AL7_Transpose2D
PRESENTATION

I. Introduction

The User Defined Function (UDF) **AL7_Transpose2D** offers an advanced and powerful solution for transposing matrices or ranges of cells into Excel, surpassing Application.Transpose in terms of robustness and flexibility.
This function is available in two versions:

- **Transpose2D (alias Trans2D)** :
  Primarily intended for use in VBA.
- **TransRange (alias TransRng)** :
  Some options are not available in order to be used more easily in the cells. If necessary, use the Transpose2D (*Trans2D*) function.

Of course, both functions can be used in cells or in VBA, the difference is that Transpose2D offers more functions than TransRange.
While keeping the advantages of Transpose2D, you can ask to keep the same characteristics/numbering behaviour of transpositions as the Transpose functions of Excel for this put the option LikeExcel = True (-1).

Additional Features:

PARTIAL TRANSPOSITION of the table indicating the beginning and end of each dimension.
DEFINE INDEPENDENTLY THE BASIS OF EACH TABLE DIMENSIONS.
... See comparative table here under.

A. Comparative

1. Transpose2D, Application.Transpose, et WorksheetFunction.Transpose

	Transpose2D	Application.Transpose	WorksheetFunction.Transpose
Maximum size managed	Unlimited (optimized). Limit: Material Capacity	Depends on the version of Excel	**65,536 items max** (error beyond)
Always returns an array. Table case T(1,1) or T(1 to 1)	**Oui**. Sauf si l'option LikeExcel = False	**No**	**No**
Converts a simple string or value to an array	**Yes**, 99 → T(1,1,) = 99 "Foo" → T(1,1) = "Foo"	**No**	**No**
If 2nd dimension = 1, conversion to a 1-dimensional array. Ex.: T(5,1)→ T(5)	**On demand** (Option: LikeExcel = True).	**Yes, always**	**Yes, always**
Basics of the 2 independent and configurable dimensions	**YES** Use Transpose2D	**NO,** always base = 1 → LBound of T,1 and T,2 = 1	**NO,** always base = 1 → LBound of T,1 and T,2 = 1
Error handling	**Yes**, with clear messages	No, common mistakes	No, common mistakes
Automatic type conversion	**Yes**	**Yes**, but sometimes unstable	**No** (errors with long numbers)
Extraction and partial transposition	**Yes,** on each of the 2 dimensions.	**No**	**No**
Use with named ranges	**Yes**	**Yes**	**Yes**
Use on multiple sheets of open or closed workbooks	**Yes**	**No**	**No**
Dynamic table compatibility	**Yes**	**Oui** (Excel 365/2019)	**No**
Performance on large matrices	**Optimized** (fast processing)	**Good** but can slow down	**Issue** beyond 65,536 items
Use in VBA	**Yes,** full parameter management	**Yes,** but less flexible	**Yes,** but requires an explicit table
Working in Excel cells	**Yes**	**Yes**	**Yes** (but limited in features)
Keeping the basics of each dimension	**Yes**	**No** (Always returns tables in base 1)	**No** (Always returns tables in base 1)
Modify each of the bases of each dimension	**Yes.** Use Transpose2D)	**No** (Always returns tables in base 1)	**No**

B. Practical cases and recommendations

- ✅ **Processing large data sets** : Faster and without limitation.
- ✅ Partial transpose on one or 2 dimensions.
- ✅ **Use with numbers and text** : Unlike Application.Transpose, no conversion errors.
- ✅ **Dynamic table compatible** : Seamless integration with new Excel features.
- ✅ **Use with named ranges and closed workbooks**.

C. Two types of functions for specific uses

1. Transpose2D (alias Trans2D) – VBA-oriented

- ✓ **Main use** :
  Function optimized for use in VBA because **it has more possibilities than TransRange**.
- ✓ **Why use it?**
  Better handles dynamic arrays and avoids common Application errors. Transpose.
- ✓ **Parameters:**

```
TRANSPOSE2D(ToTranspose, _
              [LikeExcel], _
              [StartDim1], _
              [EndDim1], _
              [StartDim2], _
              [EndDim2], _
              {KeepNumDim1], _
              [KeepNumDim2], _
              [BaseOut1], _
              [BaseOut2])
```

Examples in VBA:

```
Dim Result As Variant
' Transposition simple
Result = Trans2D(MyArray)
' Or transposition of a part of a painting: from 1 to 5 and from 2 to 4
Result = Transpose2D(MyArray; LikeExcel:=False; StartDim1:=1; EndDim1:=5; StartDim2:=2; EndDim2 :=4)
```

2. TransRange (alias TransRng) – Excel Cell Oriented

- ✓ **Primary Use:**
  Function intended for use directly in Excel sheets.
- ✓ **Why use it?**
  Simpler because of fewer parameters, it is often enough to transpose all or part of a range.
  Allows for seamless integration into Excel formulas without the need for VBA.
- ✓ **Paramètres :**

```
TransRange(Transpose_Range, _
             [StartDim1], _
             [EndDim1], _
             [StartDim2], _
             [EndDim2])
```

Example in Excel cell:

```
' Transposition simple
= TransRange(Cell Range)
' Partial transposition of the range A1:D3 from 1 to 3 and from 1 to 4
 =TransRng(A1:D3, 1, 3, 1, 4)
```

Quick User Guide

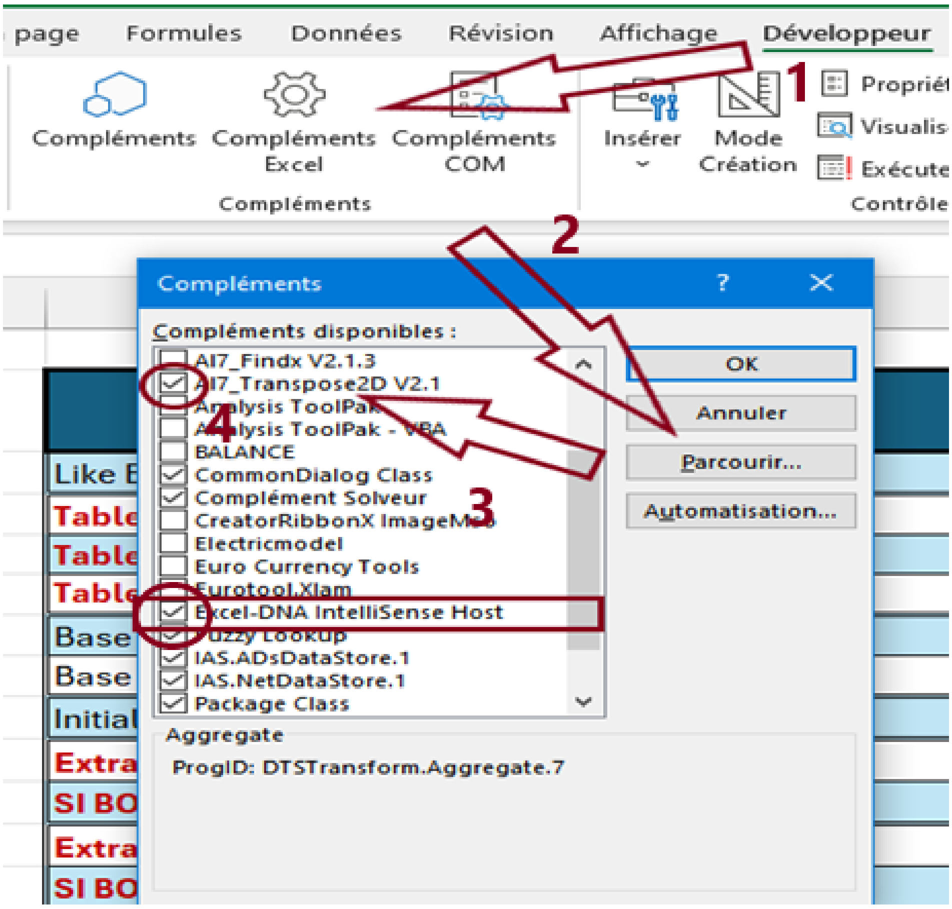
In this guide, we introduce you to the different settings you can use.
If necessary, a brief explanation of their uses will be given to you.
The parameters between [...] are optional.
With identical parameters, the Transpose2D (aka Trans2D) and TransRange (aka TransRng) functions are interchangeable. So, you will get the same results.
Although the Transpose2D examples indicate tables, you can provide an Excel address range, e.g., *Range("A1:F100")*.

I. Installation and configuration

1. Hard disk installation

a. Download *AL7_Transpose2D.zip*
b. Unzip/Extract the file to have all the files.
c. Place the AL7_Transpose2D.xlam file in an accessible folder
In general: C:\Users\YourName\AppData\Roaming\Microsoft\AddIns
d. Place the *ExcelDna.IntelliSense64.xll* file in the same directory.
e. Open Excel and go to **File > Options > Add-ins**.
f. In the **Manage** section, select **Excel Add-ins** and click **Go to**.
g. Click **Browse...** and select *AL7_Transpose2D.xlam*.
h. Repeat the same operations for the *ExcelDna.IntelliSense64.xll file*
i. Validate and make sure that the 2 add-ins are activated.

Don't forget to reference *ExcelDna.IntelliSense64* for help in entering function parameters.



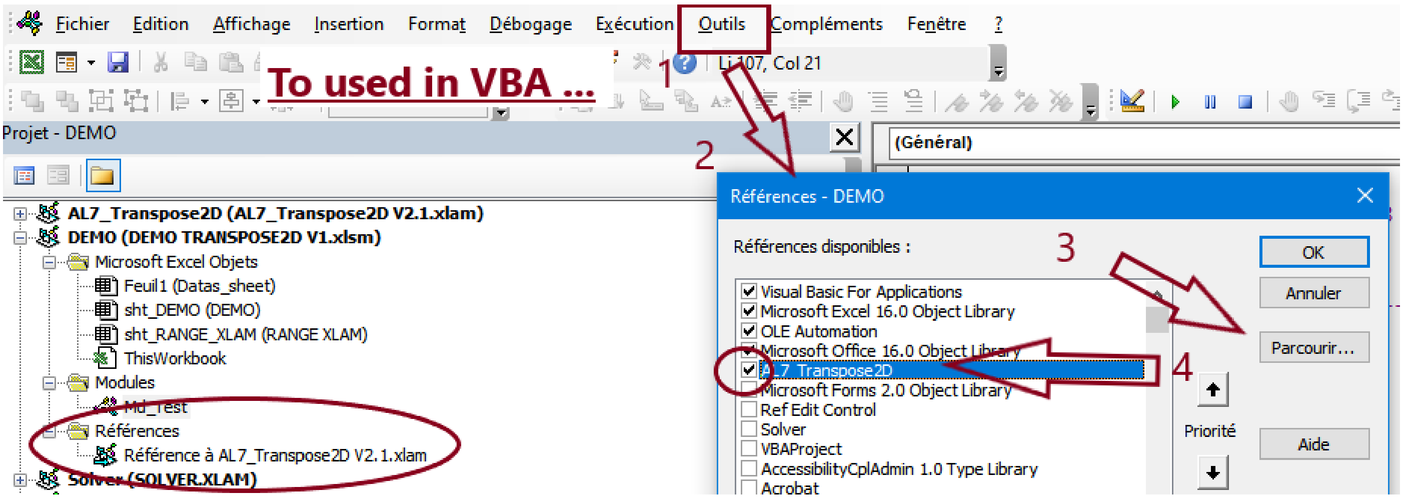
Compléments disponibles :

☐ Al7_Findx V2.1.3
☑ Al7_Transpose2D V2.1
☐ Analysis ToolPak
☐ Analysis ToolPak - VBA
☐ BALANCE
☑ CommonDialog Class
☑ Complément Solveur
☐ CreatorRibbonX ImageMso
☐ Electricmodel
☐ Euro Currency Tools
☐ Eurotool.Xlam
☑ Excel-DNA IntelliSense Host
☑ Fuzzy Lookup
☑ IAS.ADsDataStore.1
☑ IAS.NetDataStore.1
☑ Package Class

Aggregate
ProgID: DTSTransform.Aggregate.7

2. SEO in VBA

1. Open **VBA (ALT +F11)**.
2. Go to **Tools > References**.
3. Check AL7_Transpose2D to allow its use in your VBA program.



- AL7_Transpose2D (AL7_Transpose2D V2.1.xlam)
- DEMO (DEMO TRANSPOSE2D V1.xlsm)
  - Microsoft Excel Objets
    - Feuil1 (Datas_sheet)
    - sht_DEMO (DEMO)
    - sht_RANGE_XLAM (RANGE XLAM)
    - ThisWorkbook
  - Modules
    - Md_Test
  - Références
    - Référence à AL7_Transpose2D V2.1.xlam
- Solver (SOLVER.XLAM)

Références disponibles :

☑ Visual Basic For Applications
☑ Microsoft Excel 16.0 Object Library
☑ OLE Automation
☑ Microsoft Office 16.0 Object Library
☑ AL7_Transpose2D
□ Microsoft Forms 2.0 Object Library
□ Ref Edit Control
□ Solver
□ VBAProject
□ AccessibilityCplAdmin 1.0 Type Library
□ Acrobat

II. Using Transpose2D and TransRange

1. Transpose2D: Syntaxe VBA

Result_arr = Transpose2D(*Datas_array*, _
[StartDim1], _
[EndDim1], _
[StartDim2], _
[EndDim2], _
[BaseOut1], _
[BaseOut2])

The parameters between **[...] are optional**.

- *Datas_array*: An **array <u>or range</u> of cells**
- [***StartDim1***]: In the 1st dimension of the array, the start number to be transposed
- [***EndDim1***]: In the 1st dimension of the array, the end number to be transposed
- [***StartDim2***]: In the 2nd dimension of the array, the start number to be transposed
- [***EndDim2***]: In the 2nd dimension of the array, the end number to be transposed
- [***BaseOut1***]: a number used as the basis for the 1st dimension of the array.
- [***BaseOut2***]: a number used as a basis for the 2nd dimension of the array.

2. Examples:

Example in VBA:

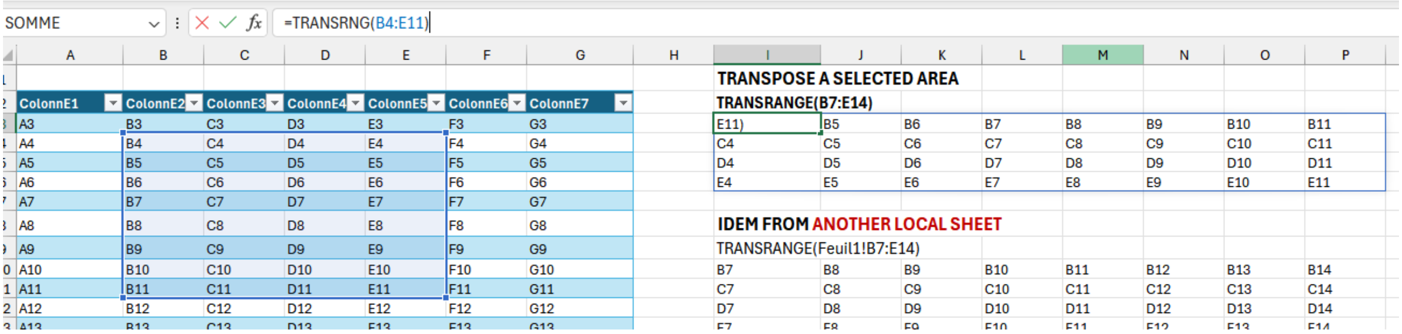
```
Dim Result_arr As Variant
'1) Transposition simple
Result_arr = Trans2D(MyArray)
'2) Transposition of a part of a table: from 1 to 5 and from 2 to 4
Result_arr = Transpose2D(MyArray; LikeExcel:=False; StartDim1:=1; EndDim1:=5; StartDim2:=2; EndDim2 :=4)
'3) Transposition by conserving and changing the basis of one or both dimensions
' Here we ask that the transposed table begin at 10 and 100.
Result_arr = Transpose2D(MyArray; BaseOut1:=10; BaseOut2:=100)
```

3. Syntax in a cell:

= TransRange(Plage_de_données, [StartDim1], [EndDim1], [StartDim2], [EndDim2])

Example with TransRange in a cell:

```
'Simple transposition to obtain transposition
 = TransRange(A1:D3)
```



=TRANSRNG(B4:E11)

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
1									**TRANSPOSE A SELECTED AREA**							
2	ColonnE1	ColonnE2	ColonnE3	ColonnE4	ColonnE5	ColonnE6	ColonnE7		**TRANSRANGE(B7:E14)**							
3	A3	B3	C3	D3	E3	F3	G3		E11)	B5	B6	B7	B8	B9	B10	B11
4	A4	B4	C4	D4	E4	F4	G4		C4	C5	C6	C7	C8	C9	C10	C11
5	A5	B5	C5	D5	E5	F5	G5		D4	D5	D6	D7	D8	D9	D10	D11
6	A6	B6	C6	D6	E6	F6	G6		E4	E5	E6	E7	E8	E9	E10	E11
7	A7	B7	C7	D7	E7	F7	G7									
8	A8	B8	C8	D8	E8	F8	G8		**IDEM FROM ANOTHER LOCAL SHEET**							
9	A9	B9	C9	D9	E9	F9	G9		TRANSRANGE(Feuil1!B7:E14)							
10	A10	B10	C10	D10	E10	F10	G10		B7	B8	B9	B10	B11	B12	B13	B14
11	A11	B11	C11	D11	E11	F11	G11		C7	C8	C9	C10	C11	C12	C13	C14
12	A12	B12	C12	D12	E12	F12	G12		D7	D8	D9	D10	D11	D12	D13	D14
13	A13	B13	C13	D13	E13	F13	G13		E7	E8	E9	E10	E11	E12	E13	E14

III. Conclusion

With **AL7_Transpose2D**, gain in efficiency and reliability in the handling of your Excel matrices.
Designed to outperform Excel's Transpose, this tool is a must-have for any user working with complex data.
Download it, activate it and enjoy optimized transposition!