Space physics made **easy**

An overview for the non developers

Alexis Jeandet[1], Nicolas Aunai[1], Benjamin Renard[2], Myriam Bouchemit[2], Nicolas André[2], Vincent Génot[2], Christian Jacquey[2]

How do you get in-situ plasma physics measurements?

Issues when the SciQLop project started (2014)

- Impossible to get all data from a single server (CDA, AMDA ,CSA, CLWeb, PDS,...)
- Many file formats (TAB, CEF, CSV, CSV, CDF, netCDF,…)
- Relative compliance to standards (IST.. what?)
- Few Python packages, usually for **one** mission or **one** server
- Not ready for ML pipelines

Speasy

As easy as:

my_data = **spz.get_data(** *what* **)**

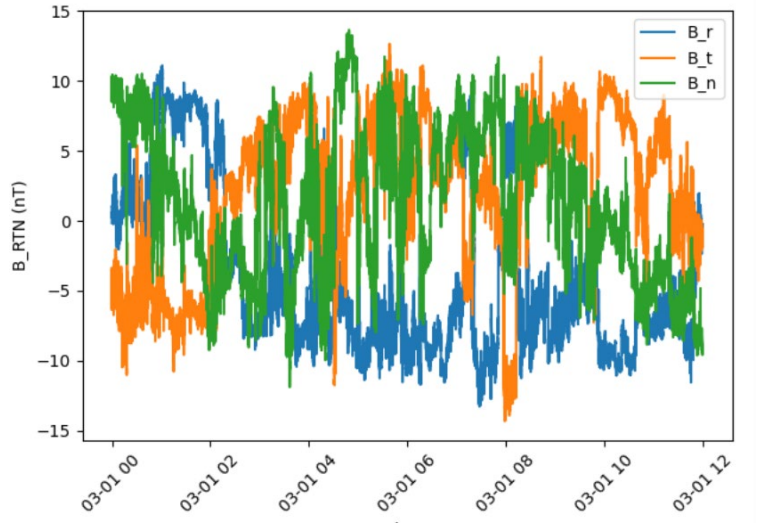
my_data = **spz.get_data(** *what, start_time, stop_time* **)**

Design principles:

- Do not ever change the data
- **E**asy to **U**se **H**ard to **M**isuse (**EUHM**)
- **K**eep **I**t **S**imple, **S**tupid (**KISS**)
- **S**eparation **O**f **C**oncerns (**SOC**)
- Avoid global states (except for cache)
- Performances matter

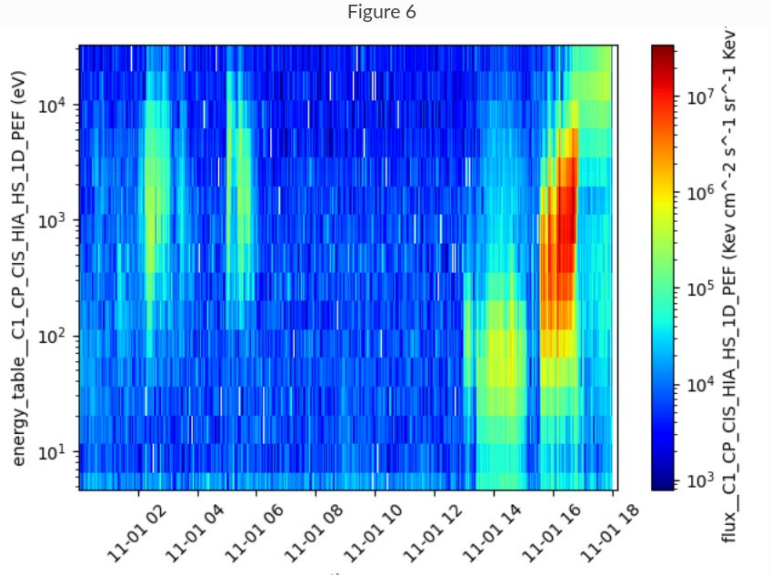
```
solo_fgm: SpeasyVariable = spz.get_data(
    spz.inventories.tree.cda.Solar_Orbiter.SOLO.MAG.SOLO_L2_MAG_RTN_NORMAL.B_RTN,
    "2022-03-01",
    "2022-03-01T12",
)
plt.figure()
solo_fgm.plot()
plt.tight_layout()
plt.show()
```

Figure 1



```
plt.figure()
spz.get_data(
    spz.inventories.tree.csa.Cluster.Cluster_1.CIS_HIA1.C1_CP_CIS_HIA_HS_1D_PEF.flux__C1_CP
    "2006-11-01",
    "2006-11-02",
).plot(cmap="jet")
plt.tight_layout()
plt.show()
```

Figure 6



You can also:

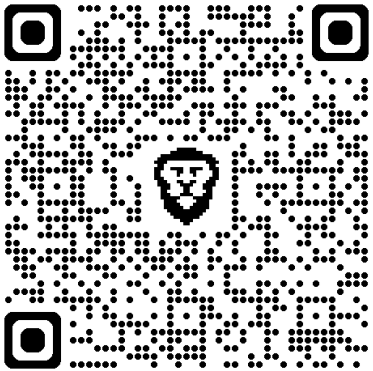
my_datas = **spz.get_data(** *[what,...]* **)**

my_datas = **spz.get_data(** *[what,...], [(start, stop),...]* **)**

```
products = [
    spz.inventories.tree.amda.Parameters.Wind.SWE.wnd_swe_kp.wnd_swe_vth,
    spz.inventories.tree.amda.Parameters.Wind.SWE.wnd_swe_kp.wnd_swe_pdyn,
    spz.inventories.tree.amda.Parameters.Wind.SWE.wnd_swe_kp.wnd_swe_n,
    spz.inventories.tree.cda.Wind.WIND.MFI.WI_H2_MFI.BGSE,
    spz.inventories.tree.ssc.Trajectories.wind,
]

data_several_dates: List[List[SpeasyVariable]] = spz.get_data(
    products,
    spz.inventories.tree.amda.TimeTables.SharedTimeTables.SOLAR_WIND.Magnetic_Clouds
)

for i in range(5):
    #fig = plt.figure(figsize=(20, 6))
    fig = plt.figure()
    gs = fig.add_gridspec(5, hspace=0)
    axes = gs.subplots(sharex=True, sharey=False)
    for j in range(5):
        data_several_dates[j][i].plot(ax=axes[j])
    plt.tight_layout()
plt.show()
```



Data discovery: an up to date hierarchical inventory

```
❯ jupyter-console
Jupyter console 6.6.3

Python 3.12.6 (main, Sep  9 2024, 00:00:00) [GCC 14.2.1 20240801 (Red Hat 14.2.1-1)]
Type 'copyright', 'credits' or 'license' for more information
IPython 8.25.0 -- An enhanced Interactive Python. Type '?' for help.

In [1]: import speasy as spz

In [2]: spz.get_data(spz.inventories.tree.
```

compatibility

Speasy variables implements “Numpy” hooks (Numpy’s Universal Functions)

```
def vect_to_mfa_delta(start, stop):
    B=spz.get_data(spz.inventories.tree.amda.Parameters.Cassini.MAG.orbit_saturn.cass_mag_krtphr.cass_b_krtphr,
                   start-timedelta(seconds=500),
                   stop+timedelta(seconds=500))
    B0=np.median(B, axis=0)
    delta_B=B-B0
    # REF = [er, etheta, ephi]
    matrix_REF_to_MFA=np.empty((3,3))
    matrix_REF_to_MFA[0][:]=B0/np.linalg.norm(B0)
    matrix_REF_to_MFA[1]=np.cross(matrix_REF_to_MFA[0],[1,0,0])
    matrix_REF_to_MFA[2]=np.cross(matrix_REF_to_MFA[0],matrix_REF_to_MFA[1])

    B_mfa = np.einsum('kj,ij->ik', matrix_REF_to_MFA, delta_B)

    B_mfa = sosfiltfilt( sos=sos,var=B_mfa)

    return B_mfa
```

Scipy compatibility

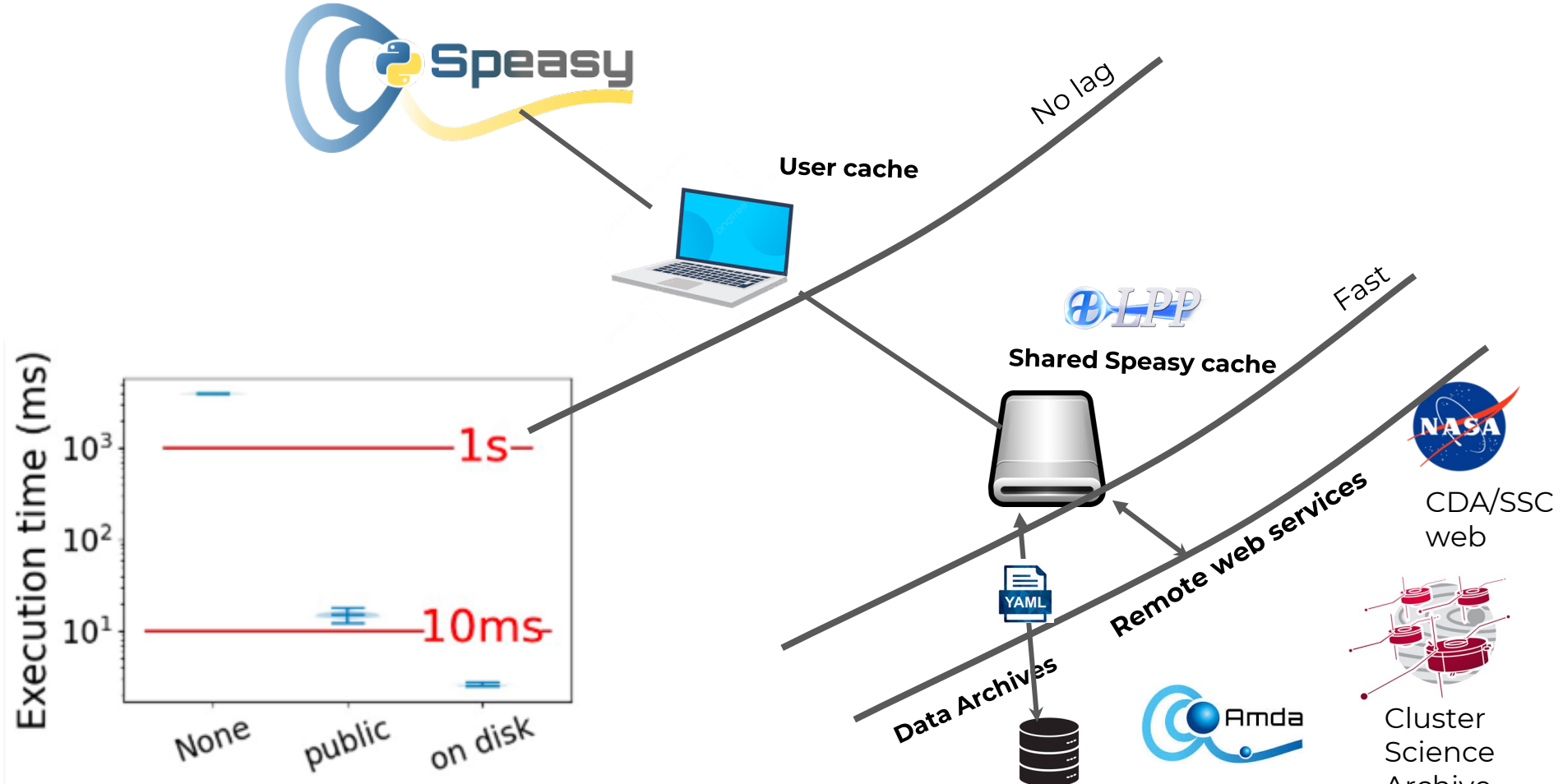
- Resampling
- Interpolating a list of variables onto a reference one
- Filtering signal

```
def mirror_mode_threshold(start_time: float, stop_time: float) -> SpeasyVariable or None:
    mms1_products = spz.inventories.data_tree.cda.MMS.MMS1
    products = [mms1_products.DIS.MMS1_FPI_FAST_L2_DIS_MOMS.mms1_dis_temppara_fast,
                mms1_products.DIS.MMS1_FPI_FAST_L2_DIS_MOMS.mms1_dis_tempperp_fast,
                mms1_products.FGM.MMS1_FGM_SRVY_L2.mms1_fgm_b_gse_srvy_l2,
                mms1_products.DIS.MMS1_FPI_FAST_L2_DIS_MOMS.mms1_dis_numberdensity_fast]

    tpara, tperp, b, n = spz.get_data(products, start_time, stop_time)

    anisotropy = tperp / tpara
    Pperp = tperp * n * 1e6
    b = interpolate(tperp, b)
    betaperp = Pperp * cst.mu_0 * cst.e * 2 / (b["Bt"] * 1e-9) ** 2
```

Super fast access to data

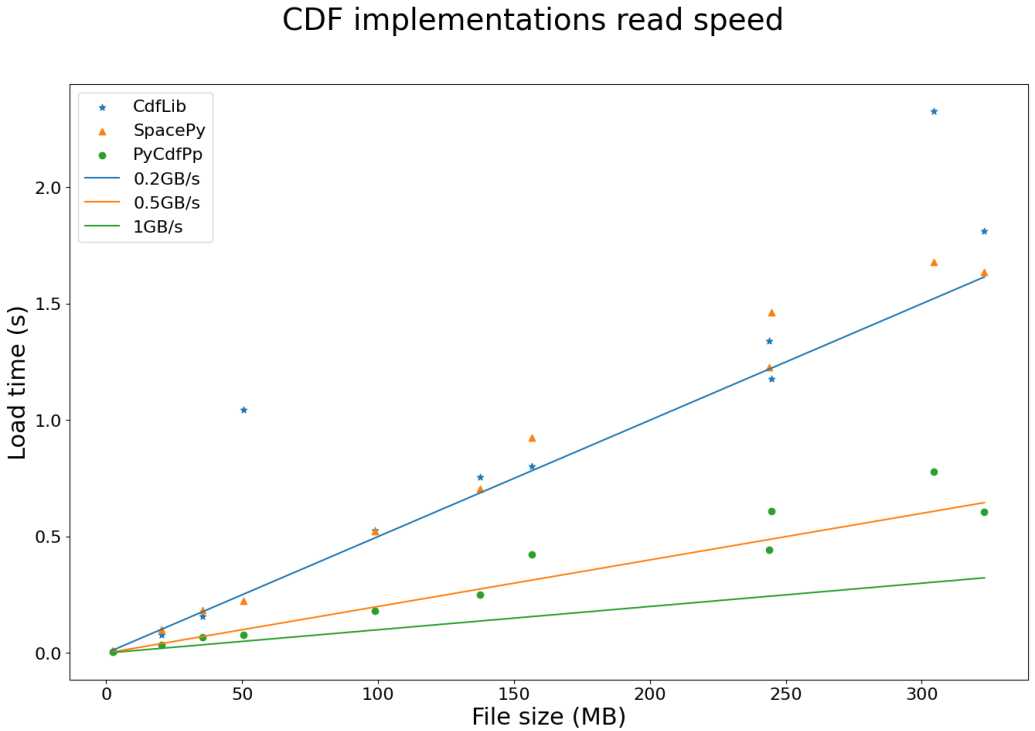


A custom CDF codec, CDF++/PyCDF++

Designed to solve those issues:

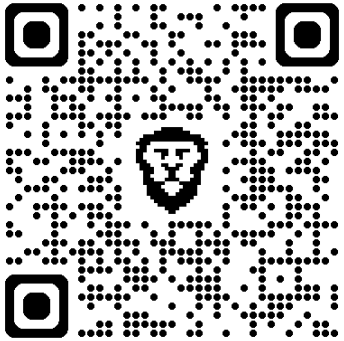
- Thread safety
- Performances up to Python
- Fast and exact time conversion
- Use of modern C++
- Decouple logic from file format
- In memory files
- GPL license
- Can be used as testbed for CDF evolution or compression standards

POC of WebAssembly build ;)



Access any local or remote archive

A “simple” YAML file to describe your archive:



```yaml
mms1_scm_srvy_l1b_scsrvy:
  inventory_path: mammouth/MMS/MMS1/scm/scsrvy
  master_cdf: >-
    /mammouth/data/mms1/scm/srvy/l1b/scsrvy/2017/01/mms1_scm_srvy_l1b_scsrvy_20170101_v1.1.0.cdf
  split_frequency: daily
  split_rule: regular
  url_pattern: >-
    /mammouth/data/mms1/scm/srvy/l1b/scsrvy/{Y}/{M:02d}/mms1_scm_srvy_l1b_scsrvy_{Y}{M:02d}{D:02d}_v\d+.\d+.\d+.cdf
  use_file_list: true

mms1_edp_brst_l2_hmfe:
  fname_regex: 'mms1_edp_brst_l2_hmfe_(?P<start>\d+)_v(?P<version>[\d\.]+)\.cdf'
  inventory_path: cda/MMS/MMS1/EDP/BURST
  master_cdf: >-
    https://cdaweb.gsfc.nasa.gov/pub/software/cdawlib/0MASTERS/mms1_edp_brst_l2_hmfe_00000000_v01.cdf
  split_frequency: monthly
  split_rule: random
  url_pattern: >-
    https://cdaweb.gsfc.nasa.gov/pub/data/mms/mms1/edp/brst/l2/hmfe/{Y}/{M:02d}/mms1_edp_brst_l2_hmfe_{Y}{M:02d}\d+_v\d+.\d+.\d+.cdf
  use_file_list: true
```

- Easy to use (**spz.get_data(***product*, *time range***)**)
- ~70k products “out of the box” from 4 different servers (more to come)
- Actively developed and maintained
  - HAPI client soon (I swear!)
  - Custom user codecs
- Already used for publications
- Plays well with Numpy and SciPy
- Will be used as AMDA gateway to CDAWeb