



Friedrich-Alexander-Universität
Philosophische Fakultät und
Fachbereich Theologie



UNIVERSITY OF
BIRMINGHAM



Deutsche
Forschungsgemeinschaft



Arts and
Humanities
Research Council

living in some squalor; the other exhibitions were **IN** houses, old shops, small factory spaces, offices and
might be described as 'wet' or 'dry' (W or D) and **IN** a humid temperate climate such as Britain experie
e powerful message put across by the curriculum **IN** most schools is that knowledge can be divided into
ot like the trouble we had half an hour ago, when **IN** fact there were three lines, sections of which there
opportunities for oral **READING** arise naturally **IN** the course of the day; for instance, in the activity of
traditional hand-made **CONCORDANCES** **IN** several ways. One is that they will be much more
ched Milan. 'How old were you when you arrived' **IN** Italy?' The American journalist broke into his thoug
of Modern Art, keen in discerning what was good **IN THE** arts of many ages and styles. As for working
who felt it necessary to take aside both captains **IN** the **21ST** minute and warn them about each team
mation has undergone several media revolutions **IN** the last **CENTURY**. In principle, now that opera
to some extent the ownership of the biggest hits **IN** each year), the aggregate market share of the five
focus from arts education in general, to one area **IN** particular, namely English lessons and the fictiona
f the horse was, shall I say dreamed up, erm and **IN** fact we had one or two horses, Clydesdale horses
faded woods. It was merely that he never wanted **IN** a tournament round to risk anything which might u

2

Reading concordances with algorithms

Konvens 2025 Tutorial | 9 Sep 2025

Reading Concordances in the 21st Century (RC21) project team

Nathan Dykes • **Stephanie Evert** • Michaela Mahlberg • Alexander Piperski

Overview of the tutorial

Part I

1. Concordance reading
2. Strategies for organising concordances
3. Reading concordances with algorithms
4. A mathematical framework

Part II

5. Hands-on: FlexiConc in Jupyter
6. Implementation details
7. Loading your own data

Overview of the tutorial

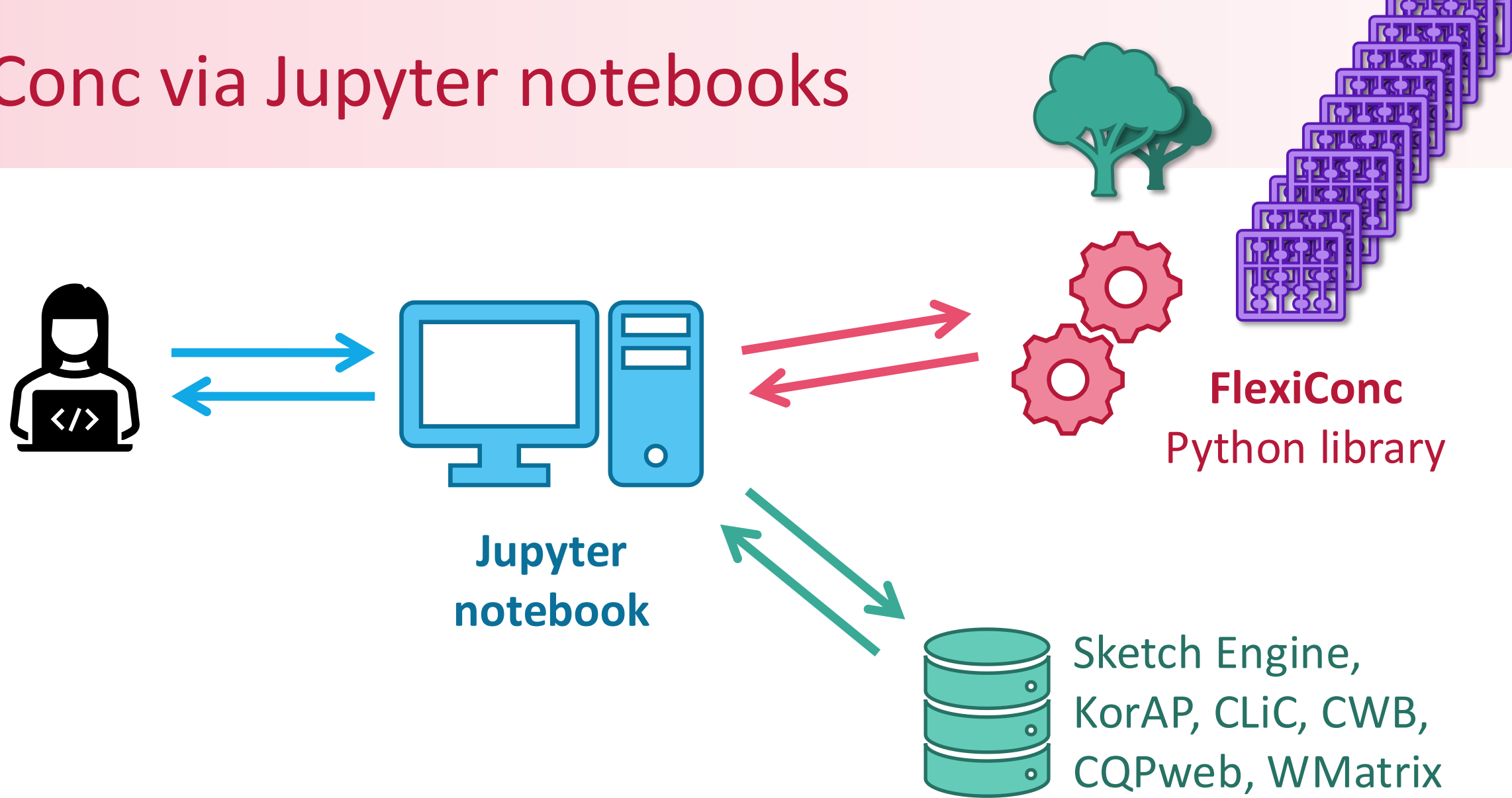
Part I

1. Concordance reading
2. Strategies for organising concordances
3. Reading concordances with algorithms
4. A mathematical framework

Part II

- 5. Hands-on: FlexiConc in Jupyter**
6. Implementation details
7. Loading your own data

FlexiConc via Jupyter notebooks



<https://pypi.org/project/FlexiConc/>

FlexiConc via Jupyter notebooks

Why FlexiConc? → benefit from algorithmic innovation!

- Use FlexiConc library from Python code in Jupyter notebook
- Combines (some) interactivity with custom programming, e.g. to carry out additional analyses not available in concordancer
- Connect with various indexing & search backend to access your own corpora there → FlexiConc has ready-made functions
- The Right Way: install Jupyter Lab + FlexiConc on your computer
- The easy way: run notebook in Google Colab ... in a moment

Installing FlexiConc (Windows / MacOS)

- install Anaconda Python or miniconda (anaconda.com/download)
- create a separate environment for FlexiConc
(`conda create -n FlexiConc python=3.13`)
- don't forget to activate it (`conda activate FlexiConc`)
- install PyICU from Anaconda (`conda install pyicu`)
- install JupyterLab (`conda install jupyterlab`)
- install FlexiConc and dependencies from PyPI with pip:
`pip install -U
"flexiconc[notebooks,web,ICU,partitioning,annotation]"`
- now start JupyterLab (`jupyter lab .`) ... or ask us for help

Installing FlexiConc (Linux)

- create virtualenv: `python3 -mvenv --upgrade-deps venv`
- activate virtualenv: `source venv/bin/activate`
- install ICU (e.g. `sudo apt install libicu-dev` on Ubuntu)
- install JupyterLab (`pip install jupyterlab`)
- install FlexiConc and dependencies from PyPI with pip:
`pip install -U
"flexiconc[notebooks,web,ICU,partitioning,annotation]"`
- start JupyterLab in your working directory: `jupyter lab .`
- ask us for help if it doesn't work ...

FlexiConc in Jupyter notebooks

- First steps

[flexiconc_introduction_Konvens 2025.ipynb](#)

Download from Google Colab or copy and use it there

https://colab.research.google.com/drive/1sAC7c0vljy6og_dSwHbvbCs3QTs4a6Lx?usp=sharing



Overview of the tutorial

Part I

1. Concordance reading
2. Strategies for organising concordances
3. Reading concordances with algorithms
4. A mathematical framework

Part II

5. Hands-on: FlexiConc in Jupyter
- 6. Implementation details**
7. Loading your own data

Representation of concordances in FlexiConc

- In FlexiConc, a Concordance object must include two Pandas dataframes
 - metadata
 - information about concordance lines
 - tokens
 - information about individual tokens

Representation of concordances in FlexiConc: metadata

	line_id	doc.printed	doc.title	p.speaker
0	0	1831	Antonius und Cleopatra	ENOBARBUS
1	1	1831	Coriolanus	ERSTER BÜRGER
2	2	1832	Cymbeline	FRANZOSE
3	3	1832	Cymbeline	IMOGEN
4	4	1832	Das Wintermärchen	LEONTES
5	5	1799	Der Kaufmann von Venedig	BASSANIO
6	6	1831	Der Widerspenstigen Zähmung	SCHLAU
7	7	1890	Die beiden edlen Vettern	EMILIA
8	8	1890	Die beiden edlen Vettern	KERKERMEISTER
9	9	1832	Die lustigen Weiber von Windsor	FRAU PAGE
10	10	1832	Die lustigen Weiber von Windsor	FENTON

Representation of concordances in FlexiConc: tokens

	line_id	id_in_line	offset	word	tag	lemma	pos
125	2	10	-10	ich	PRO.Pers.Subst.1.Nom.S...	sie	p
126	2	11	-9	Euch	PRO.Pers.Subst.2.Dat.Pl.*	sie	p
127	2	12	-8	und	CONJ.Coord	und	c
128	2	13	-7	meinen	PRO.Poss.Attr.Acc.Sg.M...	meine	p
129	2	14	-6	Landsmann	N.Reg.Acc.Sg.Masc	Landsmann	n
130	2	15	-5	versöhnen	VINF.Full	versöhnen	v
131	2	16	-4	konnte	VFIN.Mod.3.Sg.Past.Ind	können	v
132	2	17	-3	;	SYM.Pun.Sent	;	x
133	2	18	-2	es	PRO.Pers.Subst.3.Nom.S...	sie	p
134	2	19	-1	wäre	VFIN.Aux.3.Sg.Past.Subj	sein	v
135	2	20	0	schade	ADJD.Pos	schade	j
136	2	21	1	gewesen	VPP.Aux.Psp	sein	v
137	2	22	2	,	SYM.Pun.Comma	,	x
138	2	23	3	wäret	VFIN.Aux.2.Pl.Past.Subj	sein	v
139	2	24	4	Ihr	PRO.Pers.Subst.2.Nom.P...	sie	p
140	2	25	5	mit	APPR	mit	i
141	2	26	6	so	ADV	so	a

Representation of algorithms in FlexiConc

- Algorithms are stored as individual .py files in the `algorithms` folder provided with the package
- Algorithms can also be loaded from user-specified folders
- Each algorithms includes an exact specification as a JSON schema, which can be updated dynamically for each node
 - name
 - description
 - algorithm type
 - arguments
 - availability of an algorithm for a node

Representation of algorithms in FlexiConc: arguments

- Schema for individual arguments
 - name
 - description
 - type
 - restrictions
 - list of possible values
 - min/max
 - ...

Internal structure of an algorithm

Docstring

```
1  def partition_ngrams(conc, **args):
2      """
3          Partition concordance lines by the n-gram tuples of tokens at the specified offsets. Compare Anthony's
          (2018) KWIC Patterns and subsequent work.
4
5          Parameters
6          -----
7          positions : list[int]
8              Offsets (relative token positions) that form the pattern.
9          tokens_attribute : str, optional
10             Token-attribute column used to fetch the tokens
11             (default ``"word"``).
12          case_sensitive : bool, optional
13             Preserve original casing if *True*; otherwise tokens are converted
14             to lowercase (default ``False``).
15
16          Returns
17          -----
18          dict
19             ``{"partitions": [ {"id": int,
20                             "label": str,
21                             "line_ids": list[int]}, ... ]}``
22
23          Notes
24          -----
25          * The ``label`` of each partition is the stringified tuple of tokens
26            extracted at *positions*.
27          * Partitions are ordered by descending size and, for equal sizes,
28            alphabetically by their label.
29      """
```


Metadata

```
32     # Metadata for the algorithm
33     partition_ngrams._algorithm_metadata = {
34         "name": "Partition by Ngrams",
35         "description": "Extracts ngram patterns from specified positions and partitions the
concordance according to their frequency in the concordance lines. Compare Anthony's (2018) KWIC
Patterns and subsequent work.",
36         "algorithm_type": "partitioning",
```

Argument
schema

Dynamic list of
possible values

```
37     "args_schema": {
38         "type": "object",
39         "properties": {
40             "positions": {
41                 "type": "array",
42                 "items": {"type": "integer"},
43                 "description": "The list of positions (offsets) to extract for the ngram
pattern."
44             },
45             "tokens_attribute": {
46                 "type": "string",
47                 "description": "The positional attribute to search within (e.g., 'word').",
48                 "default": "word",
49                 "x-eval": "dict(enum=list(set(conc.tokens.columns) - {'id_in_line', 'line_id',
'offset'})))"
50             },
51             "case_sensitive": {
52                 "type": "boolean",
53                 "description": "If True, the search is case-sensitive.",
54                 "default": False
55             }
56         },
57         "required": ["positions"]
58     }
59 }
```

```
66     # Step 1: Filter tokens based on the specified positions
67     filtered_tokens = conc.tokens[conc.tokens["offset"].isin(positions)].copy()
68
69     # Step 2: Apply case sensitivity
70     if not case_sensitive:
71         filtered_tokens[tokens_attribute] = filtered_tokens[tokens_attribute].str.lower()
72
73     # Step 3: Aggregate ngrams by line_id as tuples of tokens at the specified positions
74     ngram_dict = filtered_tokens.groupby("line_id")[tokens_attribute].apply(
75         lambda x: tuple(x.tolist())
76     ).to_dict()
```

```
78     # Step 4: Group line IDs by unique ngram patterns
79     group_dict = {}
80     for line_id, ngram in ngram_dict.items():
81         group_dict.setdefault(ngram, []).append(line_id)
```

```
82
83     # Step 5: Sort ngram patterns: first by descending size, then alphabetically by their string
representation.
84     sorted_group_dict = {
85         str(ngram): line_ids
86         for ngram, line_ids in sorted(
87             group_dict.items(),
88             key=lambda item: (-len(item[1]), str(item[0])))
89     }
90 }
```

```
92     # Step 6: Format the output as a list of dictionaries with an "id" for each group.
93     result = {"partitions": [
94         {
95             "id": idx,
96             "label": label,
97             "line_ids": line_ids
98         }
99         for idx, (label, line_ids) in enumerate(sorted_group_dict.items())
100     ]}
101
102     return result
```

Creating custom algorithms should be doable!

Executing algorithms

- Algorithms are applied to subsets of concordance lines and produce information about:
 - selected lines;
 - ordering of lines;
 - grouping of lines;
 - token spans to be highlighted in the concordance view.
- Algorithms can only be accessed via `add_subset_node` and `add_arrangement_node`, ensuring that:
 - only permissible algorithm combinations are used;
 - any algorithm call is documented in the analysis tree;
 - the results of algorithm execution are stored correctly.

Overview of the tutorial

Part I

1. Concordance reading
2. Strategies for organising concordances
3. Reading concordances with algorithms
4. A mathematical framework

Part II

5. Hands-on: FlexiConc in Jupyter
6. Implementation details
- 7. Loading your own data**

FlexiConc in Jupyter notebooks

- Loading your own data:

`flexiconc_import_Konvens 2025.ipynb`

Download from Google Colab or copy and use it there

<https://colab.research.google.com/drive/1HuVRl748lWe65Mzl5HNge4Wwgq5xiqw2?usp=sharing>



- After you've learned how to load your own data, explore a concordance of your choice. Then we'll discuss your experiences and results.

Import from CLiC

<https://clic-fiction.com/>

- Preparation: none

```
C = Concordance()  
C.retrieve_from_clic(query=["head"], corpora="dickens")
```

Import from Sketch Engine

<https://app.sketchengine.eu/>

- Prerequisite: paid SkE account (free 30-day trials available)

Import from Sketch Engine

https://app.sketchengine.eu/#ca?corpname=user%2FSEvert%2Ftta

MANAGE CORPUS

CORPUS: TTA (English)

Trump Twitter Archive

Browse
View documents and folders, edit metadata

Delete
Remove corpus permanently

Stephanie Evert (ID: 80014)

SUBSCRIPTION & INVOICING CHANGE PASSWORD

Username SEvert
E-mail stephanie.evert@fau.de
Account type Multi-user account
Group account end date April 8, 2026
Corpus storage used (words) 4,098,805 of 100,000,000 (4%)
Academic user yes
Sketch Engine API key 78b4f7fd6c3e75e62469f535d88fcd69
Account admins Stephanie Evert

My account
My Sketch Engine
Settings
Local administration
Logout

Generate new key

9 Sep 2025 | © RC21 Team

Import from Sketch Engine

https://app.sketchengine.eu/#ca?corpname=user%2FSEvert%2Ftta

Chii News Fun Science Software Projects Actions Drop Box HoloMem CL Journal TACL DFG BMBF TODO Smart

TTA user/SEvert/tta created May 7, 2025 at 8:52:32 PM

Trump Twitter Archive

MANAGE CORPUSMANAGE SUBCORPORACOMPARE CORPORATEXT TYPE ANALYSIS

GENERAL INFO

Language: English

CORPUS DESCRIPTION & BIBLIOGRAPHY

TAGSET

WORD SKETCH GRAMMAR

TERM GRAMMAR

COUNTS

Tokens	1,359,510
Words	1,036,092
Sentences	377
Documents	56,570

TEXT TYPES

TEXT TYPE ANALYSIS

<doc> (2)	13
File ID , doc.id	13
File name , doc.filename	13
<text> (0)	56,570
<g> (0)	27,609
<s> (0)	377

LEXICON SIZES

word?	75,630
tag	63
lempos?	49,323
pos	9
lemma	46,084
lempos_lc	46,180
lemma_lc	42,597
lc	64,983

COMMON TAGS

adjective	J.*
adverb	RB.?
conjunction	CC
determiner	DT
noun	N.*
numeral	CD
particle	RP
preposition	IN
pronoun	PP.?
verb	V.*

All tags

9 Sep 2025 | © RC21 Team

29

Import from Sketch Engine

<https://app.sketchengine.eu/>

- Prerequisite: paid SkE account (free 30-day trials available)
- Generate API access token (as shown on previous slides)
- Note down full path to desired corpus (as shown on previous slides)

```
C = Concordance()  
C.retrieve_from_sketchengine(  
    query='[lc="fake"] [lc="news"]',  
    corpus="user/SEvert/tta", # insert your path here  
    api_key="[YOUR API KEY]")
```

Import from CQPweb

<https://corpora.linguistik.uni-erlangen.de/cqpweb/>

Your query "water and sanitation" returned 669 matches in 424 different texts (in 409,134,520 words [1,956,223 texts]; frequency: 1.64 instances per million words)

[0.107 seconds - retrieved from cache]

Detailed output options

Formatting options

Choose operating system on which you will be working with the file:

UNIX (incl. Mac OS X & iOS/Android) ▾

Print short handles or full values for text categories:

full values ▾

Mark query results as <<< result >>>:

No ▾

Size of context:

20 words each way ▾

Download both tagged and untagged version of your results:

Yes ▾

Write information about table columns at the beginning of file:

Yes - column headings ▾

Format of output - KWIC or line:

KWIC ▾

Include sub-text region boundary markers:

Yes ▾

Include corpus positions (required for re-import)

Yes ▾

Include URL to context display

Yes ▾

Enter name for the downloaded file:

CQPweb_WaterSanitation_ParlUK

Please tick the text metadata categories that you want to include in your download:

Method:

Download text metadata ticked below ▾

Select from available text metadata:

☐ Agenda

☐ Chair?

☒ Date

☐ Month/Year

☒ Party

☐ Party Facts ID

☐ Speaker

☐ Speech number

☐ Terms

☒ Year

- ✓ Choose action...
- New query
- Thin...
- Frequency breakdown
- Distribution
- Dispersion
- Sort
- Collocations...
- Download...**
- Categorise...
- Save current query result...

Import from CQPweb

<https://corpora.linguistik.uni-erlangen.de/cqpweb/>

- Demo account credentials:
username: **studentN**, $N \in \{1, 2, \dots, 15\}$ password: **erlangen**
- Obtain desired concordance in CQPweb and save it with download action (as shown on previous slide)
- Make sure to include all relevant metadata
- Save the downloaded file in the same directory as the Jupyter notebook

```
C = Concordance()  
C.load_from_cqpweb_export("CQPweb_WaterSanitation_Par1UK.txt")
```

Import from WMatrix

<https://ucrel-wmatrix7.lancaster.ac.uk/>

- **Tag Wizard:** Create your own corpus from text files in ZIP archive
- Annotated with POS, lemma, semantic concept in 9 languages
- Corpus (“folder”) can be downloaded in SQLite format

Wmatrix7: Wmatrix multilingual tag wizardWmatrix

You are logged in to Wmatrix7 as: demo1@esslii.2025 (1217)

[Tagging > Tag Wizard...]
[Folders > My folders | Details | Delete... | Archive... | Extract... | Library... | CrossTab... | Empty TRASH]
[Options > Switch to Simple Interface | Edit user options...]
[Help > Contents | Availability | Tagsets: POS & Semantic | USAS: Lexicon & MWEs & Context rules | Updates | Feedback]

[You are here > My folders]

Upload file → Part-of-speech tagging
Lemmatisation → Semantic tagging → MatrixDB indexing → Frequency lists
N-gram frequency lists
Collocation table

1. Choose language: English

2. Enter new folder name:

3. Click the button to select a file:
Choose file No file chosen

4. Upload now
Reset form

The Wmatrix tag wizard puts your corpus through the automatic language specific POS and Semantic analysis stages, indexes the results in the MatrixDB database, and produces frequency lists, n-gram frequency lists and a collocation table from your text file. For English text, this wizard runs the CLAWS tagger and USAS tagger. For other supported languages, the wizard runs the PyMUSAS tagger tagging pipeline. For English, please do not run large texts (e.g. a file with more than 1 million words, or a collection of files in a zip where any one file is larger than 1 million words) through the tag wizard. These are better run off-line and loaded into Wmatrix afterwards. Please get in touch with Paul to do this. For texts run through PyMUSAS, spaCy currently sets a default maximum length per file of 1 million characters.

File types:
Please view [Tutorial A](#) for details on how to convert your PDF, DOC(X) or RTF files to TXT format which is suitable for Wmatrix. Further [input format guidelines](#) are available for the English CLAWS/USAS taggers, including for example how to avoid problems with less-than and greater-than symbols in the input text.

One corpus per folder:
If you do not specify a folder, one will be created with a unique name. It is recommended that you use a **new folder for each corpus**. If your corpus consists of more than one file, then we recommend concatenating the files together first or using a zip file to load all the files together.

i

Wmatrix

©2000-25 UCREL, Lancaster University.
For technical queries please contact Paul Rayson : p.rayson@lancaster.ac.uk

9 Sep 2025 | © RC21 Team

FlexiConc: Reading concordances with algorithms

33

Import from WMatrix

<https://ucrel-wmatrix7.lancaster.ac.uk/>

- Use own corpus (“folder”) or copy **ESSLI_Water_Par1UK** from library
- Main function: keyword analysis (for word, lemma, POS, concept)
- Note interesting keywords → concordance analysis in FlexiConc

Wmatrix7: Folder ESSLI_Water_Par1UK

You are logged in to Wmatrix7 as: demo1@essli.2025 (1217)

[**Tagging** > Tag Wizard...]


[**Folders** > My folders | Details | Delete... | Archive... | Extract... | Library... | CrossTab... | Empty TRASH]

[**Options** > Switch to Simple Interface | Edit user options...]

[**Help** > Contents | Availability | Tagsets: POS & Semantic | USAS: Lexicon & MWEs & Context rules | Updates | Feedback]

[You are here > My folders > ESSLI_Water_Par1UK]

	Frequency list	Concordance	N-grams	Collocation	Keyness analysis
Word	Word only (Sorted by: Frequency ; Word)		2 3 4 5	Word	Key words compared to: <input type="text" value="British English 2021 (BE21)"/> <input type="button" value="Go"/>
Lemma	Lemma only (Sorted by: Frequency ; Lemma)				Key lemmas compared to: <input type="text" value="British English 2021 (BE21)"/> <input type="button" value="Go"/>
Part of speech	POS only (Sorted by: Frequency ; POS) Word and POS (Sorted by: Frequency ; Word ; POS)				Key POS compared to: <input type="text" value="British English 2021 (BE21)"/> <input type="button" value="Go"/>
Semantic	USAS Tag only (Sorted by: Frequency ; USAS tag) Word and USAS tag (Sorted by: Frequency ; Word ; USAS tag)				Key concepts compared to: <input type="text" value="British English 2021 (BE21)"/> <input type="button" value="Go"/>



Import from WMatrix

<https://ucrel-wmatrix7.lancaster.ac.uk/>

- Get a free WMatrix account (or use our demo account)
- Demo account credentials:
username: **demo1@essli.2025** password: **u73ripee4y**
- Download complete annotated WMatrix corpus
- Then use query to obtain concordance for desired keyword

```
labour2005 = wmatrix.load(  
    corpus_name="LabourManifesto2005",  
    username="[USER]", password="[PASSWORD]",  
    db_filename="labour2005.db")  
C = labour2005.concordance_from_query("community")
```

Import your own files

```
from flexiconc import TextImport
T = TextImport()
T.load_files(
    paths=["data"],
    shorten_paths=True,
    db_name="db.sqlite",
    use_spacy=True,
    lemma=True,
    pos=True,
    tag=True
)
C = T.concordance_from_query("stomach")
```

Feedback and contributions welcome!

Please fill in our feedback form for this tutorial!

<https://forms.gle/tRRVRbQHA4Qa2iFeA>

Home / Research / Current Projects / Reading Concordances in the 21st Century (RC21) /

RC21 Blogs

< Research

Current Projects

Reading Concordances in the
21st Century (RC21)

RC21 Blogs

RC21 Events

Publications

Talks

Doctorates

Concordance Reading × Association Measures



11. June 2025 Category: [RC21](#)

Concordance Reading × Association Measures Methods to Organize a Dataset of German Support Verb Constructions Author: Xinyao Lu, Friedrich-Alexander-Universität Erlangen-Nürnberg Published: 11 June 2025 Introduction How can we collect a dataset of German Support Verb Constructions (SVCs) fr...

Continue >

Meeting corpus users' needs



Meeting corpus users' needs Author: Yukio Tono (Tokyo University of Foreign Studies) Published: 17 January 2025 As language tools evolve in the digital age, having the right tools can make all the difference in how we interact with and analyze language. But what makes a tool truly effective...

Get in touch with us to contribute to our project blog!

- experience with FlexiConc
- concordance reading for your own research
- needs/ideas for algorithms