PyChart -A Python module for analysis and visual view of 2D/3D Charts

PyChart -A Python module for analysis and visual view of 2D/3D Charts

Dorian Nedelcu
Babeş-Bolyai University
96 PUBLICATIONS **306** CITATIONS

Tihomir Latinovic
University “VITEZ” Travnik
153 PUBLICATIONS **229** CITATIONS

Journal of Physics: Conference Series

PAPER • OPEN ACCESS

PyChart – A Python module for analysis and visual view of 2D/3D Charts

To cite this article: D Nedelcu and T Latinovic 2021 *J. Phys.: Conf. Ser.* **1781** 012044

PyChart – A Python module for analysis and visual view of 2D/3D Charts

D Nedelcu[1] and T Latinovic[2]

[1]"Eftimie Murgu" University of Resita, Romania, Faculty of Engineering and Management, Department of Mechanical Engineering and Management, Traian Vuia square, no.1-4, 320085, Resita
[2]University of Banja Luka, Bosnia and Herzegovina, Faculty of Mechanical Engineering, Department of Mechatronics and Robotics, Stepe Stepanovica 71, 78000, Banja Luka

E-mail: d.nedelcu@uem.ro

Abstract. The paper describes the PyChart module (aimed at analysis and visual view of 2D/3D Charts), which was created with the help of free and Open Source resources, using Python as a programming language and wx.Python as a graphical user interface toolkit. The chart data is imported from a Excel/ CSV file with a template structure and is drawn in the PyChart module as XY or XYZ curves similar with Excel scatter with smooth lines and markers style. The main functions of the charts are activated using the toolbar. The module is provided with zooming instruments (fit, pan, zoom in, zoom out), cubic spline curves interpolation, chart intersection with constant X, Y or Z values, visual follow of the 2D chart points to view coordinates, export of data in Windows Clipboard, Excel or Microsoft Word format and saving the chart as a image file.

1. Introduction

There are some software packages on the market for 2D or 3D curves representation, that can be accesed on-line [1-4] or as a standalone application [5-8]. Why create a new package? The answer can be synthesized by the following considerations:

- a challenge to create a 2D/3D chart module using only Open Source and free resources;
- to obtain an engineering and practical tool for exploring 2D/3D curves;
- to intersect 2D / 3D curves with X / Y or X / Y / Z values and obtain numerical intersection coordinates;
- to have a module which can be extended with the future necessities or included in other future software.

2. The software packages

The PyChart software is created with the following free and Open Source resources, Figure 1:

- **Python** – a high-level programming language [9]. Python runs on Windows, Linux/Unix, Mac OS X, and has been ported to the Java and .NET virtual machines. Python is free to use, even for commercial products, because of its OSI-approved open source license. Python is an interpreted, interactive, object-oriented programming language. It incorporates modules, exceptions, dynamic typing, very high level dynamic data types, and classes. Python combines remarkable power with very clear syntax. It has interfaces to many system calls and libraries, as well as to various window systems, and is extensible in C or C++.

- **wxPython** – is a graphical user interface toolkit (G.U.I.) for the Python programming language [10]. It allows Python programmers to create programs with a robust, highly functional graphical user interface, simply and easily. It is implemented as a Python extension module (native code) that wraps the popular wxWidgets cross platform G.U.I. library, which is written in C++. Like Python and wxWidgets, wxPython is Open Source which means that it is free for anyone to use and the source code is available for anyone to look at and modify. wxPython is a cross-platform toolkit. This means that the same program will run on multiple platforms without modification [11].
- **Matplotlib** – is a comprehensive library for creating static, animated, and interactive visualizations in Python. Its variety of output formats, several chart types, and capability to run interactively or non-interactively makes Matplotlib suitable for use in many different situations [12].
- **NumPy** – is the fundamental package for scientific computing with Python. It contains among other things: a powerful N-dimensional array object, sophisticated functions, tools for integrating C/C++ and Fortran code, useful linear algebra, Fourier transform, and random number capabilities [13].
- **SQLite** – is a library [14] that implements a self-contained, serverless, zero-configuration, transactional SQL database engine. SQLite is the most widely deployed SQL database engine in the world, with the source code launched in the public domain.

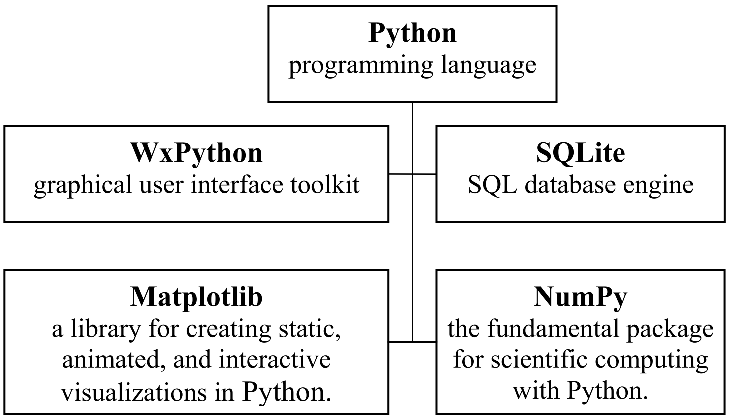

Python
programming language
WxPython
graphical user interface toolkit
SQLite
SQL database engine
Matplotlib
a library for creating static, animated, and interactive visualizations in Python.
NumPy
the fundamental package for scientific computing with Python.

Figure 1. The software packages

3. The Input Data Format File

The input data is imported from a Excel or CSV file with the following template structure:

- the first line contains the number of curves followed by the chart title, in two cells for the Excel file or separated by a comma in the CSV file;
- the second line contains the X and Y labels (for 2D chart) or X, Y and Z labels (for 3D chart) of the chart, in two / three cells for the Excel file or separated by a comma in the CSV file;
- a number of lines equal to the number of curves, containing the curve parameter label followed by the number of points of the curve, in two / three cells for the Excel file or separated by a comma in the CSV file;
- for every curve, the file format continues with:
  - one empty line;
  - a number of lines equal to the number of points of the curve, containing X and Y values (for 2D chart) or X, Y and Z values (for 3D chart) for every point, in two (for 2D chart) or three (for 3D chart) cells for the Excel file or separated by a comma in the CSV file.

4. The PyChart main window

The PyChart main window is presented in Figure 2 for the 2D chart and in Figure 3 for the 3D chart.

- the application title bar – contains the application name, version and name of the loaded file;
- the application toolbar– contains icons for the main functions of the application;
- the application chart area – reserved for the drawing of chart components;
- the application status bar – reserved for the application's messages.

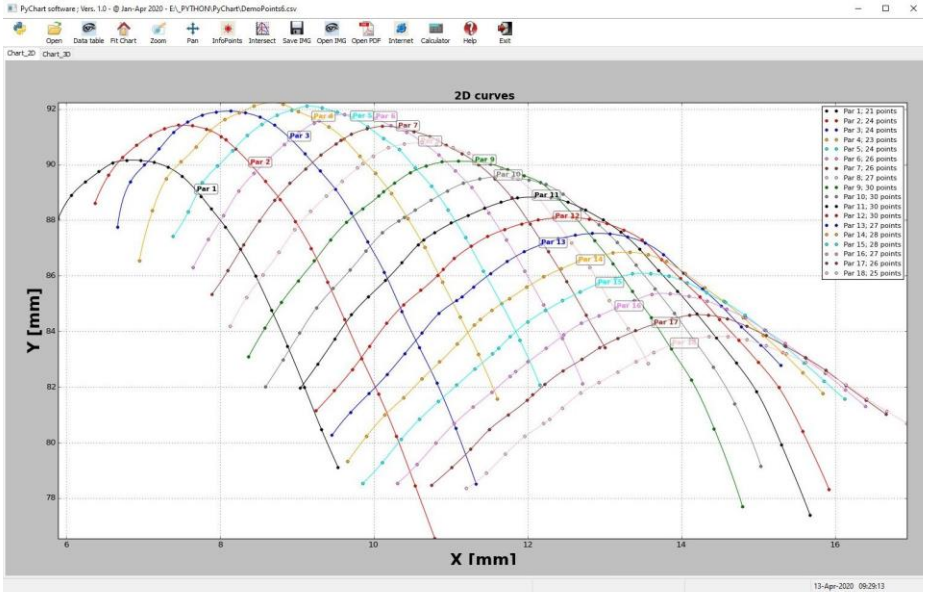


Figure 2. The **PyChart** main window for the 2D chart

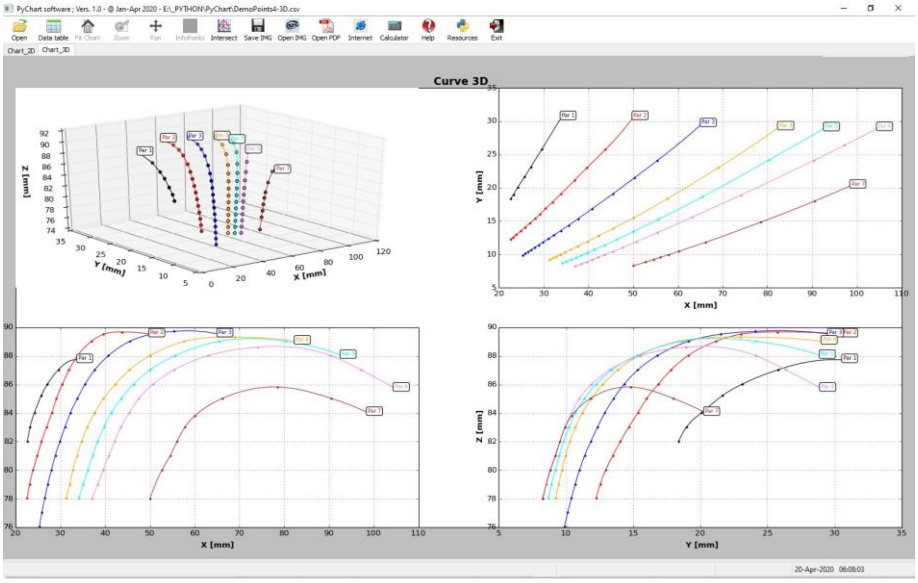


Figure 3. The **PyChart** main window for the 3D chart

5. The PyChart interface

The PyChart toolbar icons are presented in Table 1.

Table 1. The PyChart functions

No.	Toolbar Icon	Icon name	Description
1		**Open**	Open the input data file in Excel or CSV format
2		**Data Table**	Show the numerical values of the input data file in a table, copy to clipboard , export to Excel , Word or create a new chart with selected cells from the table
3		**Fit Chart**	Zoom chart to fit into main window (for 2D Chart)
4		**Zoom**	Zoom rectangular area of the chart (for 2D Chart)
5		**Pan/Zoom**	Pan or zoom the chart (for 2D Chart)
6		**InfoPoints**	Identify, visualize chart points and show the numerical values in the status bar (for 2D Chart)
7		**Intersect**	Chart intersection with X/Y or X/Y/Z constant values through Chart Intersect window
8		**Save IMG**	Save the chart as a JPG image file
9		**Open IMG**	Open an image file (JPG, PNG, TIF, BMP)
10		**Open PDF**	Open a PDF file
11		**Internet**	Access the Internet through Windows' default browser
12		**Calculator**	Open the window calculator for simple calculations
13		**Help**	Accesing the Python Chart help
14		**Resources**	Show information about the free and Open Source resources used by the PyChart software and send to the proper site page on the Internet
15		**Exit**	Exit the application

6. The PyChart Data Table

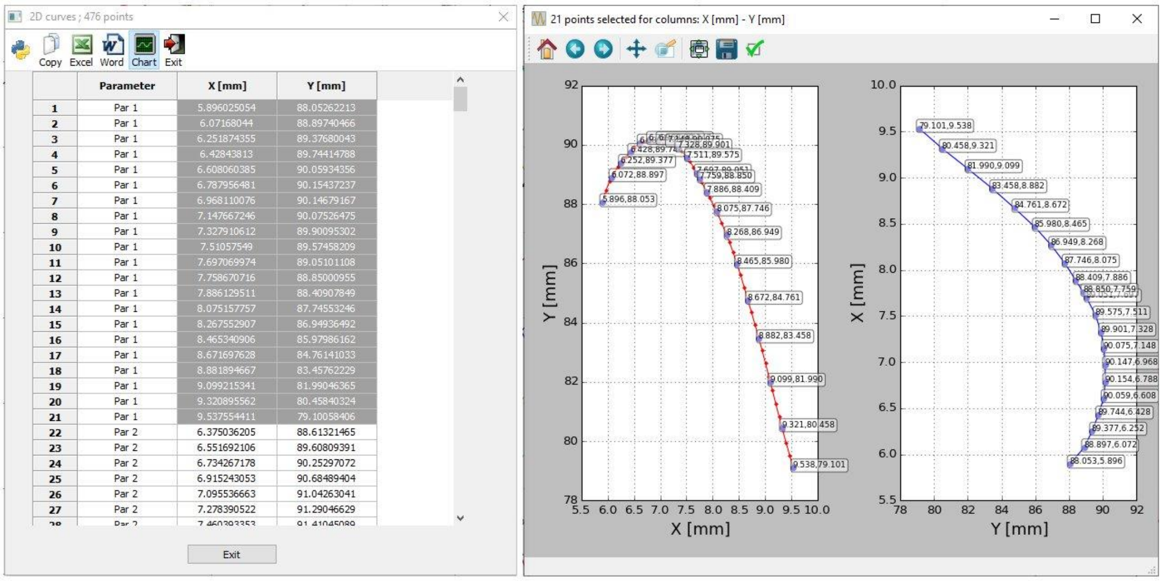
The Data Table window is presented in Figure 4. The toolbar contains icons for exporting the coordinates to Windows clipboard, Microsoft Excel or Microsoft Word, creating a new 2D chart, y=f(x) and x=f(y), based on selected cells.

7. The PyChart Zooming Tools

The **Fit Chart** command is activated through the toolbar icon , which is active only for 2D charts. This command places the entire chart area in the current application window.

The **Zoom** command is activated through the toolbar icon , which is active only for 2D charts. This command shows a chart region defined by two mouse clicks.

The **Pan/Zoom** command is activated through the toolbar icon , which is active only for 2D charts. This command shows a chart region defined by two mouse clicks. Press the left mouse button and hold it to pan the chart and drag it to a new position; pressing x or y while panning the motion will be constrained to the x or y axis, respectively; press the right mouse button to zoom; the x axis will be zoomed in proportionately to the rightward movement and zoomed out proportionately to the leftward movement.



2D curves ; 476 points

	Parameter	X [mm]	Y [mm]
1	Par 1	5.896025054	88.05262213
2	Par 1	6.07168044	88.89740466
3	Par 1	6.251874355	89.37680043
4	Par 1	6.42843813	89.74414788
5	Par 1	6.608060385	90.05934356
6	Par 1	6.787956481	90.15437237
7	Par 1	6.968110076	90.14679167
8	Par 1	7.147667246	90.07526475
9	Par 1	7.327910612	89.90095302
10	Par 1	7.51057549	89.57458209
11	Par 1	7.697069974	89.05101108
12	Par 1	7.758670716	88.85000955
13	Par 1	7.886129511	88.40907849
14	Par 1	8.075157757	87.74553246
15	Par 1	8.267552907	86.94936492
16	Par 1	8.465340906	85.97986162
17	Par 1	8.671697628	84.76141033
18	Par 1	8.881894667	83.45762229
19	Par 1	9.099215341	81.99046365
20	Par 1	9.320895562	80.45840324
21	Par 1	9.537554411	79.10058406
22	Par 2	6.375036205	88.61321465
23	Par 2	6.551692106	89.60809391
24	Par 2	6.734267178	90.25297072
25	Par 2	6.915243053	90.68489404
26	Par 2	7.095536663	91.04263041
27	Par 2	7.278390522	91.29046629
28	Par 2	7.460393353	91.41045089

21 points selected for columns: X [mm] - Y [mm]

Figure 4. The **PyChart Data Table** and the chart of selected cells

8. The PyChart InfoPoints command

The **InfoPoints** command is activated through the toolbar icon ☀, which is active only for 2D charts. This command will generate a special marker which follows the mouse position by placing the marker to the nearest chart point; also, the status bar will show the ID curve and ID point, the parameter of the curve and X/Y coordinates of the marked point. A middle click mouse will exit the **InfoPoints** command.

9. The PyChart Intersect window

The **Intersect** window is activated through the **Intersect** toolbar icon, Figure 5. For 2D charts that window is designated to generate the intersection points of the chart with X/Y constant values. To intersect the chart with X or Y values, the option must be selected from the left control list and an input numerical value is required in the associated text control; finally, the **Intersect** button ▣ will calculate the intersection points of the chart with the previous selected value. The **Copy** button ▯ will send the intersection points coordinates to Windows clipboard. The **Excel** button ▣ will send the intersection points coordinates to Microsoft Excel. The **Word** button W will send the intersection points coordinates to Microsoft Word. The **Erase** button ☒ will erase all the previous intersection points from the chart area and coordinate's table. The **Quit** button ➔ hides the **Intersect** window. The coordinate's table shows the ID, the intersected curve's parameter and the X/Y coordinates of the intersections points. The points resulted from the intersections are marked with distinct colour in the chart area. The values of the intersection points are calculated by cubic spline interpolations functions; the number of founded intersections points is placed on the title bar of the **Intersect** window.

For 3D charts another window is designated to generate the intersection points of the chart with X/Y/Z constant values, Figure 6. To intersect the chart with X, Y or Z values, the corresponding control must be selected and an input numerical value is required in the text control; finally, the **Intersect** button ▣ will calculate the intersection points of the chart with the previous selected value. The points resulted from the intersections are marked with distinct colour in the chart area and the curve passing through this points will be drawn.

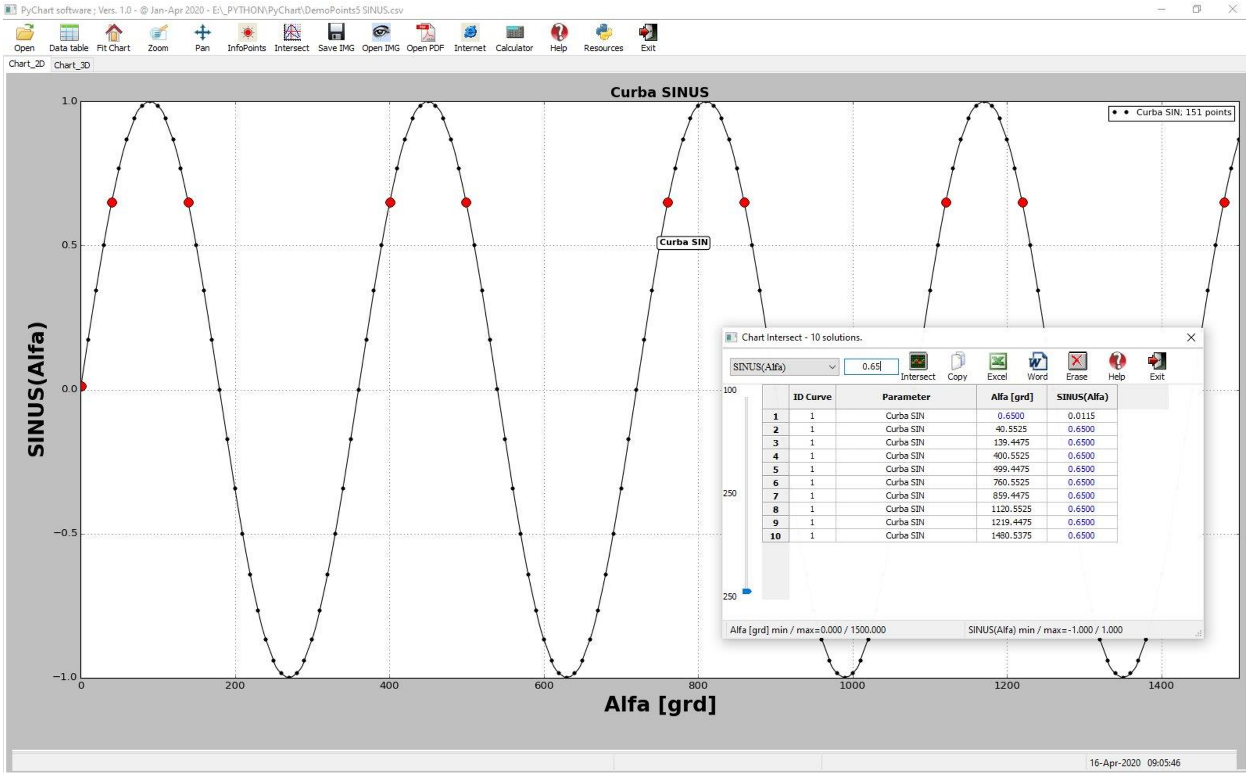


Chart Intersect - 10 solutions.

SINUS(Alfa) 0.65

	ID Curve	Parameter	Alfa [grd]	SINUS(Alfa)
1	1	Curba SIN	0.6500	0.0115
2	1	Curba SIN	40.5525	0.6500
3	1	Curba SIN	139.4475	0.6500
4	1	Curba SIN	400.5525	0.6500
5	1	Curba SIN	499.4475	0.6500
6	1	Curba SIN	760.5525	0.6500
7	1	Curba SIN	859.4475	0.6500
8	1	Curba SIN	1120.5525	0.6500
9	1	Curba SIN	1219.4475	0.6500
10	1	Curba SIN	1480.5375	0.6500

Alfa [grd] min / max=0.000 / 1500.000 SINUS(Alfa) min / max=-1.000 / 1.000

Figure 5. The **INTERSECT** window for 2D charts

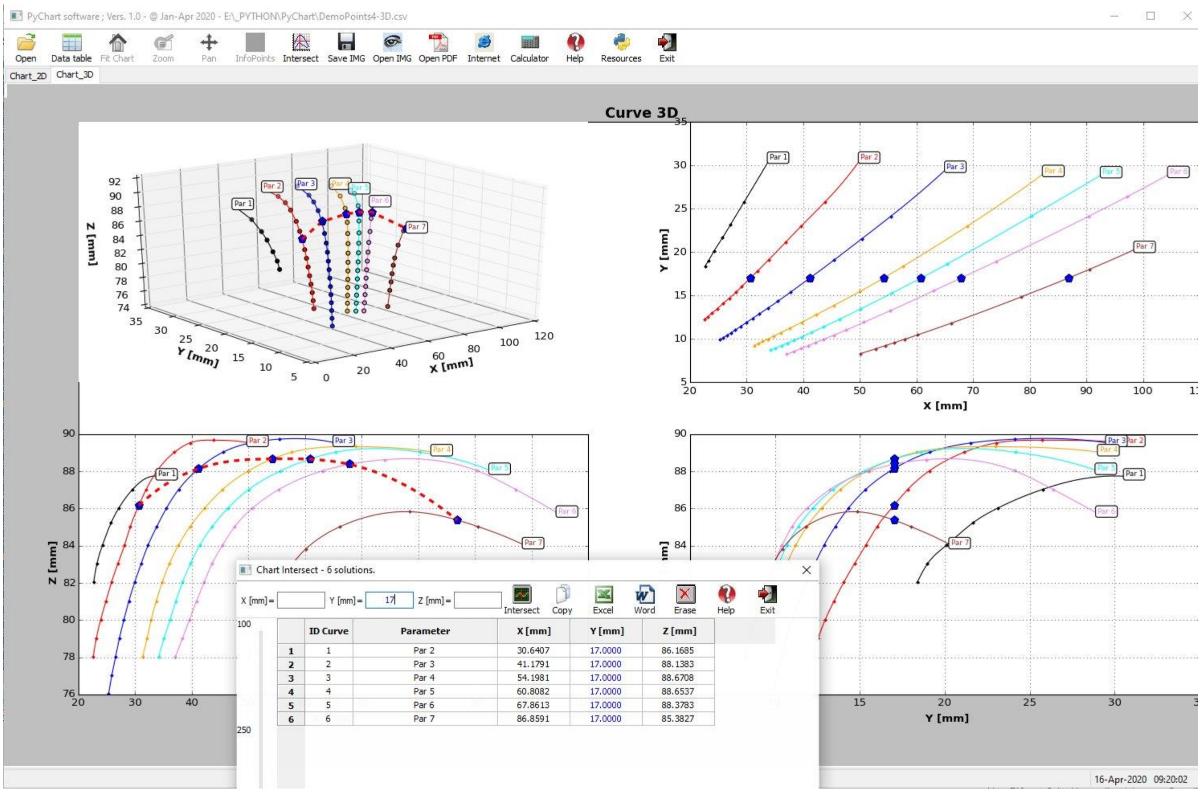


Chart Intersect - 6 solutions.

X [mm]= Y [mm]= 17 Z [mm]=

	ID Curve	Parameter	X [mm]	Y [mm]	Z [mm]
1	1	Par 2	30.6407	17.0000	86.1685
2	2	Par 3	41.1791	17.0000	88.1383
3	3	Par 4	54.1981	17.0000	88.6708
4	4	Par 5	60.8082	17.0000	88.6537
5	5	Par 6	67.8613	17.0000	88.3783
6	6	Par 7	86.8591	17.0000	85.3827

Figure 6. The **INTERSECT** window for 3D charts

10. The programme description and programming tools

The programme consists of ~3200 instructions, grouped in the following main classes:

- **PyChart** – a class for 2D or 3D chart drawing: drawing normal and spline curves, normal markers, point's coordinates values, grid lines, grid values, axis, X, Y and Z labels, title chart;
- **Intersect** – a class for 2D chart intersections with X or Y values, which includes functions for: spline calculations, points export to Windows clipboard, Microsoft Excel or Microsoft Word, deletion of the previous calculated points;
- **Intersect_3d**– a class for 3D chart intersections with X,Y or Z values, which includes functions for: spline calculations, points export to Windows clipboard, Microsoft Excel or Microsoft Word, deletion of the previous calculated points;
- **BIBLIO** – a global library with functions for spline system calculation and spline intersection with a constant value;
- **Table_Grid** – a class that shows the numerical values of the input data file in a table, copies to Windows clipboard, exports to Excel and Word or creates a new chart with selected cells from the table;
- other specific classes and functions.

Python was the chosen programming language to create the application. The wxPython module was used to create the frames and all Windows controls: toolbar, individuals controls (text, buttons, lists, etc.). Python is a remarkably powerful dynamic programming language that is used in a wide variety of application domains. Python doesn't come prepackaged with Windows. It's not quite a simple task to install the desired version together with required dependencies; the easy way is to use the "Python(x,y)" package, a free scientific and engineering development software for numerical computations, data analysis and data visualization based on the Python programming language. The download file is "Python(x,y)-2.7.10.0.exe" with a 808 MB size [15].

The application was written in UliPad, a free wxPython powered, programmer oriented and flexible editor, with many features such as class browser, code auto-complete, HTML viewer, directory browser, wizard and many others [16].

The SQLite module was used to create a database to store additional elements of the software: icon images, help file contents, text files, additional image files. To manage the application database the SQLite Expert was used [17], a powerful visual tool that enables to easily administer the SQLite databases. SQLite Expert integrates database management and maintenance into a single, seamless environment, with a clear and intuitive user interface.

11. Conclusions

The developed application is designated to visualize and explore 2D and 3D curves, including constant X, Y or Z chart intersection calculated through spline interpolation functions. One of the application's goals: to use only free and Open Source resources, was achieved through the Python language and its associated modules wxPython, matplotlib, NumPy, SQLite. The code is available (for download and run) as dataset at Mendeley [18].

References

[1] ***https://www.onlinecharttool.com/graph?selected_graph=scatter
[2] ***https://www.canva.com/graphs/
[3] ***https://www.visme.co/graph-maker/
[4] ***https://imclient.herokuapp.com/chartsbuilder/?charttype=3d
[5] ***https://www.edrawsoft.com/download-edrawmax.html
[6] ***https://www.originlab.com/
[7] ***https://www.microsoft.com/en-us/garage/profiles/charts-3d/
[8] ***https://www.highcharts.com/products/highcharts/
[9] ***(http://www.python.org/about/apps/).
[10] ***http://www.wxpython.org

[11] Rappin N and Dunn R 2006 *wxPython in Action*, ISBN 1-932394-62-1, Manning Publications Co., USA

[12] Hunter J D 2007 Matplotlib: A 2D Graphics Environment, *Computing in Science & Engineering* **9**(3) 90-95

[13] ***https://numpy.org/

[14] Owens M 2006 *The Definitive guide to SQLite*, ISBN-13: 978-1-59059-673-9, Apress, USA

[15] ***https://python-xy.github.io/

[16] ***http://code.google.com/p/ulipad/

[17] ***http://www.sqliteexpert.com/

[18] Nedelcu D and Latinovic T 2020 *PyChart_Python_Application*, Mendeley Data, V1, DOI: http://dx.doi.org/10.17632/35jrmwd4fw.1