**pyglow: Upper atmosphere climatological models in Python**

Abstract

Climatological numerical models (e.g., IGRF, IRI, MSIS, HWM93/07), commonly written in FORTRAN, are often used within the upper atmosphere research community for a wide variety of applications (e.g., initializing values in numerical models and for comparisons against measured data). Here we present pyglow, a Python module that includes wrappers for climatological models. In this way, the climatological models can be accessed in the high-level and scriptable Python programming language. This implementation provides quick access to each model under a common framework. Pyglow is open-sourced (https://github.com/timduly4/pyglow/) to encourage improvement and expansion of the code base.

Introduction

- **There are primarily two different methods to access climatological models:**
  - Compiling the FORTRAN code or using a web service provided by the Community Coordinated Modeling Center (CCMC)
  - Both offer various levels of computer architecture abstraction

  - In terms of abstraction level, pyglow is between these two, offering access to climatological models in the scriptable Python programming language

  - **pyglow offers access to several climatological models for various upper atmosphere parameters**
  - **Future models can be easily implemented as they become available to the research community**

- **F2py (available in numpy as np)** is used as the “glue” between the pyglow module and the Fortran models
- **Access to models under common framework enables derived parameters to be calculated (e.g., airglow emission)** and synergy between models (e.g., integrating electron densities along magnetic field line)

Example Usage

- Example Python script using pyglow
  - `# import all climatological models` and `# and electron density`

- **To plot profiles of airglow emission** and `# and electron density`

- `from matplotlib.pyplot import *` and `import numpy as np`

- `from pyglow import Point, IRI, MSIS, HWM93, HWM07`

- `pyglow.r藩Man.update('`-font size: 10`)`

- **Setting lon, lat, and a range of altitudes**
  - `lat = -60.0` and `lon = -60.0`

- `alt = np.linspace(0.0, 1000.0)`

- `for alt in alts` and `for lat in alts`

- `pt = Point(dn, lat, lon, alt)`

- `pt.run_iri()` and `pt.run_msis()`

- `pt.run_ag6300()`

- `ag.append(pt.ag6300)`

- `plt.figure(1, figsize=(7,8)); clf()` and `from matplotlib.pyplot import *`

- `ax1 = plt.subplot(211)` and `ax2 = plt.subplot(212)`

- `polar(ax1, r, alts, theta, c='k', lw = 4, label = r'$V_{630.0}$')`

- `ax1.set_theta_zero_location('North')`

- `ax1.tick_params(labelsize=16)`

- `ax2.set_xlabel('Electron density [items/cm$^3$]')`

- `ax2.set_ylabel('Altitude [km]')`

- `ax2.semilogx(ne, alts, c='k', lw = 4, label = r'$n_e$')`

- `ax2.legend(loc=1)`

- `ylim([alts[0], alts[-1]])`

- **In this example, profiles of the electron density and 630.0-nm airglow emission are produced in 50 lines of code!**

Additional Examples

- **pyglow is used to calculate a dip angle contour map in conjunction with the Basemap package for producing high quality maps**

- **Combining the climatological models and Basemap under a common framework provide quick development of results**

References