PyGEDM

D. C. Price

Mar 04, 2026

API INDEX:

CHAPTER

ONE

PYGEDM

Python bindings for the YMW16, NE2001, NE2025 and YT2020 electron density models

This package is a Python interface to the YMW16, NE2001, NE2025 electron density models, and YT2020 halo model. The Yao, Manchester and Wang (2017, Astrophys. J., 835, 29; arXiv:1610.09448) YMW16 electron density model, is written in C++, and the Cordes and Lazio (2001, arXiv:0207156) NE2001 model is written in FORTRAN. This package, PyGEDM, wraps these two codes using pybind11 to make them usable from Python. Here, we have converted NE2001 to C++ using `f2c`. This package also provides an interface to the NE2025 package arxiv:2602.11838

1.1 PyGEDM paper

PyGEDM is detailed in Price, D. C., Flynn, C., and Deller, A. (2021):

Price, D. C., Flynn, C., and Deller, A., "A comparison of Galactic electron density models using PyGEDM", *Publications of the Astronomical Society of Australia*, vol. 38, 2021. doi:10.1017/pasa.2021.33.

If you use PyGEDM, be sure to reference the underlying NE2001, NE2025, YMW16 and YT2020 codes.

1.2 Web app

We provide a web app at https://apps.datacentral.org.au/pygedm/

The pygedm web app is kindly hosted by Data Central.

1.3 Usage

Some usage examples can be found in the examples directory.

```
import pygedm

# calculate DM at a given distance
DM, tau_sc = pygedm.dist_to_dm(204.0, -6.5, 200, method='ne2025')
DM, tau_sc = pygedm.dist_to_dm(204.0, -6.5, 200, method='ne2001p')
DM, tau_sc = pygedm.dist_to_dm(204.0, -6.5, 200, method='ne2001')
DM, tau_sc = pygedm.dist_to_dm(204.0, -6.5, 200, method='ymw16')

# calculate distance for a given sky position and DM
dist, tau_sc = pygedm.dm_to_dist(123.4, 4.0, 200)

# calculate N_e density at xyz galactocentric coordinates
ne = pygedm.calculate_electron_density_xyz(1.0, 2.0, 3.0)
```

(continues on next page)

(continued from previous page)

```
# calculate N_e density at Galactic lat/long/distance coords
ne = pygedm.calculate_electron_density_lbr(204.0, -6.5, 3000.0)

# Calculate halo DM contribution
dm_halo = pygedm.calculate_halo_dm(gl=0, gb=30)
```

The methods return astropy Quantities, which have units attached, and can accept astropy Angles and Quantities as arguments:

```
import pygedm
import astropy.units as u
import astropy.coordinates as c

DM = u.Quantity(10.0, unit='pc cm^-3')
ra, dec = c.Angle(23.0, unit='hourangle'), c.Angle('-43:00:02', unit='degree')
sky_coords = c.SkyCoord(ra, dec, frame='icrs')
dist, tau_sc = pygedm.dm_to_dist(sky_coords.galactic.l, sky_coords.galactic.b, DM)

print(dist.to('lyr'))
>> 3362.16343117 lyr
print(tau_sc.to('ns'))
>> 7.758686138 ns
```

1.4 Installation

Requires `pybind11`, `astropy`, `numpy`, `scipy`, a newish C compiler with C++11 support (Ubuntu 16.04+ default gcc will work), plus `f2c`.

Basic installation is via `pip`:

```
pip install pygedm
```

or

```
pip install git+https://github.com/telegraphic/pygedm
```

to install the latest version from github. Alternatively, download this repository and install via

```
pip install .
```

1.4.1 Linux

`f2c` can be installed via `apt-get f2c` in Ubuntu, or via `conda install -c conda-forge f2c` if you use conda.

1.4.2 Mac OSX

For MacOS, you are best off using `conda` and getting `f2c` via `conda install -c conda-forge f2c`.

Some users have had success using `f2c` in macports, which installs into `/opt/local/include` and `/opt/local/lib`. If using macports, the install command is:

```
pip install --global-option=build_ext --global-option="-I/opt/local/include" pygedm
```

1.4.3 Windows

Windows is not currently supported.

1.4.4 Unit tests

To run unit tests, run `python setup.py test`. Note that these tests only check the Python bindings, not the underlying C/FORTRAN source code (which is not supplied with unit tests).

1.5 References

If using PyGEDM in a journal article, please remember to cite the underlying electron density models:

Cordes, J. M., & Lazio, T. J. W. (2002), *NE2001.I. A New Model for the Galactic Distribution of Free Electrons and its Fluctuations*, arXiv e-prints, astro-ph/0207156.

Cordes, J. M., & Lazio, T. J. W. (2003), *NE2001. II. Using Radio Propagation Data to Construct a Model for the Galactic Distribution of Free Electrons*, arXiv e-prints, astro-ph/0301598.

Yao, J. M., Manchester, R. N., & Wang, N. (2017), *A New Electron-density Model for Estimation of Pulsar and FRB Distances*, The Astrophysical Journal, Volume 888, Issue 2, id.105, Colume 835, id.29

Yamasaki, S., & Totani, T. (2020), *The Galactic Halo Contribution to the Dispersion Measure of Extragalactic Fast Radio Bursts*, The Astrophysical Journal, Volume 888, Issue 2, id.105

S.K. Ocker, J.M. Cordes (2026), *NE2025: An Updated Electron Density Model for the Galactic Interstellar Medium* arXiv e-prints, astro-ph/2602.11838

These are available in bibtex format in references.bib, and also as an ADS library.

YMW16 is a model for the distribution of free electrons in the Galaxy, the Magellanic Clouds and the inter-galactic medium, that can be used to estimate distances for real or simulated pulsars and fast radio bursts (FRBs) based on their position and dispersion measure.

The Galactic model is based on 189 pulsars that have independently determined distances as well as dispersion measures, whereas simpler models are used for the electron density in the MC and the IGM. It is estimated that the 95% of predicted Galactic pulsar distances will have a relative error of less than a factor of 0.9. Pulse broadening due to scattering in the Galactic interstellar medium, the Magellanic Clouds, the intergalactic medium and FRB host galaxies is estimated.

As well as the ymw16 dm-distance program, we also provide a program, ymw16_ne, which gives the electron density at any point in the Galaxy or Magellanic Clouds.

A paper (Yao, Manchester and Wang, 2017, Astrophys. J., 835, 29; arXiv:1610.09448) describes the model and compares its predictions with those of earlier Galactic electron density models. YMW16 is the first electron-density model to estimate extragalactic pulsar distances and FRB distances.

To make a command-line executable version of the program, download and unpack the latest version of the program. Then run "make_ymw16" to create the executable. The environment variable YMW16_DIR may be set up to point at a directory containing ymw16par.txt and spiral.txt. Access to these files is needed at runtime.

Websites allowing interactive access to the YMW16 distance model and download of the latest program version are available at:

- http://www.xao.ac.cn/ymw16/,
- http://www.atnf.csiro.au/research/pulsar/ymw16/ and

- https://bitbucket.org/psrsoft/ymw16/.

Please report any issues or bugs at https://bitbucket.org/psrsoft/ymw16/issues/new/ or directly to the authors. Please provide an example illustrating the problem.

1.6 YMW16 C LICENSE

```
Copyright (C) 2016, 2017  J. M. Yao, R. N. Manchester, N. Wang.

YMW16 is free software: you can redistribute it and/or modifyit
under the terms of the GNU General Public License as published by the
Free Software Foundation, either version 3 of the License, or (at your
option) any later version.

YMW16 is distributed in the hope that it will be useful,but WITHOUT
ANY WARRANTY; without even the implied warranty ofMERCHANTABILITY or
FITNESS FOR A PARTICULAR PURPOSE. See theGNU General Public License,
available at http://www.gnu.org/licenses/, for more details.

Jumei Yao (yaojumei _@_ xao.ac.cn), Richard N Manchester
(dick.manchester _@_ csiro.au), Na Wang (na.wang _@_ xao.ac.cn)
```

07 July 2002 To compile and execute the code, see code.pdf.

1.6.1 pygedm.pygedm

Python API to YMW16, NE2001, and YT2020 Galactic electron density models

References

[1] Cordes, J. M., & Lazio, T. J. W. (2002), *NE2001.I. A New Model for the Galactic Distribution of Free Electrons and its Fluctuations*, arXiv e-prints, astro-ph/0207156.

[2] Cordes, J. M., & Lazio, T. J. W. (2003), *NE2001. II. Using Radio Propagation Data to Construct a Model for the Galactic Distribution of Free Electrons*, arXiv e-prints, astro-ph/0301598.

[3] Yao, J. M., Manchester, R. N., & Wang, N. (2017), *A New Electron-density Model for Estimation of Pulsar and FRB Distances*, ApJ, 835, 29.

[4] Yamasaki S, Totani T (2020), *The Galactic Halo Contribution to the Dispersion Measure of Extragalactic Fast Radio Bursts* The Astrophysical Journal, Volume 888, Issue 2, id.105

pygedm.pygedm.**calculate_electron_density_lbr**(*gl*, *gb*, *dist*, *method='ymw16'*)

Calculate electron density at a point with Galactic coords (ga, gl) at given distance

Parameters

- **gl** (*float, Angle, or Quantity*) – Galactic longitude in degrees (or astropy Angle)
- **gb** (*float, Angle, or Quantity*) – Galactic latitude in degrees (or astropy Angle)
- **dist** (*float or Quantity*) – Distance in pc

Returns
electron density in cm^-3

Return type
N_e (astropy.Quantity)

pygedm.pygedm.**calculate_electron_density_xyz**(*x*, *y*, *z*, *method='ymw16'*)

Calculate electron density at a point with galactocentric coords (x, y, z)

Parameters

- **x** (*float or Quantity*) – galactocentric X coordinate in pc
- **y** (*float or Quantity*) – galactocentric Y coordinate in pc
- **z** (*float or Quantity*) – galactocentric Z coordinate in pc

Returns
electron density in cm^-3

Return type
N_e (astropy.quantity)

pygedm.pygedm.**calculate_halo_dm**(*gl*, *gb*, *method='yt2020'*, *component='both'*)

Compute halo DM

Parameters

- **gl** (*float, Angle, or Quantity*) – Galatic latitude in degrees (or astropy Angle)
- **gb** (*float, Angle, or Quantity*) – Galactic latitude in degrees (or astropy Angle)
- **method** (*str*) – one of 'yt2020' (only YT2020 supported currently)
- **component** (*str*) – Compute 'spherical' component of halo, 'disk', or 'both' components.

Returns
Dispersion measure in (pc/cm3)

Return type
DM (float)

pygedm.pygedm.**convert_lbr_to_xyz**(*gl*, *gb*, *dist*, *method='ymw16'*)

Convert Galactic (l,b,r) coords to Galactocentric (x,y,z) coords

Parameters

- **gl** (*float, Angle, or Quantity*) – Galatic longitude in degrees (or astropy Angle)
- **gb** (*float, Angle, or Quantity*) – Galactic latitude in degrees (or astropy Angle)
- **dist** (*float or Quantity*) – Distance in pc
- **method** (*str*) – one of 'ymw16', 'ne2001', or 'astropy'

Returns
Galactocentric X, Y, Z coordinates

Return type
xyz (tuple)

Notes

For transform, the Sun is located at (x=0, y=R_sun, z=z_sun) YMW16 uses R_sun of 8300 pc and z_sun of 6.0 pc NE2001 uses R_sun of 8500 pc and z_sun of 0.0 pc Both of these do a basic spherical to cartesian conversion.

astropy does a much more complicated conversion, see https://astropy.readthedocs.io/en/latest/coordinates/galactocentric.html This is the 'proper' coordinate system, but note that it is NOT COMPATIBLE WITH NE2001 OR YMW16! (!SEE EXAMPLE OUTPUT BELOW!)

Example output:

```
pygedm.convert_lbr_to_xyz(0, 0, 0, method='ymw16')
(<Quantity 0. pc>, <Quantity 8300. pc>, <Quantity 6. pc>)

pygedm.convert_lbr_to_xyz(0, 0, 0, method='ne2001')
(<Quantity 0. pc>, <Quantity 8500. pc>, <Quantity 0. pc>)

pygedm.convert_lbr_to_xyz(0, 0, 0, method='astropy')
(<Quantity -8499.95711754 pc>, <Quantity 0. pc>, <Quantity 27. pc>)
```

pygedm.pygedm.**dist_to_dm**(*gl*, *gb*, *dist*, *mode='gal'*, *method='ymw16'*, *nu=1.0*)

Convert a distance to a DM

Parameters

- **gl** (*float in deg or astropy.Angle*) – galactic longitude
- **gb** (*float in deg or astropy.Angle*) – galactic latitude
- **dist** (*float or astropy.Quantity*) – distance to source (pc) or if in mode IGM use (Mpc)
- **method** (*str*) – choose electron density model, either 'ymw16' or 'ne2001'
- **mode** (*str*) – Gal, MC, or IGM (for YMW16 only)
- **nu** (*float in GHz or astropy.Quantity*) – observing frequency (GHz)

Returns

Dispersion measure (pc / cm3), scattering timescale at 1 GHz (s)

Return type

dm (astropy.Quantity), tau_sc (astropy.Quantity)

pygedm.pygedm.**dm_to_dist**(*gl*, *gb*, *dm*, *dm_host=0*, *mode='gal'*, *method='ymw16'*, *nu=1.0*)

Convert a DM to a distance

Parameters

- **gl** (*float in deg or astropy.Angle*) – galactic longitude
- **gb** (*float in deg or astropy.Angle*) – galactic latitude
- **dm** (*float in pc/cm3 or astropy.Quantity*) – dispersion measure (pc cm^-3)
- **method** (*str*) – choose electron density model, either 'ymw16' or 'ne2001'
- **mode** (*str*) – Gal, MC, or IGM (for YMW16 only)
- **nu** (*float in GHz or astropy.Quantity*) – observing frequency (GHz)

Returns

Distance (pc), scattering timescale at 1 GHz (s)

Return type

dist (astropy.Quantity), tau_sc (astropy.Quantity)

pygedm.pygedm.**generate_healpix_dm_map**(*dist=1*, *nside=64*, *method='ymw16'*)

Generate an all-sky healpix map for a given distance.

Parameters

- **dist** (*float or Quantity*) – Distance to integrate EDM out to. 30 kpc will go to edge

- **nside** (*int*) – The NSIDE parameter for the healpix map (power of 2, larger => higher resolution)
- **method** (*str*) – one of 'ymw16', 'ne2001', 'ne2025', 'yt2020' or 'yt2020_analytic'

Notes

This method takes a fair amount of time to run – tens of seconds for NSIDE=32. YT2020 method is even slower, consider using yt2020_analytic

Returns
Healpix map as a numpy array (1D), which can be viewed using the healpy.mollview() method

Return type
hmap (np.array)

1.6.2 pygedm.ne2001_wrapper

Python wrapper for NE2001 model code.

References

[1] Cordes, J. M., & Lazio, T. J. W. (2002), *NE2001.I. A New Model for the Galactic Distribution of Free Electrons and its Fluctuations*, arXiv e-prints, astro-ph/0207156.

[2] Cordes, J. M., & Lazio, T. J. W. (2003), *NE2001. II. Using Radio Propagation Data to Construct a Model for the Galactic Distribution of Free Electrons*, arXiv e-prints, astro-ph/0301598.

`pygedm.ne2001_wrapper.TAUISS`(*d*, *sm*, *nu*)

Convert a scattering measure (SM) to scattering timescale at given frequency.

Direct port from FORTRAN code scattering98.f

Parameters

- **d** (*float*) – Distance in kpc
- **sm** (*float*) – Scattering measure (kpc m^{-20/3})
- **nu** (*float*) – Radio frequency in GHz

Returns
pulse broadening time (ms)

Return type
tauiss (float)

Fortran equiv:

```
      REAL FUNCTION TAUISS(d, sm, nu)
c
c calculates the pulse broadening time in ms
c from distance, scattering measure, and radio frequency
c
c input:      d = pulsar distance       (kpc)
c            sm = scattering measure    (kpc m^{-20/3})
c            nu = radio frequency       (GHz)
c output: tausis = pulse broadening time (ms)
c
      implicit none
```

(continues on next page)

(continued from previous page)

```
    real d, sm,  nu
    tauiss = 1000. * (sm / 292.)**1.2 * d * nu**(-4.4)
    end
```

pygedm.ne2001_wrapper.**TRANSITION_FREQUENCY**(*d_eff: float*, *sm: float*, *xi: float = None*) → float

The transition frequency between weak and strong scattering

Python implementation of Equation 17 in C&L 2002 [1].

Computes nu = (318 GHz) * xi^(10/17) * sm^(6/17) * d_eff^(5/17)

Parameters

- **d_eff** (*float*) – effective distance to the scattering medium (kpc)
- **sm** (*float*) – Scattering measure (kpc m^{-20/3})
- **xi** (*float*) – Fresnel scale scaling factor (~1). If not supplied, sqrt(2 pi) is used as preferred definition.

Returns
Transition frequency in GHz

Return type
nu (float)

pygedm.ne2001_wrapper.**calculate_electron_density_xyz**(*x*, *y*, *z*)

Compute electron density at Galactocentric X, Y, Z coordinates

x,y,z are Galactocentric Cartesian coordinates, measured in kpc (NOT pc!) with the axes parallel to (l, b) = (90, 0), (180, 0), and (0, 90) degrees

Parameters

- **x** (*float*) – Galactocentric coordinates in kpc
- **y** (*float*) – Galactocentric coordinates in kpc
- **z** (*float*) – Galactocentric coordinates in kpc

Returns
Electron density in cm-3

Return type
ne_out (astropy.Quantity)

pygedm.ne2001_wrapper.**dist_to_dm**(*l*, *b*, *dist*, *nu=1.0*, *full_output=False*)

Convert distance to DM and compute scattering timescale

Parameters

- **l** (*float*) – galactic longitude in degrees
- **b** (*float*) – galactic latitude in degrees
- **dist** (*float*) – Distance in kpc
- **nu** (*float in GHz or astropy.Quantity*) – observing frequency (GHz)
- **full_output** (*bool*) – Return full raw output (dict) from NE2001 if set to True

Returns
Dispersion measure (pc / cm3), scattering timescale at 1 GHz (s)

Return type
dm (astropy.Quantity), tau_sc (astropy.Quantity)

pygedm.ne2001_wrapper.**dm_to_dist**(*l*, *b*, *dm*, *nu=1.0*, *full_output=False*)

Convert DM to distance and compute scattering timescale

Parameters

- **l** (*float*) – galactic longitude in degrees
- **b** (*float*) – galactic latitude in degrees
- **dm** (*floa*) – Dispersion measure
- **nu** (*float in GHz or astropy.Quantity*) – observing frequency (GHz)
- **full_output** (*bool*) – Return full raw output (dict) from NE2001 if set to True

Returns
Distance (pc), scattering timescale at 1 GHz (s)

Return type
dist (astropy.Quantity), tau_sc (astropy.Quantity)

pygedm.ne2001_wrapper.**run_from_pkgdir**(*f*)

Decorator function to chdir() into package directory when running

NE2001 code doesn't know the relative path to its data files. This wraps the function call, changing into the right directory first, calling it, then changing back to original directory.

1.6.3 pygedm.ymw16_wrapper

Python/C++ port of YMW16 C code

References

[1] Yao, J. M., Manchester, R. N., & Wang, N. (2017), *A New Electron-density Model for Estimation of Pulsar and FRB Distances*, ApJ, 835, 29.

pygedm.ymw16_wrapper.**calculate_electron_density_lbr**(*gl*, *gb*, *dist*)

Calculate electron density at a point with Galactic coords (ga, gl)
at a given distance in pc

Parameters

- **gl** (*float, Angle, or Quantity*) – Galatic longitude in degrees (or astropy Angle)
- **gb** (*float, Angle, or Quantity*) – Galactic latitude in degrees (or astropy Angle)
- **dist** (*float or Quantity*) – Distance in pc

Returns
electron density in cm^-3

Return type
N_e (astropy.Quantity)

pygedm.ymw16_wrapper.**calculate_electron_density_xyz**(*x*, *y*, *z*)

Calculate electron density at a point with galactocentric coords (x, y, z)

Parameters

- **x** (*float or Quantity*) – galactocentric X coordinate in pc
- **y** (*float or Quantity*) – galactocentric Y coordinate in pc
- **z** (*float or Quantity*) – galactocentric Z coordinate in pc

Returns
electron density in cm^-3

Return type
N_e (astropy.quantity)

pygedm.ymw16_wrapper.**dist_to_dm**(*gl*, *gb*, *dist*, *mode='gal'*, *nu=1.0*)

Convert a distance to a DM

Parameters

- **gl** (*float in deg or astropy.Angle*) – galactic longitude
- **gb** (*float in deg or astropy.Angle*) – galactic latitude
- **dist** (*float or astropy.Quantity*) – distance to source (pc) or if in mode IGM use (Mpc)
- **mode** (*str*) – Gal, MC, or IGM (for YMW16 only)
- **nu** (*float in GHz or astropy.Quantity*) – observing frequency (GHz)

Returns
dispersion measure (pc/cm3) and scattering time scale (s)

Return type
dm (astropy.Quantity), tau_sc (astropy.Quantity)

pygedm.ymw16_wrapper.**dm_to_dist**(*gl*, *gb*, *dm*, *dm_host=0*, *mode='gal'*, *nu=1.0*)

Convert a DM to a distance

Parameters

- **gl** (*float in deg or astropy.Angle*) – galactic longitude
- **gb** (*float in deg or astropy.Angle*) – galactic latitude
- **dm** (*float in pc/cm3 or astropy.Quantity*) – dispersion measure (pc cm^-3)
- **mode** (*str*) – Gal, MC, or IGM (for YMW16 only)
- **nu** (*float in GHz or astropy.Quantity*) – observing frequency (GHz)

Returns
distance (pc) and scattering time scale (s)

Return type
dist (astropy.Quantity), tau_sc (astropy.Quantity)

1.6.4 pygedm.yt2020

Python implementation of Yamasaki & Totani DM Halo model

References

[1] Yamasaki S, Totani T (2020), *The Galactic Halo Contribution to the Dispersion Measure of Extragalactic Fast Radio Bursts* The Astrophysical Journal, Volume 888, Issue 2, id.105

Notes

Adapted from S. Yamasaki's `DM_halo_yt2020_numerical.py` command-line python code

`pygedm.yt2020.calculate_halo_dm`(*l*, *b*, *component='both'*)

Compute halo DM

Parameters

- **l** (*float*) – Galactic longitude, in degrees (-180 to +180)
- **b** (*float*) – Galactic latitude, in degrees (-90 to 90)
- **component** (*str*) – Compute 'spherical' component of halo, 'disk' component, or 'both' components.

Returns
Dispersion measure in [pc/cm^3]

Return type
DM (float)

`pygedm.yt2020.calculate_halo_dm_analytic`(*l*, *b*)

Calculate the DM contribution of the Galactic halo.

Uses an analytical formula for speed. Useful for all-sky mapping.

Parameters

- **l** (*float*) – Galactic longitude, in degrees (-180 to +180)
- **b** (*float*) – Galactic latitude, in degrees (-90 to 90)

`pygedm.yt2020.ne_disk`(*l*, *b*, *s*)

Compute electron density for spherical component for (l, b) at distance s

Parameters

- **l** (*float*) – Galactic longitude, in radians (-pi to +pi)
- **b** (*float*) – Galactic latitude, in radians (-pi/2 to pi/2)
- **s** (*float*) – Distance (kpc)

Returns
electron density in cm^{-3}

Return type
ne (float)

`pygedm.yt2020.ne_sphe`(*l*, *b*, *s*)

Compute electron density for spherical component for (l, b) at distance s

Parameters

- **l** (*float*) – Galactic longitude, in radians (-pi to +pi)
- **b** (*float*) – Galactic latitude, in radians (-pi/2 to pi/2)
- **s** (*float*) – Distance (kpc)

Returns
electron density in cm^{-3}

Return type
ne (float)

pygedm.yt2020.**s_max**(*l*, *b*)

Compute integration limit s_max for given sky coordinates

Parameters

- **l** (*float*) – Galactic longitude, in radians (-pi to +pi)
- **b** (*float*) – Galactic latitude, in radians (-pi/2 to pi/2)

Returns

s_max (float), maximum integration limit corresponsing to r = r_vir

PYTHON MODULE INDEX

p

INDEX