Python tools for Excel

Release 0.0.2b0

May 31, 2020

Contents

1 Calculation (evaluating Excel formulas in Python) 3
- 1.1 Formulas . . . 3
- 1.2 Koala . . . 7
- 1.3 Pandas . . . 9
- 1.4 PyCel . . . 9
- 1.5 xlcalculator . . . 10

2 File Tools 13
- 2.1 openpyxl . . . 13
- 2.2 Pandas . . . 14
- 2.3 Pyxlsx . . . 15
- 2.4 xlrd . . . 15
- 2.5 XlsxWriter . . . 15
- 2.6 xlutils . . . 17
- 2.7 xlwt . . . 17

3 Formula Libraries 19
- 3.1 xlFormulas . . . 19
- 3.2 xlfunctions . . . 19

4 Integrations 21
- 4.1 DataNitro . . . 21
- 4.2 ExcelPython . . . 21
- 4.3 ExPy . . . 21
- 4.4 pywin32 . . . 22
- 4.5 PyXll . . . 22
- 4.6 xlwings Community Edition (CE) . . . 23
- 4.7 xlwings Pro . . . 24

5 Integrations with Calculation 25
- 5.1 FlyingKoala . . . 25
- 5.2 PyCel with PyXll . . . 26

6 Microsoft Add-ins 27
- 6.1 funfun . . . 27
- 6.2 XLMiner Analysis ToolPak . . . 27
- 6.3 Blockspring . . . 27

6.4 price f(x) . . . 27
6.5 Scott's Xero . . . 27

7 Blog posts **29**
7.1 Joel Spolsky . . . 29
7.2 Dirk Gorissen . . . 29
7.3 Vincenzo Arcidiacono . . . 29
7.4 Steve Holden . . . 30

8 Non-Excel tools **31**
8.1 ModelX . . . 31

9 Indices and tables **33**

This is a software review for software relating to integrations for, and interaction between, Python and Excel.

In the context of this document a software review is much like a "literature review" as found at the beginning of a thesis. Largely it's an exploration of the knowledge domain cataloging what can be found and what the state of knowledge is at the moment of writing. This is quite different to a road-test which is often what is meant by a review.

Tabulating such data often doesn't provide much genuine assistance. To a large extent these tools are found in the open source domain and the motivations of people creating solutions in that space are often addressing a specific "personal" need rather than necesarily considering the development of a product. With that consideration very few of these tools are intended to be precise replicas of each other so comparison tables would be sparse and nothing more than a distraction.

I usually find there is enough value in collating information about the software that exists, grouping the software solutions into categories for comprable solutions and specifying the feature-sets (as advertised). I have found this kind of documentation can provide enough value to enable people to make decisions on whether a particular solution fits what they are looking for. It's rare to get many more than five or six solutions in the same problem domain or addressing the same problem.

CHAPTER 1

Calculation (evaluating Excel formulas in Python)

These tools help calculate the result of an Excel equation within Python, using the functions as defined in an Excel file without the need for Excel to be installed.

1.1 Formulas

Formulas has come from an engieering background.

Formulas homepage

Licence:

formulas implements an interpreter for Excel formulas, which parses and compile Excel formulas expressions.

Moreover, it compiles Excel workbooks to python and executes without using the Excel COM server. Hence, Excel is not needed.

Supported functions;

- AND
- ARABIC
- ASIN
- ASINH
- ATAN
- ATAN2
- ATANH
- AVERAGE
- AVERAGEA
- AVERAGEIF

- BIN2DEC
- CEILING
- CEILING.MATH
- CEILING.PRECISE
- COLUMN
- CONCAT
- CONCATENATE
- COS
- COSH
- COT
- COTH
- COUNT
- COUNTA
- COUNTBLANK
- COUNTIF
- CSC
- CSCH
- DATEVALUE
- DAY
- DEC2BIN
- DEC2HEX
- DEC2OCT
- DECIMAL
- DEGREES
- EVEN
- EXP
- FACT
- FACTDOUBLE
- FALSE
- FIND
- FLOOR
- FLOOR.MATH
- FLOOR.PRECISE
- GCD
- HEX2DEC
- HLOOKUP

- HOUR
- IF
- IFERROR
- INDEX
- INT
- IRR
- ISERR
- ISERROR
- ISO.CIELING
- LARGE
- LCM
- LEFT
- LEN
- LOG10
- LOG
- LOOKUP
- LOWER
- LN
- MATCH
- MINUTE
- MAX
- MID
- MIN
- MOD
- MONTH
- MROUND
- NA
- NOT
- NOW
- NPV
- OCT2DEC
- ODD
- OR
- PI
- POWER
- RADIANS

- RAND
- RANDBETWEEN
- REPLACE
- RIGHT
- ROMAN
- ROUND
- ROUNDDOWN
- ROUNDUP
- ROW
- SEC
- SECH
- SECOND
- SIGN
- SIN
- SINH
- SMALL
- SQRT
- SQRTPI
- SUMPRODUCT
- SUM
- SUMIF
- SWITCH
- TAN
- TANH
- TODAY
- TIME
- TIMEVALUE
- TRIM
- TRUE
- TRUNC
- UPPER
- VLOOKUP
- XIRR
- XNPV
- XOR
- YEAR

- YEARFRAC

1.2 Koala

Koala homepage

Licence:

Koala converts any Excel workbook into a python object that enables on the fly calculation without the need of Excel.

Koala parses an Excel workbook and creates a network of all the cells with their dependencies. It is then possible to change any value of a node and recompute all the depending cells.

You can read more on the origins of Koala here.

Supported Functions;

- ALL
- AND
- ARRAY
- ARRAYROW
- ATAN2
- AVERAGE
- CHOOSE
- COLUMNS
- CONCAT
- CONCATENATE
- COUNT
- COUNTA
- COUNTIF
- COUNTIFS
- DATE
- EOMONTH
- GAMMALN
- IF
- IFERROR
- INDEX
- IRR
- ISBLANK
- ISNA
- ISTEXT
- LINEST
- LOG

- LOOKUP
- LN
- MATCH
- MAX
- MID
- MIN
- MOD
- MONTH
- NPV
- OFFSET
- OR
- PI
- PMT
- POWER
- RAND
- RANDBETWEEN
- RIGHT
- ROUND
- ROUNDUP
- ROWS
- SLN
- SQRT
- SUM
- SUMIF
- SUMIFS
- SUMPRODUCT
- TAN
- TODAY
- VALUE
- VDB
- VLOOKUP
- XIRR
- XLOG
- XNPV
- YEAR
- YEARFRAC

1.3 Pandas

Pandas does not do this. To do this you need to write code to read the functions and map them to Pandas, numpy, numpy.finance or scipy functions which is the service the other solutions in this category offer.

1.4 PyCel

PyCel Homepage

Licence:

Pycel is a small python library that can translate an Excel spreadsheet into executable python code which can be run independently of Excel.

The python code is based on a graph and uses caching & lazy evaluation to ensure (relatively) fast execution. The graph can be exported and analyzed using tools like Gephi. See the contained example for an illustration.

The full motivation behind pycel including some examples & screenshots is described in this blog post.

It's possible to run pycel as an excel addin using PyXLL.

Supported Functions;

- Abs
- Atan2
- Average
- Averageif
- Averageifs
- Cieling
- Cieling.Math
- Cieling.Precise
- Count
- Countif
- CountIfs
- Even
- Fact
- FactDouble
- Floor
- Floor.math
- Floor.precise
- Int
- IsErr
- IsError
- IsEven
- IsText

- IsNa
- IsOdd
- IsNumber
- Large
- Linest
- Ln
- Log
- Max
- Maxifs
- Min
- Minifs
- Mod
- NPV
- Odd
- Power
- Round
- Rounddown
- Roundup
- Sign
- Small
- Sum
- Sumif
- Sumifs
- SumProduct
- Trunc

1.5 xlcalculator

xlcalculator homepage

Licence:

xlcalculator converts a given Excel workbook into a Python object (model) that enables calculation (evaluation) without the need of Excel.

It uses the xlfunctions library for the Python implementation fo Excel functions.

- Loading an Excel file into a Python compatible state
- Saving Python compatible state
- Loading Python compatible state
- Ignore worksheets

- Extracting sub-portions of a model. “focussing” on provided cell addresses or defined names
- Evaluating cells, ranges defined names and formulas

CHAPTER 2

File Tools

There are a number of tools for Python which help manage reading and writing Excel files. Most have been around for a long time and are mature.

The below list of them came from the website http://www.python-excel.org/.

2.1 openpyxl

openpyxl homepage

Licence: MIT/Expat

Openpyxl is a Python library to read/write Excel 2010 xlsx/xlsm/xltx/xltm files.

It was born from lack of existing library to read/write natively from Python the Office Open XML format.

It is a comprehensive library to create, modify and save Excel files using operations akin to Excel itself.

Features:

- Cells
- Charts
- Chartsheets
- Comments
- Descriptors
- Drawing
- Formatting
- Pivot Tables
- Read Excel files
- Styles

- Workbook operations
- Worksheet operations
- Write Excel files

2.2 Pandas

pandas homepage

Licence:

pandas is an open source, BSD-licensed library providing high-performance, easy-to-use data structures and data analysis tools for the Python programming language.

It is comprehensive for data analyisis and plays well with numpy and scipy. It facilitates a huge range of operations for data analysis. The focus of this section of this review is Excel file operations so I will only list the related aspect. pandas may well be mentioned elsewhere in this review if it does things related to that section.

The read_excel() method can read Excel 2003 (.xls) files using the xlrd Python module. Excel 2007+ (.xlsx) files can be read using either xlrd or openpyxl. Binary Excel (.xlsb) files can be read using pyxlsb. The to_excel() instance method is used for saving a DataFrame to Excel. Generally the semantics are similar to working with csv data. See the cookbook for some advanced strategies.

To write a DataFrame object to a sheet of an Excel file, you can use the to_excel instance method. The arguments are largely the same as to_csv, the first argument being the name of the excel file, and the optional second argument the name of the sheet to which the DataFrame should be written. Files with a .xls extension will be written using xlwt and those with a .xlsx extension will be written using xlsxwriter (if available) or openpyxl.

Features:

- IO tools (xls, xlsx, xlsb, text, CSV, HDF5, . . .)
  - Excel Files
    - Reading Excel files
      - Specifying sheets
      - Reading a MultiIndex
      - Parsing specific columns
      - Parsing dates
      - Cell converters
      - Dtype specifications
    - Writing Excel files
      - Writing Excel files to disk
      - Writing Excel files to memory
      - Excel writer engines
      - Style and formatting

2.3 Pyxlsx

pyxlsx homepage

Licence:

A package to read/write xlsx worksheet like dictionary, based on openpyxl.

- Create a new xlsx file and write to it
- Open an existing xlsx file
- Append rows to a worksheet
- Read from / write to a worksheet by row
- Read from a worksheet by column
- Read cell directly from Worksheet, Header, ContentRow
- Read adjacent cells of a certain cell

2.4 xlrd

xlrd homepage

Licence:

xlrd is a library for reading data and formatting information from Excel files, whether they are .xls or .xlsx files.

- Handling of Unicode
- Dates in Excel spreadsheets
- Named references, constants, formulas, and macros
- Formatting information in Excel Spreadsheets
- Loading worksheets on demand
- XML vulnerabilities and Excel files
- API Reference

2.5 XlsxWriter

xlsxwriter homepage

Licence:

XlsxWriter is a Python module that can be used to write text, numbers, formulas and hyperlinks to multiple worksheets in an Excel 2007+ XLSX file. It supports features such as formatting and many more, including:

- 100% compatible Excel XLSX files.
- Full formatting.
- Merged cells.
- Defined names.
- Charts.

- Autofilters.
- Data validation and drop down lists.
- Conditional formatting.
- Worksheet PNG/JPEG/BMP/WMF/EMF images.
- Rich multi-format strings.
- Cell comments.
- Textboxes.
- Integration with Pandas.
- Memory optimization mode for writing large files.

It supports Python 2.7, 3.4+ and PyPy and uses standard libraries only.

- Workbook
- Worksheet
- Worksheet (Page Setup)
- Format
- The Chart
- The Chartsheet
- Working with Cell Notation
- Working with and Writing Data
- Working with Formulas
- Working with Dates and Time
- Working with Colors
- Working with Charts
- Working with Object Positioning
- Working with Autofilters
- Working with Data Validation
- Working with Conditional Formatting
- Working with Worksheet Tables
- Working with Textboxes
- Working with Sparklines
- Working with Cell Comments
- Working with Outlines and Grouping
- Working with Memory and Performance
- Working with VBA Macros
- Working with Python Pandas and XlsxWriter

2.6 xlutils

xlutils homepage

Licence:

This package provides a collection of utilities for working with Excel files. Since these utilities may require either or both of the xlrd and xlwt packages, they are collected together here, separate from either package. The utilities are grouped into several modules within the package, each of them is documented below:

- xlutils.copy
  - Tools for copying xlrd.Book objects to xlwt.Workbook objects.
- xlutils.display
  - Utility functions for displaying information about xlrd-related objects in a user-friendly and safe fashion.
- xlutils.filter
  - A mini framework for splitting and filtering existing Excel files into new Excel files.
- xlutils.margins
  - Tools for finding how much of an Excel file contains useful data.
- xlutils.save
  - Tools for serializing xlrd.Book objects back to Excel files.
- xlutils.styles
  - Tools for working with formatting information expressed the styles found in Excel files.
- xlutils.view
  - Easy to use views of the data contained in a workbook's sheets.

2.7 xlwt

xlwt homepage

Licence:

xlwt is a library for writing data and formatting information to older Excel files (ie: .xls)

- add_sheet
- save
- write

Formatting

- Number format
- Font
- Alignment
- Border
- Background
- Protection

CHAPTER 3

Formula Libraries

These are libraries which deal with Excel formulas in the Python space. In the case of xlFormulas, it's supporting Excel style formulas in pandas. In the case of xlfunctions, it's defining Excel functions in Python.

3.1 xlFormulas

xlFormulas homepage

Licence:

Helper class to write Excel-style formula strings to worksheets when saving from a Pandas dataframe.

Pass in mathematical operators with strings, limited support currently for Excel built-in functions.

3.2 xlfunctions

xlfunctions homepage

Licence:

A collection of classes which implement functions as used in Microsoft Excel. The intent is to be a definitive library to support evaluating Excel calculations.

- ABS
- AVERAGE
- CHOOSE
- CONCAT
- COUNT
- COUNTA
- DATE

- IRR
- LN
- MAX
- MID
- MIN
- MOD
- NPV
- PMT
- POWER
- RIGHT
- ROUND
- ROUNDDOWN
- ROUNDUP
- SLN
- SQRT
- SUM
- SUMPRODUCT
- TODAY
- VLOOKUP
- XNPV
- YEARFRAC

CHAPTER 4

Integrations

4.1 DataNitro

According to Tony Roberts of PyXll fame in his blog post of August 2018, DataNitro is no longer under active development and is not available to license anymore.

4.2 ExcelPython

ExcelPython homepage

ExcelPython is a lightweight COM library which enables you to call Python code and manipulate Python objects from Excel VBA (or indeed any language supporting COM).

https://www.codeproject.com/Articles/639887/Calling-Python-code-from-Excel-with-ExcelPython

4.3 ExPy

ExPy homepage

Licence: BSD-3

The ExPy add-in allows easy use of Python directly from within an Microsoft Excel spreadsheet, both to execute arbitrary code and to define new Excel functions. Features:

- Based on the standard Python interpreter (i.e., not IronPython or other alternatives). Therefore it is fully compatible with all standard Python extensions
- Easy installation – just unpack and add the DLL to Excel as an Add-In. No registry modification, no installation to system directories
- Define new Excel function at run-time directly from the Excel worksheets
- No COM - based on the pure C-language Excel API

The ExPy add-in is made available to you free-of-charge on this web-site, under the licensing terms detailed below. If you need to integrate Excel and Python we can help! For all enquiries please contact us at webs@bnikolic.co.uk.

The source code: https://github.com/bnikolic/ExPy

4.4 pywin32

pywin32 homepage

Licence:

PyWin32 uses the Common Object Model (COM) to communicate with MS Windows applications. COM is the Windows infrastructure for intra-application communication. It's been around for a very, very long time (even in "people" years). And it's a two-way conduit.

COM allows a program to "tap" Windows on the shoulder and say, "Hey, if you know a thing called 'Excel', can you start that application? Cheers. Now that it's started, can you please run the method that does X, Y, Z?".

Due to this, it's the reason most of the other options in this category leverage it do do their magic.

It's entirely possible to use this library to integrate Python and Excel without using one of the wrappers but you'll have to do a lot of things manually that the other solutions provide. Stackoverflow has a lot of help on this.

http://timgolden.me.uk/pywin32-docs/contents.html

4.5 PyXll

PyXll homepage

Licence:

PyXLL is an Excel Add-In that enables developers to extend Excel's capabilities with Python code.

PyXLL makes Python a productive, flexible back-end for Excel worksheets, and lets you use the familiar Excel user interface to interact with other parts of your information infrastructure.

With PyXLL, your Python code runs in Excel using any common Python distribution(e.g. Anaconda, Enthought's Canopy or any other CPython distribution from 2.3 to 3.8).

Because PyXLL runs your own full Python distribution you have access to all third party Python packages such as NumPy, Pandas and SciPy and can call them from Excel.

It has a great strength in being able to manage custom buttons on the Ribbon menu.

Example use cases include:

- Calling existing Python code to perform calculations in Excel
- Data processing and analysis that's too slow or cumbersome to do in VBA
- Pulling in data from external systems such as databases
- Querying large datasets to present summary level data in Excel
- Exposing internal or third party libraries to Excel users

Features:

- Worksheet Functions
  - Argument and Return Types

    - Array Functions
    - Asynchronous Functions
    - Handling Errors
    - Function Documentation
    - Variable Arguments
    - Interrupting Functions
- Using Pandas in Excel
- Menu Functions
- Customizing the Ribbon
- Context Menu Functions
- Macro Functions
- Real Time Data
- Reloading and Rebinding
- Error Handling
- Python as a VBA Replacement
- Distributing Python Code

4.6 xlwings Community Edition (CE)

xlwings CE homepage

Licence:

xlwings CE is a BSD-licensed Python library that makes it easy to call Python from Excel and vice versa:

- Scripting: Automate/interact with Excel from Python using a syntax close to VBA.
- Macros: Replace VBA macros with clean and powerful Python code.
- UDFs: Write User Defined Functions (UDFs) in Python (Windows only).
- REST API: Expose your Excel workbooks via REST API.

xlwings is a sophisticated COM wrapper.

Numpy arrays and Pandas Series/DataFrames are fully supported. xlwings-powered workbooks are easy to distribute and work on Windows and Mac.

- Top-Level functions
    - view
- Object Model
    - Apps
    - App
    - Books
    - Book
    - Sheets

  - Sheet
  - Range
  - Range Rows
  - Range Columns
  - Shapes
  - Shape
  - Charts
  - Chart
  - Pictures
  - Picture
  - Names
  - Name
- Extensions
  - In-Excel SQL
- MatPlotLib

4.7 xlwings Pro

xlwings Pro homepage

Licence:

xlwings PRO adds the following features to xlwings CE:

- Dedicated support via email, phone, screenshare
- Access to video training
- Additional features like embedded code (see below)
- Build zero-configuration installers for easy deployment (see below)
- Access to the reports add-on

Help on the features;

- Embedded Code: Store your Python source code directly in Excel for easy deployment.
- One-Click Zero-Config Installer: Guarantees that the end user does not need to know anything about Python.
- xlwings Reports: A template based reporting mechanism, allows business users to change the layout of the report whithout having to change Python code.
- Plotly static charts: Support for Plotly static charts.

CHAPTER 5

Integrations with Calculation

These solutions integrate Excel and Python but have an added feature-set due to providing access to a library that can evaluate Excel formulas and functions in Python.

5.1 FlyingKoala

FlyingKoala is a purpose built Python library and Excel Add-In which adds functionality to the xlwings Excel/Python integration by adding calculation through xlcalculator.

This provides the ability to evaluate Excel formulas in Python (eg; in pandas/numpy/scipy) while you are in Excel and also evaluate Excel formulas in Python where Excel isn't installed.

Key Features;

- Adds the ability to define the behaviour of a UDF using an Excel formula
  - Makes a particular calculation transparent to managers, domain experts and potentially other companies (where you need to share techniques but can't be certain of coding skill).
  - UDF worked example
- Unit testing of Excel models either "in Excel" (while it's running) or without Excel (doesn't need to be installed/server)
  - Comprehensively exercising a model written in Excel helps provide evidence that the model operates as you claim
  - Use unit tests to help ensure a Python coded replica of an Excel workbook has correctly replicated the model
  - Unit test example
- Is a great repository for generic UDFs that use specialist Python libraries
  - PVLib for Photovoltaic analysis
  - Pandas for timeseries transformations

- numpy for differential equations
- numpy.finance for financial modelling
- Harvest and Xero for timesheeting, invoicing and accounting integrations

5.2 PyCel with PyXll

It appears to be possible to get Excel/Python integration and calculation working with PyXll using PyCel. There are some notes on the PyCel GitHub page to say this works and provides an explaination as to how.

From the presentation it seems as though is something that contributors to Pycel have created due to having used PyXll. So the level of integration is such that PyXll doesn't need to know about the existence of PyCel.

There's no discernable list of features or benefits except what you can figure for yourself.

From the PyCel GitHub page; It's possible to run pycel as an excel addin using PyXLL. Simply place pyxll.xll and pyxll.py in the lib directory and add the xll file to the Excel Addins list as explained in the pyxll documentation.

CHAPTER 6

Microsoft Add-ins

These solutions are found in the Microsoft Marketplace, the in-app store for Microsoft products. I have found these particular Add-ins by going through the MS Marketplace in MS Excel.

6.1 funfun

https://appsource.microsoft.com/en-us/product/office/WA104381207?tab=Overview

6.2 XLMiner Analysis ToolPak

https://appsource.microsoft.com/en-us/product/office/WA104379190?tab=Overview

6.3 Blockspring

https://appsource.microsoft.com/en-us/product/office/WA104379518?tab=Overview

6.4 price f(x)

https://appsource.microsoft.com/en-us/product/office/WA104381177?src=office&tab=Overview

6.5 Scott's Xero

https://apps.xero.com/au/app/scotts-add-ins

CHAPTER 7

Blog posts

These are cool blog posts I've come across in the journey to both put this software review together and also in my jorney to develop xlcalculator and FlyingKoala.

7.1 Joel Spolsky

An ex-Excel team leader discussing some how's and why's of working in the Excel team.

https://www.joelonsoftware.com/2001/10/14/in-defense-of-not-invented-here-syndrome/

This blog post explains why Excel calculates numbers that are different to the ones you get through other tools.

https://www.joelonsoftware.com/2007/09/26/explaining-the-excel-bug/

7.2 Dirk Gorissen

The genesis of PyCel.

https://dirkgorissen.com/2011/10/19/pycel-compiling-excel-spreadsheets-to-python-and-making-pretty-pictures/

7.3 Vincenzo Arcidiacono

Formulas creator Vincenzo is involved in this project CO2MPASS which is a vehicle simulator predicting NEDC CO2 emissions from WLTP. Formulas is used in CO2MPAS. It's a good illustration on how these tools can potantially be used.

https://co2mpas.readthedocs.io/en/stable/

7.4 Steve Holden

Steve Holden is behind PyXll and runs a blog highlighting the potential of tools in this space using PyXll in the worked examples.

https://www.pyxll.com/blog/

CHAPTER 8

Non-Excel tools

8.1 ModelX

https://docs.modelx.io/en/latest/

modelx is a Python package to build object-oriented models consisting of formulas and values to carry out complex calculations. You can think of it as a hierarchical and multidimensional extension of spreadsheet, but there's so much more to it!

modelx enables you to interactively develop, run and scrutinize complex models in smart ways:

- Only basic Python knowledge required
- Formulas defined by Python functions
- Object-oriented, supports composition and inheritance
- Parameterization
- Dependency tracing
- Reading from Excel and CSV files
- GUI as Spyder plugin (spyder-modelx)
- Saving to text files, enabling use of version control systems
- Document integration enabling use of document generators
- Pandas interface

CHAPTER 9

Indices and tables

- genindex
- search