JOURNAL OF AERONAUTICAL MATERIALS
ISSN: 1005-5053
Vol. 43, Issue-01, 2023
pp. 879-889

DETERMINATION OF OPTIMUM TAKE OFF SPEED BY INTERPOLATING DATA IN RTOW CHART FOR AN AIRCRAFT

Lovely Jain[1] & Samarth Srivastava[2]
Amity Institute of Applied Sciences, Amity University, Sector 125 Noida 201303
luvlyjain@gmail.com[1], samarth.s2003@gmail.com[2]

Abstract
Takeoff speeds are a crucial piece of information which are required before commencing any flight. A technique to calculate V-Speeds for aircraft using appropriate interpolation in RTOW charts is proposed in this paper. The RTOW (Regulated Takeoff Weight) charts provide a valuable source of data for evaluating takeoff speeds for different weather conditions for a particular runway at a particular airport. This paper applies interpolation techniques to accurately interpret the data from the RTOW chart and hence precisely calculate the V-Speeds for different temperature and wind speed values.
Keywords: Aviation Industry, Takeoff Speeds, Air Travel, RTOW Chart

1. Introduction

Recent papers have provided with application of interpolation in aviation such as wind analysis, and the time interpolation method on the altitude optimization algorithm for FMS CMA-9000. [1],[2],[3].
This paper identifies techniques to use interpolation in RTOW charts to calculate the optimum takeoff speed for an aircraft for suitable weather conditions. V1, VR and V2 speeds have their own functions, as explained in the paper's later sections. This paper talks about new developments related to the application of interpolation in calculating the appropriate takeoff speeds based on changing meteorological conditions.
Being an important stage of a flight, it is a must to guarantee a safe takeoff. To do the same, calculating V-speeds must be done accurately while keeping in mind all the meteorological data, the aircraft payload, and fuel weight. This speed depends on aircraft type, weight, and meteorological conditions. For example, an A320-216S with CFM 56-5B6 engines, a takeoff weight of 78.2 tons departing from Hyderabad runway 27R Dry runway, with flaps 1+F, TOGA, without anti-ice and air-conditioning at an outside air temperature of 29 degrees Celsius, and calm winds will have its V1, VR and V2 speeds as 152, 160 and 161, respectively. The interpolation techniques used in this paper include Newton Forward Interpolation, Newton Backward Interpolation, Newton Divided Difference Interpolation, Gauss Interpolation and Bessel Interpolation. Python formulas are used to carry out the necessary calculations. [4]
This paper is organized as follows, explanation of different V-Speeds during takeoff, theoretical explanation of various mentioned interpolation methods, RTOW Chart for Hyderabad and python formulas for different interpolation techniques. It is followed by the result of the Numerical Analysis and the conclusion which contains the interpretation of those results. It is proceeded by the list of references which ends the paper.

2. Takeoff Speeds

For a safe takeoff execution, 3 takeoff speeds are calculated. These V-speeds depend on several factors including aircraft type, weight, available runway length for departure, degree of flap extension, engine thrust and defilements on the surface of the runway. V-speeds are defined as follows:

2.1. V_1 Speed

V_1: Also known as the takeoff decision speed, is described as the speed after which takeoff should not be aborted in any case. The takeoff will continue even if the aircraft has reached its V_1 speed even if there is an engine failure or tyre burst, etc. Aborting the takeoff after V_1 will, by definition, result in the aircraft being unable to stop before the runway end and suffering a runway overrun.

2.2. V_R Speed

V_R: It is the speed when rotation is commenced to carry out a safe takeoff. It is the speed when the pilot pulls back the control column to pitch the nose up. Or it is the speed when the nose wheel (frontmost wheel) of the airplane is lifted from the ground.

2.3. V_2 Speed

V_2: Also called "take-off safety speed", it refers to the speed of safe climb at 200 feet per minute that can be carried out with 1 engine inoperative [5]

Fig. 1: Takeoff Speeds

3. Analysis of RTOW Chart

For the calculation of takeoff speeds, data used in this paper is provided by an RTOW (Regulated takeoff weight) chart. Let us assume that an Airbus A320-216S with CFM56-5B6 Engines, is departing Hyderabad runway 27R with flaps 1+F, and TOGA thrust. The QNH (Air Pressure) is 1013.25 HPA, and the runway is dry. Then, according to the RTOW chart, the V_1 speed, V_R speeds and V_2 speeds of that aircraft with varying temperatures would be gives as

OAT (C)	Tailwind -10KT	Tailwind -5KT	Wind 0KT	Headwind 5KT	Headwind 10KT
13	144	150	156	158	161
17	143	149	155	157	159
21	142	148	154	156	158
25	142	147	153	155	157
29	141	146	152	154	156
33	140	145	151	153	155
37	139	145	150	152	154
41	139	144	149	151	153
45	140	145	151	153	155
49	142	147	153	155	157
53	143	149	155	157	159
57	145	151	157	159	160
61	147	153	158	158	157

Fig. 2: RTOW Chart for V_1 Speed

OAT (C)	Tailwind -10KT	Tailwind -5KT	Wind 0KT	Headwind 5KT	Headwind 10KT
13	153	158	163	165	167
17	153	157	162	164	166
21	152	157	161	163	165
25	151	156	160	162	164
29	150	155	160	161	163
33	150	154	159	160	162
37	149	154	158	160	161
41	149	153	157	159	160
45	148	153	157	159	161
49	148	153	158	159	161
53	149	153	158	160	162
57	149	154	159	161	162
61	149	155	159	159	159

Fig. 3: RTOW Chart for V_R Speed

OAT (C)	Tailwind -10KT	Tailwind -5KT	Wind 0KT	Headwind 5KT	Headwind 10KT
13	155	159	164	166	168
17	154	159	163	165	167
21	153	158	163	164	166
25	153	157	162	163	165
29	152	156	161	162	164
33	151	156	160	162	163
37	151	155	159	161	162
41	150	154	159	160	162
45	149	154	158	160	162
49	149	154	159	160	162
53	149	154	159	161	163
57	149	154	160	161	162
61	150	155	160	160	160

Fig. 3: RTOW Chart for V_2 Speed

4. Applying Interpolation in RTOW Chart

4.1. Calculating v-speeds by interpolating for varying temperature

To calculate 3 v-speeds at 15 degrees Celsius at a tailwind of 10 knots, Newton Forward Interpolation is used as the value at which interpolation is carried out, is near the beginning of the table. The python code for Newton Forward Interpolation [6] used to calculate the V_1, V_R and V_2 speeds is given below which gives the V-speeds to be 146.040817, 158.726703 and 153.283333 knots at 15 degrees centigrade with 10 knots of tailwind.

```
def calculate(u,n):
tmp=u
for i in range(1,n):
tmp=tmp*(u-i)
return tmp
def factorial(n):
fact=1
for i in range(2,n+1):
fact=fact*i
return fact
def newton_forward():
global n,x,y,v
for i in range(1,n):
for j in range(0,n-1):
y[j][i]=y[j+1][i-1]-y[j][i-1]
for i in range (0,n):
print(round(x[i],6),end="\t")
for j in range(0,n-1):
print(round(y[i][j],6),end="\t")
print()
s=y[0][0]
u=(v-x[0])/(x[i]-x[0])
for i in range(1,n):
s=s+(calculate(u,i)*y[0][i])/factorial(i)
print("value at ",v," is ",s)
n=int(input("Enter the number of elements: "))
x=[]
y=[[0 for i in range(n)]
for j in range(n)]
print("Enter the values of x: ")
for i in range(0,n):
x.append(float(input()))
print("Enter the values of y: ")
for i in range(0,n):
y[i][0]=float(input())
v=float(input("Enter the value of interpolation: "))
newton_forward()
```

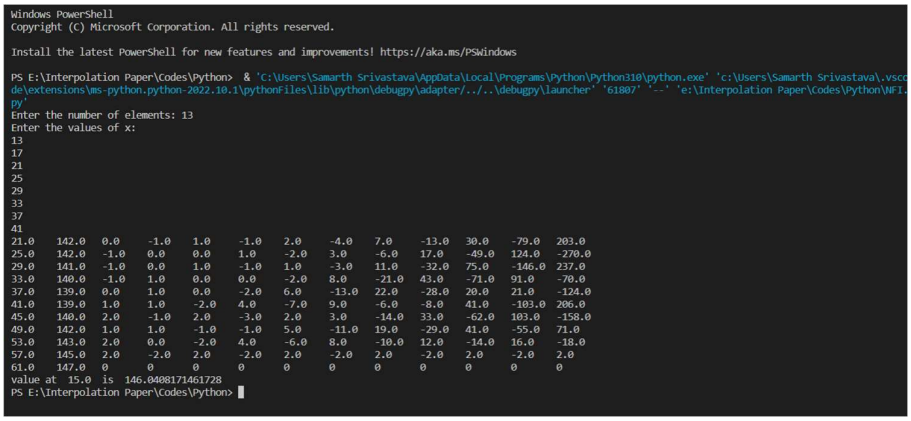

Fig. 4: Value of V_1 Speed after applying Newton Forward Interpolation

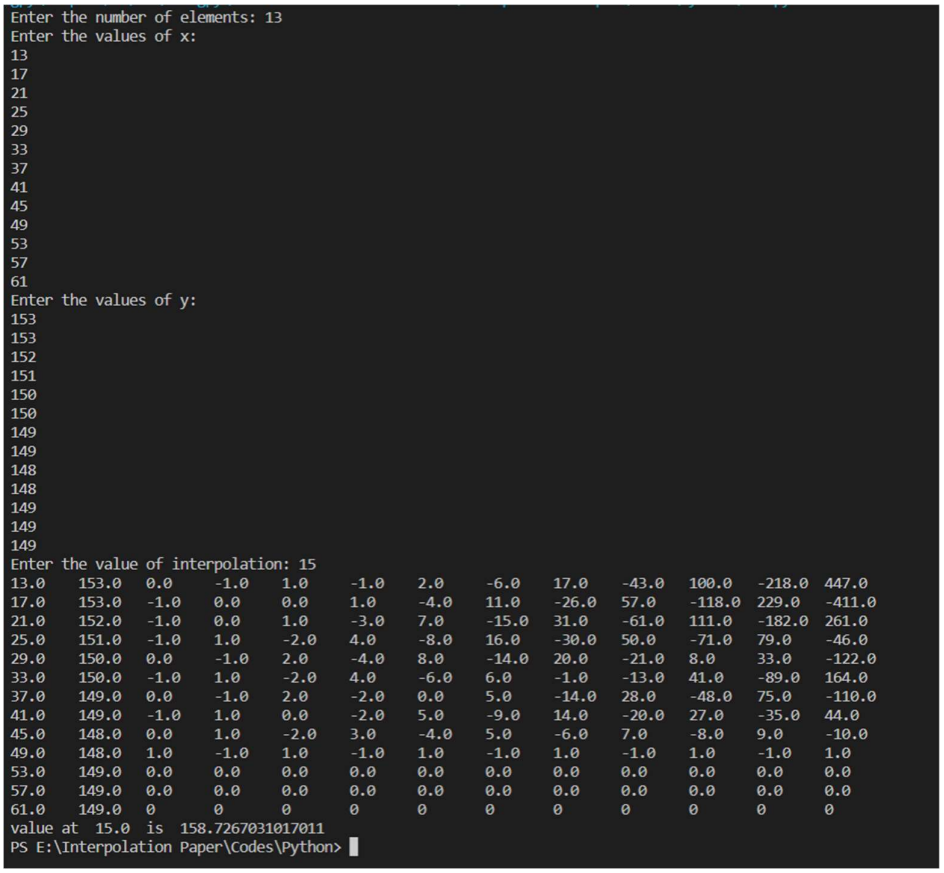

Fig. 5: Value of V_R Speed after applying Newton Forward Interpolation

```
Enter the number of elements: 13
Enter the values of x:
13
17
21
25
29
33
37
41
45
49
53
57
61
Enter the values of y:
155
154
153
153
152
151
151
150
149
149
149
149
150
Enter the value of interpolation: 15
13.0    155.0   -1.0    0.0     1.0     -3.0    6.0     -9.0    9.0     0.0     -27.0   82.0    -171.0
17.0    154.0   -1.0    1.0     -2.0    3.0     -3.0    0.0     9.0     -27.0   55.0    -89.0   117.0
21.0    153.0   0.0     -1.0    1.0     0.0     -3.0    9.0     -18.0   28.0    -34.0   28.0    -3.0
25.0    153.0   -1.0    0.0     1.0     -3.0    6.0     -9.0    10.0    -6.0    -6.0    25.0    -36.0
29.0    152.0   -1.0    1.0     -2.0    3.0     -3.0    1.0     4.0     -12.0   19.0    -11.0   -46.0
33.0    151.0   0.0     -1.0    1.0     0.0     -2.0    5.0     -8.0    7.0     8.0     -57.0   175.0
37.0    151.0   -1.0    0.0     1.0     -2.0    3.0     -3.0    -1.0    15.0    -49.0   118.0   -243.0
41.0    150.0   -1.0    1.0     -1.0    1.0     0.0     -4.0    14.0    -34.0   69.0    -125.0  209.0
45.0    149.0   0.0     0.0     0.0     1.0     -4.0    10.0    -20.0   35.0    -56.0   84.0    -120.0
49.0    149.0   0.0     0.0     1.0     -3.0    6.0     -10.0   15.0    -21.0   28.0    -36.0   45.0
53.0    149.0   0.0     1.0     -2.0    3.0     -4.0    5.0     -6.0    7.0     -8.0    9.0     -10.0
57.0    149.0   1.0     -1.0    1.0     -1.0    1.0     -1.0    1.0     -1.0    1.0     -1.0    1.0
61.0    150.0   0       0       0       0       0       0       0       0       0       0       0
value at  15.0  is  153.2833123834557
PS E:\Interpolation Paper\Codes\Python>
```

Fig. 6: Value of V_2 Speed after applying Newton Forward Interpolation

4.2. Calculating v-speeds by interpolating for varying wind speeds

To calculate 3 v-speeds at 17 degrees Celsius, but at a headwind of 2 knots, Bessel Interpolation is used since the value at which interpolation needs to be carried out, is near the centre of the table.

The python code for Bessel Interpolation [6] is :

```
def ucal(u, n):
if (n == 0):
return 1
temp = u;
for i in range(1, int(n / 2 + 1)):
temp = temp * (u - i)
for i in range(1, int(n / 2)):
temp = temp * (u + i)
return temp;
```

```
def fact(n):
f = 1;
for i in range(2, n + 1):
f *= i;
return f;
n = 6;
x = [25, 26, 27, 28, 29, 30];
y = [[0 for i in range(n)]
for j in range(n)];
y[0][0] = 154;
y[1][0] = 159;
y[2][0] = 163;
y[3][0] = 165;
y[4][0] = 167;
for i in range(1, n):
for j in range(n - i):
y[j][i] = y[j + 1][i - 1] - y[j][i - 1];
for i in range(n):
for j in range(n - i):
print(y[i][j], "\t", end = " ");
print("");
value = 2;
sum = (y[2][0] + y[3][0]) / 2
k = 0;
if ((n % 2) > 0):
k = int(n / 2);
else:
k = int(n / 2 - 1); # origin for even
u = (value - x[k]) / (x[1] - x[0]);
for i in range(1, n):

if (i % 2):
sum = sum + ((u - 0.5) *
ucal(u, i - 1) *
y[k][i]) / fact(i);
else:
sum = sum + (ucal(u, i) * (y[k][i] +
y[k - 1][i]) / (fact(i) * 2));
k -= 1;
print("Value at", value, "is", round(sum, 5));
```

According to this code, the value of V_1, V_R, V_2 are 156.1456, 163.0704, and 163.9616 knots respectively.

```
PS E:\Interpolation Paper\Codes\Python>  e:; cd 'e:\Interpolation Paper\Codes\Python'; & 'C:\Users\Samarth Srivastava\AppData\Loca
l\Programs\Python\Python310\python.exe' 'c:\Users\Samarth Srivastava\.vscode\extensions\ms-python.python-2022.10.1\pythonFiles\lib
\python\debugpy\adapter/../..\debugpy\launcher' '61064' '--' 'e:\Interpolation Paper\Codes\Python\BI.py'
143     6       0       -4      8
149     6       -4      4
155     2       0
157     2
159
Value at 2 is 156.1456
PS E:\Interpolation Paper\Codes\Python>
```

Fig. 7: Value of V_1 Speed after applying Bessel Interpolation

```
PS E:\Interpolation Paper\Codes\Python>  e:; cd 'e:\Interpolation Paper\Codes\Python'; & 'C:\Users\Samarth Srivastava\AppData\Loca
l\Programs\Python\Python310\python.exe' 'c:\Users\Samarth Srivastava\.vscode\extensions\ms-python.python-2022.10.1\pythonFiles\lib
\python\debugpy\adapter/../..\debugpy\launcher' '61131' '--' 'e:\Interpolation Paper\Codes\Python\BI.py'
153     4       1       -4      7
157     5       -3      3
162     2       0
164     2
166
Value at 2 is 163.0704
PS E:\Interpolation Paper\Codes\Python>
```

Fig. 8: Value of V_R Speed after applying Newton Forward Interpolation

```
PS E:\Interpolation Paper\Codes\Python>  e:; cd 'e:\Interpolation Paper\Codes\Python'; & 'C:\Users\Samarth Srivastava\AppData\Loca
l\Programs\Python\Python310\python.exe' 'c:\Users\Samarth Srivastava\.vscode\extensions\ms-python.python-2022.10.1\pythonFiles\lib
\python\debugpy\adapter/../..\debugpy\launcher' '61181' '--' 'e:\Interpolation Paper\Codes\Python\BI.py'
154     5       -1      -1      3
159     4       -2      2
163     2       0
165     2
167
Value at 2 is 163.9616
PS E:\Interpolation Paper\Codes\Python>
```

Fig. 9: Value of V_2 Speed after applying Newton Forward Interpolation

4.3. Error in Interpolation

4.3.1. Newton Forward Interpolation

To estimate the error in Newton Forward Interpolation, we can generate v-speeds for a known temperature and compare the calculated values with the given values in the above-mentioned table. The known values of V_1, V_R, and V_2 speeds at 17 degrees Celsius for calm winds (winds not blowing) are 155, 162 and 163 respectively. On applying Newton Forward Interpolation, we get the V_1, V_R and V_2 speeds to be 155.7216, 162.6639 and 164.0585 knots respectively. Hence, the percentage error for V_1, V_R and V_2 is 0.46%, 0.40% and 0.649% respectively.

It is to be noted that the values of OAT taken for this calculation are taken in intervals of 8, starting from 13 to 77.

```
Enter the value of interpolation: 17
13.0    156.0   -2.0    0.0     0.0     3.0     -6.0    5.0     -1.0
21.0    154.0   -2.0    0.0     3.0     -3.0    -1.0    4.0     -2.0
29.0    152.0   -2.0    3.0     0.0     -4.0    3.0     2.0     7.0
37.0    150.0   1.0     3.0     -4.0    -1.0    5.0     9.0     -68.0
45.0    151.0   4.0     -1.0    -5.0    4.0     14.0    -59.0   141.0
53.0    155.0   3.0     -6.0    -1.0    18.0    -45.0   82.0    -129.0
61.0    158.0   -3.0    -7.0    17.0    -27.0   37.0    -47.0   57.0
69.0    155.0   -10.0   10.0    -10.0   10.0    -10.0   10.0    -10.0
77.0    145.0   0       0       0       0       0       0       0
value at  17.0  is  155.72162328371815
```

Fig. 10: Error in V_1 speed using Newton Forward Interpolation

```
Enter the value of interpolation: 17
13.0    163.0   -2.0    1.0     -2.0    4.0     -5.0    3.0     0.0
21.0    161.0   -1.0    -1.0    2.0     -1.0    -2.0    3.0     1.0
29.0    160.0   -2.0    1.0     1.0     -3.0    1.0     4.0     2.0
37.0    158.0   -1.0    2.0     -2.0    -2.0    5.0     6.0     -52.0
45.0    157.0   1.0     0.0     -4.0    3.0     11.0    -46.0   110.0
53.0    158.0   1.0     -4.0    -1.0    14.0    -35.0   64.0    -101.0
61.0    159.0   -3.0    -5.0    13.0    -21.0   29.0    -37.0   45.0
69.0    156.0   -8.0    8.0     -8.0    8.0     -8.0    8.0     -8.0
77.0    148.0   0       0       0       0       0       0       0
value at  17.0  is  162.66393213192896
```

Fig. 11: Error in V_R speed using Newton Forward Interpolation

Fig. 12: Error in V_2 speed using Newton Forward Interpolation

4.3.2. Bessel interpolation

To estimate the error in Bessel Interpolation, we can generate v-speeds for a known temperature and compare the calculated values with the given values in the above-mentioned table. The known values of V_1, V_R, and V_2 speeds at 17 degrees Celsius for calm winds (winds not blowing) are 155, 162 and 163 respectively. On applying Bessel Interpolation, we get the V_1, V_R and V_2 speeds to be 155, 162 and 163 knots respectively.

```
143     6       0       -4      8       -173
149     6       -4      4       -165
155     2       0       -161
157     2       -161
159     -159
0
Value at 0 is 155.0
PS E:\Interpolation Paper\NTCC\Codes\Python>
```

Fig. 13: Error in V_1 speed using Bessel Interpolation

```
153     4       1       -4      7       -178
157     5       -3      3       -171
162     2       0       -168
164     2       -168
166     -166
0
Value at 0 is 162.0
PS E:\Interpolation Paper\NTCC\Codes\Python>
```

Fig. 14: Error in V_R speed using Bessel Interpolation

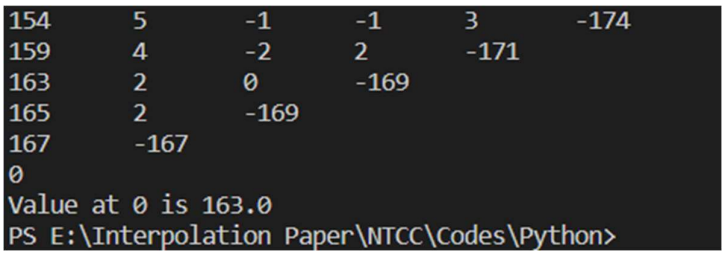

Fig. 15: Error in V_R speed using Bessel Interpolation

Hence, for Bessel Interpolation, the error is 0% for all the 3 V-speeds.

5. Conclusion:

The method to calculate takeoff speed using suitable interpolation presented in this paper is highly accurate with negligible errors of 0.46%, 0.40% and 0.649%. Due to the high level of precision, the proposed methodology for calculating takeoff speeds can be implemented without any safety concerns. Future research will investigate the accuracy of curve fitting to calculate the takeoff speeds using RTOW charts.

References

1. Yajaun Zhu, Jiangfeng Wang, Yongliang Chen and Yizhao Wu (2016), Calculation of Takeoff and Landing Performance under different environments, not. J. Mod. Phys. Conf. Ser..
2. Soni, N. Kumar, Y. K. Sharma, V. Kumar and A. Aggarwal, "Generalization of Fourier Transformation of Scaling Function using Riesz basis on L2 (K)," *2022 10th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions) (ICRITO)*, Noida, India, 2022, pp. 1-5, doi: 10.1109/ICRITO56286.2022.9965138.
3. Christopher M. Wynnyk (2012), Wind Analysis in Aviation Applications, 31st Digital Avionics Systems Conference.
4. Roberto Salvador Félix Patrón and Ruxandra Mihaela Botez (2013), Low calculation time interpolation method on the altitude optimization algorithm for the FMS CMA-9000 improvement on the A310 and L-1011 aircraft, American Institute of Aeronautics and Astronautics.
5. Sastry S.S. (2012), Introductory Methods of Numerical Analysis, PHI Learning Private Limited, New Delhi 11001.
6. ICAO Annex 6 (Operation of Aircraft), Tenth Edition, July 2018.
7. Rana Shubham, (25th May 2021), geeksforgeeks https://www.geeksforgeeks.org/bessels-interpolation/
8. Rana Shubham (21st July 2022), geeksforgeeks, https://www.geeksforgeeks.org/newton-forward-backward-interpolation/
9. Soni, N. Kumar, V. Kumar and A. Aggarwal, "Biorthogonality Collection of Finite System of Functions in Multiresolution Analysis on L2(K)," *2022 10th International Conference*

on Reliability, Infocom Technologies and Optimization (Trends and Future Directions) (ICRITO), Noida, India, 2022, pp. 1-5, doi: 10.1109/ICRITO56286.2022.9964791.

10. Soni, N. Kumar, A. Aggarwal and S. Aggarwal, "Characterization of Dual Multiresolution Analysis by Orthogonality of System of Functions: An application to communication engineering," *2022 Fourth International Conference on Emerging Research in Electronics, Computer Science and Technology (ICERECT)*, Mandya, India, 2022, pp. 1-5, doi: 10.1109/ICERECT56837.2022.10060271.
11. Soni, N. Kumar, A. Aggarwal and S. Aggarwal, "Characterization of Dual Multiresolution Analysis by Orthogonality of System of Functions: An application to communication engineering," *2022 Fourth International Conference on Emerging Research in Electronics, Computer Science and Technology (ICERECT)*, Mandya, India, 2022, pp. 1-5, doi: 10.1109/ICERECT56837.2022.10060271.