# PyDelphin (and related software) Updates

Michael Wayne Goodman    Nanyang Technological University

# What is PyDelphin?

PyDelphin is a Python library implementing many DELPH-IN technologies and formats:

- MRS, DMRS, and EDS representations with $\sim$ 14 serialization formats
- [incr tsdb()] test suites (corpus databases) and TSQL queries
- TDL grammar inspection / formatting
- REPP-based tokenization
- ACE wrapper for parsing/transfer/generation
- Web API, server and client
- Command-line interface for several common tasks
- more...

# What is PyDelphin?

- Project began in 2008
- Version 1.0 released in 2019, now version 1.3
- MIT-licensed
- Has extensive unit testing and full API documentation
- Extensible with plugins (example later)

# What’s new since Cambridge?

- 10 new versions, but mostly small bug fixes and better error messages
- `delphin.cli` namespace for plugins adding `delphin` subcommands
- Improved REPP performance
- Minimum Python version is 3.6

# delphin.edm

New plugin module[1] implementing Elementary Dependency Match (EDM)

- Reads text files of EDS or [incr tsdb()] profiles
- Also works with MRS or DMRS representations
- Weighting system allows one to recreate the exact scores of either the Dridan and Oepen, 2011 system or the newer mtool[2] implementation

```
$ delphin edm --format mrs \
>         ~/grammars/erg-2020/tsdb/gold/mrs/ \
>         ~/grammars/erg-trunk/tsdb/gold/mrs/
Precision:      1.0
   Recall:      1.0
  F-score:      1.0
```

[1] https://github.com/delph-in/delphin.edm

[2] https://github.com/cfmrp/mtool

# Penman 1.0

Penman 1.0 for DMRS and EDS serialization in the PENMAN notation:

```
(e2 / _open_v_1
    :type e
    :ARG1 (x3 / _window_n_1
        :BV-of (_1 / _the_q)
        :type x))
```

- Work with data as trees or graphs
- Semantic model (similar to SEM-I) used for validation
- Built-in transformations could be useful for simplifying DMRS/EDS

# Next Steps

What comes next for PyDelphin? (not rhetorical)

- DMRS advancements? (ICONS, unexpressed nodes, ...)
- Scope tree enumeration?
- Convert to DM (bilexical dependencies)?
- Actual parsing? (compute type hierarchy, unification, inflectional morphology, ...)
- Other?

Install the latest version from PyPI:

- `pip install pydelphin`
- `pip install delphin.highlight delphin-latex delphin.edm`

Read the documentation:

- `https://pydelphin.readthedocs.io/`

Contribute:

- `https://github.com/delph-in/pydelphin`