982r.)

(11) A. BUSZKO. Cele, zadania i metody radioterapii na przykładzie Centrum Onkologii – Instytutu Marii Skłodowskiej Curie w Warszawie przy ul. Wawelskiej 15 (dostęp: grudzień 2022r.)

(12) https://learn.microsoft.com/pl-pl/azure/cosmos-db/notebooks-overview

(13) https://numpy.org/doc/stable/reference/index.html

(14) https://pandas.pydata.org/docs/reference/index.html

(15) https://seaborn.pydata.org

(16) https://docs.scipy.org/doc/scipy/

(17) https://root.cern/about/

(18) https://primoproject.net/primo/

(19) https://blogs.sas.com/content/iml/2021/06/16/geometry-iman-conover-transformation.html

(20) https://github.com/tisimst/mcerp/blob/master/mcerp/correlate.py

Wersje programów

matplotlib 3.5.1

numpy 1.21.1

pandas 1.4.2

pyiaea 0.0.1

scipy 1.8.1

seaborn 0.11.2

root 6.24.2