e3doubt (aka ISgeometry for python)

A python- and R-based tool for EISCAT_3D experiment design and uncertainty analysis

S. M. Hatch[1], I. Virtanen[2], ISSI Team 506[3]

2023/11/30

[1]Department of Physics and Technology, University of Bergen, Bergen, Norway
[2]Space Physics and Astronomy Research Unit, University of Oulu, Oulu, Finland
[3]International Space Science Institute, Bern, Switzerland (link to team website)

Outline

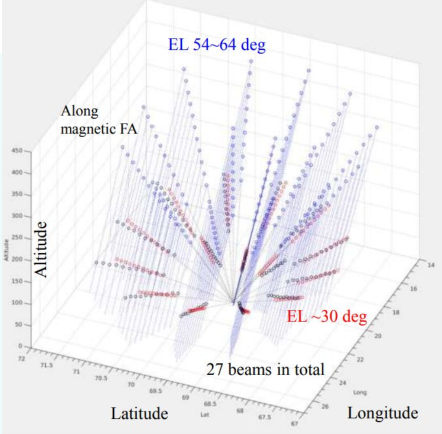
- Motivation for e3doubt: a python frontend to ISgeometry
- In(put)s and Out(put)s of e3doubt
- Demonstration I: Uncertainties of maps of ionospheric convection reconstructed from E3D measurements using SECS
- Demonstration II: Uncertainty of calculations of electromagnetic work

Motivation

- EISCAT_3D is an extremely advanced ISR system.
  - How on earth can we (plan to) use it?
- Ilkka Virtanen has written a powerful set of tools in R for ISR uncertainty analysis: the ISgeometry package
  - How on earth do we use ISgeometry without having to learn R?
- NRC-funded four-year E3D-BRITE* project at UiB begins Q1 2024
  - We need tools to perform E3D-based reconstruction of IT dynamics
- Enter: e3doubt

*EISCAT_3D-Based Reconstruction of Ionosphere-Thermosphere Electrodynamics

In(put)s and Out(put)s of e3doubt



Credit: Y. Ogawa

Required inputs

- Azimuth and elevation for each beam
- Heights at which to sample each beam
- Default is ~Ogawa's suggested CP1 program from last year

Optional inputs

- Information about Tx, Rx (location, min el, FWHM, power, duty cycles, T_{noise}, ...)
  - Can specify Tx site and completely arbitrary combination of Rx sites
- Relative beam dwell times
- Range resolution for each altitude
- Ionosphere and atmosphere parameters
  - Default is IRI and MSIS via [iri2016](iri2016) and pymsis
- … And many more (refer to Ilkka's talk)

In(put)s and Out(put)s of e3doubt - 2

Possible workflow

1. Select elevations and azimuths
2. Initialize ‘Experiment’ object
   e.g., `exp = Experiment(az=az,el=el)`
3. Run IRI and MSIS models, calculate collision frequency
   `exp.run_models()`
4. Calculate plasma parameter uncertainties using ISgeometry
   `unc = exp.get_uncertainties(integrationsec=600)`
5. Perform analysis using uncertainty estimates
   …

In(put)s and Out(put)s of e3doubt - 3

Custom ionosphere and atmosphere parameters: set_ionos/set_atmos functions

```python
def get_datacov_e3doubt(ddict):

    exp = e3doubt.Experiment(el=ddict['el'], az = ddict['az'],h=ddict['alts'])

    exp.run_models()

    exp.set_ionos('ne',ddict['ne'])
    exp.set_ionos('Te',ddict['Te'])
    exp.set_ionos('Ti',ddict['Ti'])

    uncert =exp.calc_uncertainties(integrationsec=5*60)

    cov = exp.get_velocity_cov_matrix()

    ddict['cov_vi'] = cov
    ddict['var_ne'] = uncert.dnemulti.values

    return ddict
```

Credit: J. Reistad

In(put)s and Out(put)s of e3doubt - 4

Output of get_uncertainties()

```
In [7]: df = exp.get_uncertainties()

In [8]: df.columns
Out[8]:
Index(['dne1', 'dne2', 'dne3', 'dnemulti', 'dTe1', 'dTe2', 'dTe3', 'dTemulti',
       'dTi1', 'dTi2', 'dTi3', 'dTimulti', 'dVi1', 'dVi2', 'dVi3', 'dVimulti'],
      dtype='object')

In [9]: df
Out[9]:
             dne1          dne2          dne3  ...        dVi2        dVi3     dVimulti
0    2.775050e+09  1.209054e+10  1.328753e+10  ...   97.543539  107.200598   364.616562
1    1.004876e+10  8.203632e+09  8.750000e+09  ...  136.358919  145.440520   727.078870
2    1.116370e+10  9.308450e+09  9.745883e+09  ...  222.350262  232.799176  1308.073572
3    1.601766e+10  1.184437e+10  1.212833e+10  ...  288.004052  294.908699  2066.109336
4    4.124264e+10  2.217895e+10  2.203723e+10  ...  230.818517  229.343612  2442.379623
..            ...           ...           ...  ...         ...         ...          ...
238  4.057360e+10  1.860394e+10  1.920834e+10  ...  196.892688  203.289253  2066.275313
239  1.104095e+11  4.237233e+10  4.268149e+10  ...  232.971586  234.671381  3363.021721
240  1.520652e+11  6.531773e+10  6.502533e+10  ...  270.932762  269.719925  4226.699972
241  1.390351e+11  7.586721e+10  7.544972e+10  ...  406.871173  404.632213  6215.150742
242  1.157246e+11  7.850811e+10  7.826261e+10  ...  624.767342  622.813650  9547.465558

[243 rows x 16 columns]
```

In(put)s and Out(put)s of e3doubt - 5

Lots of helper functions

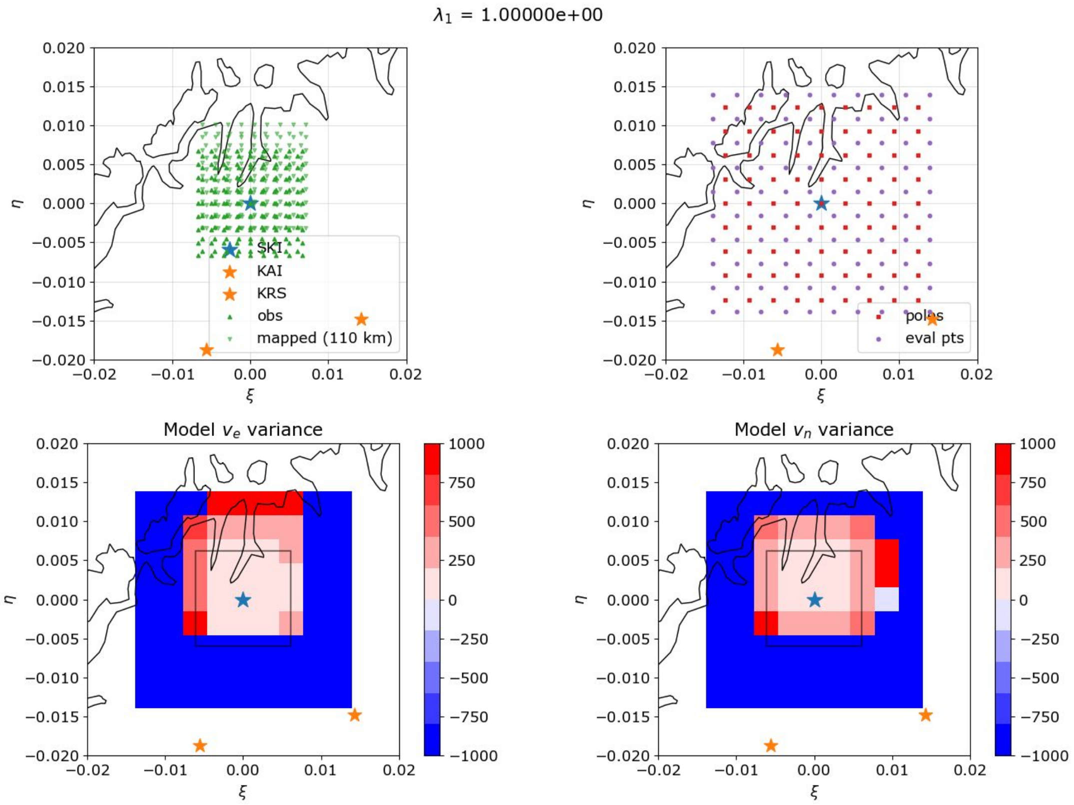
- get_velocity_cov_matrix
  - velocity covariance matrix in ENU or ECEF coordinates for each point
- get_beam_info
  - Azimuth, elevation, dwell time, and beam number for each beam
- get_atmos
  - Get a pandas DataFrame containing all atmospheric parameters
- get_ionos
  - Get pandas DataFrame containing all ionospheric parameters
- get_points
  - az, el, h, beam, gdlat gclat, glon, xecef, yecef, zecef, resR
- radar_utils.py, geodesy.py
  Many tools for radar geometry and geodesy calculations

Demonstration I: Map of convection uncertainty

- 81(!) beams on cubedsphere grid
- *Covariance* of **v**_perp from E3D mapped to 110 km using Apex basis vectors:

$$\Sigma_{v'} = \mathbf{B}\Sigma_v\mathbf{B}^T$$

- Reconstruct ionosph. potential using curl-free SECS functions



Demonstration II: $\mathbf{J} \cdot \mathbf{E}$ (EM work) uncertainty

Why do this?

Lots of people are interested in calculating Joule heating $w = \mathbf{J} \cdot (\mathbf{E} + \mathbf{v}_n \times \mathbf{B})$ using E3D. But how precise can an estimate of Joule heating be?

Neglecting $\mathbf{v}_n$ we can get a lower bound on uncertainty of Joule heating

Procedure

1. Define $\mathbf{J} \cdot \mathbf{E} = qn\,(\mathbf{v}_\mathrm{i} - \mathbf{v}_\mathrm{e}) \cdot \mathbf{E} = qn\,\mathbf{v}_\mathrm{i} \cdot \mathbf{E} = qn\,\mathbf{v}_\mathrm{i} \cdot (-\mathbf{v}_\mathrm{e} \times \mathbf{B})$
2. $\mathbf{v}_\mathrm{e}$ and $\mathrm{Cov}(\mathbf{v}_\mathrm{e})$ come from SECS reconstruction on previous slide
3. $\mathbf{v}_\mathrm{i}$ measured by E3D at 110-km altitude
4. Calculate variance of EM work using semi-terrible expression for $\mathrm{Var}(\mathbf{J} \cdot \mathbf{E})$

Semi-terrible expression for Var($\mathbf{J} \cdot \mathbf{E}$)

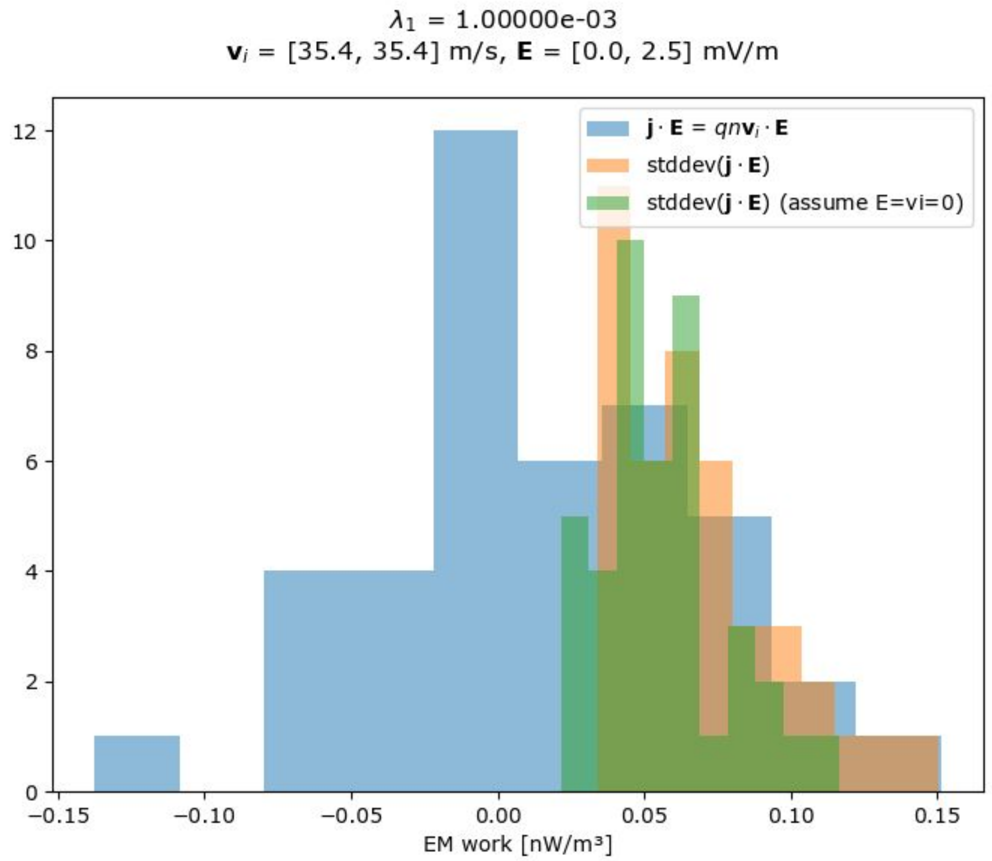
$$\mathrm{Var}(\underline{j} \cdot \underline{E}) = \mathrm{Var}(n \underline{V} \cdot \underline{E})$$

$$= E(n)^2 \left[E(\underline{V})^T \Sigma_E E(\underline{V}) + E(\underline{E})^T \Sigma_V E(\underline{E}) + \mathrm{Tr}(\Sigma_V \Sigma_E) \right]$$

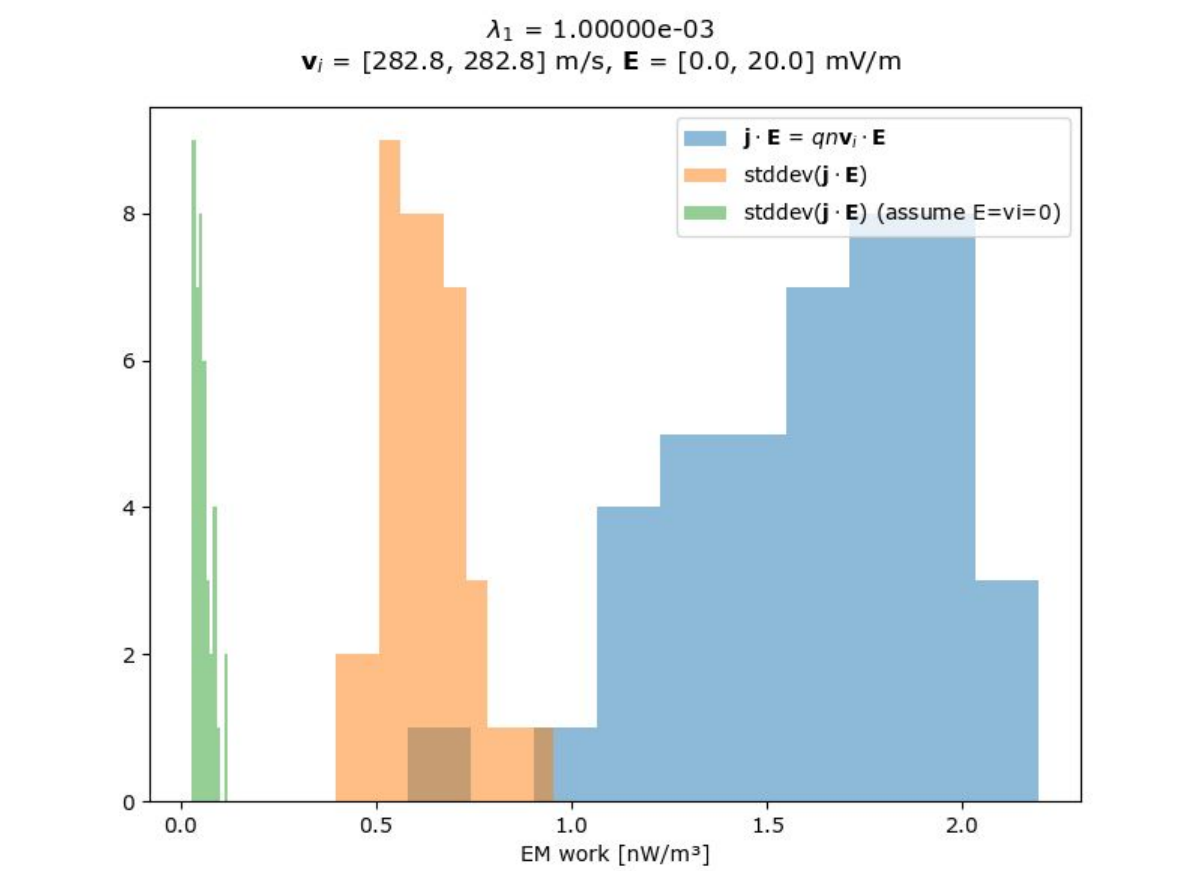
$$+ \sigma_n^2 \Big[E(\underline{E})^T E(\underline{V}) E(\underline{V})^T E(\underline{E}) + E(\underline{E})^T \Sigma_V E(\underline{E}) + E(\underline{V})^T \Sigma_E E(\underline{V})$$

$$+ \mathrm{Tr}(\Sigma_V \Sigma_E) \Big]$$

Demonstration II: **J · E** (EM work) uncertainty - 2



Demonstration II: **J** · **E** (EM work) uncertainty - 2



Give it a try

Repository is available online: https://github.com/Dartspacephysiker/e3doubt

Please file an issue on github or get in touch if you have questions!

Manuscript in preparation …

Citation

Hatch, S. M., & Virtanen, I. (2023). EISCAT_3D Uncertainty Estimation (E3Doubt) (Version v0.2.0a) [Computer software]

https://zenodo.org/badge/latestdoi/711767218

Thank you!