Problem Set 1

0.1 AE, Dst, Kp

Obtain and plot the AE [1min resolution] and Dst [1hr resolution] indices directly from their official site, WDC-C in Kyoto, and Kp [3hr resolution] index from its official site, GFZ Potsdam, the entire month of March 2015. Download ASCII and plot using your own tools which are not SPEDAS. Note it's easier to download in IAGA-2002 format for AE and Dst and to use the ftp directory download for kp.

0.1.a AE and Dst indices

[a] In a short paragraph explain: What is AE index, and Dst index, and how are they derived? How did you download and plot? Show some lines of the ascii data.

The AE index (Auroral Electrojet index) and Dst index (Disturbance Storm-Time index) are geomagnetic indices used to measure different aspects of Earth's magnetic activity. The AE index quantifies the intensity of auroral electrojets, currents flowing in the auroral regions, and is derived from horizontal magnetic field variations measured by magnetometers distributed around the auroral zone. It provides insight into substorm activity and is calculated using the difference between maximum and minimum horizontal components recorded at these stations. The Dst index, on the other hand, measures the intensity of the ring current, a system of charged particles trapped in Earth's magnetosphere. It is derived from the average horizontal magnetic field variations observed by low-latitude magnetometers. The Dst index is commonly used to assess geomagnetic storms, with negative values indicating stronger disturbances.

We use julia to download and plot the AE and Dst indices directly from their official site, WDC-C in Kyoto.

- https://wdc.kugi.kyoto-u.ac.jp/aeasy/index.html
- https://wdc.kugi.kyoto-u.ac.jp/dstae/index.html / http://wdc.kugi.kyoto-u.ac.jp/dst_final/201503

First we register and download the data dependencies using the `DataDeps` package.

```
using CairoMakie
using DataDeps
using CSV, DelimitedFiles
using TimeSeries
using DataFrames
using Dates
```

```
register(DataDep("AE Index",
    """
    Dataset: AE Index
    Website: https://wdc.kugi.kyoto-u.ac.jp/aeasy/index.html
    """,
    "https://wdc.kugi.kyoto-u.ac.jp///aeasy///wwwtmp/WWW_aeasy04087001.dat"
))

register(DataDep("Dst Index",
    """
    Dataset: Dst Index
    Website: https://wdc.kugi.kyoto-u.ac.jp/dstae/index.html
    """,
    "https://wdc.kugi.kyoto-u.ac.jp///dstae///wwwtmp/WWW_dstae04096902.dat"
))

# Step 1: Download data
println("Downloading AE index data...")
ae_file = readdir(datadep"AE Index", join=true)[1]

println("Downloading Dst index data...")
dst_file = readdir(datadep"Dst Index", join=true)[1]
```

```
┌ Warning: Over-writing registration of the datadep
│   name = "AE Index"
└ @ DataDeps ~/.julia/packages/DataDeps/Y2lje/src/registration.jl:15
┌ Warning: Over-writing registration of the datadep
│   name = "Dst Index"
└ @ DataDeps ~/.julia/packages/DataDeps/Y2lje/src/registration.jl:15
Downloading AE index data...
Downloading Dst index data...
```

```
"/Users/zijin/.julia/scratchspaces/124859b0-ceae-595e-8997-d05f6a7a8dfe/
datadeps/Dst Index/WWW_dstae04096902.dat"
```

The data is downloaded as text files in IAGA-2002 format. Sample data looks like this:

```
{txt}
DATE       TIME         DOY     AE        AU        AL        AO     |
2015-03-01 00:00:00.000 060       245.00    122.00   -123.00     -1.00
2015-03-01 00:01:00.000 060       253.00    125.00   -128.00     -2.00
2015-03-01 00:02:00.000 060       235.00    122.00   -113.00      5.00
2015-03-01 00:03:00.000 060       225.00    120.00   -105.00      8.00
```

```
{txt}
DATE       TIME         DOY     DST                                    |
2015-03-01 00:00:00.000 060       -34.00
2015-03-01 01:00:00.000 060       -44.00
2015-03-01 02:00:00.000 060       -30.00
2015-03-01 03:00:00.000 060       -31.00
```

We then parse the data into TimeSeries using `DelimitedFiles` and `TimeSeries`.

```
start_time = "2015-03-01T00:00:00"
end_time = "2015-03-31T23:59:59"

# Step 2: Parse AE data
function load_ae_data(filename)
    meta = Dict(
        "unit" => "nT",
        "label" => "AE Index"
    )

    # Skip the header lines (first 14 lines)
    data, header = readdlm(filename, skipstart=14, header=true)

    # Convert date and time columns to DateTime
    datetime = DateTime.(data[:, 1] .* "T" .* data[:, 2])

    TimeArray(datetime, data[:, 4:7], header[4:7], meta)
end

function load_dst_data(filename)
    meta = Dict(
        "unit" => "nT",
        "label" => "Dst Index"
    )

    # Skip the header lines (first 17 lines)
    data, header = readdlm(filename, skipstart=17, header=true)

    # Convert date and time columns to DateTime
    datetime = DateTime.(data[:, 1] .* "T" .* data[:, 2])

    TimeArray(datetime, data[:, 4:4], header[4:4], meta)
end

println("Parsing AE data...")
ae_data = load_ae_data(ae_file)
```

```
println("Parsing Dst data...")
dst_data = load_dst_data(dst_file)
```

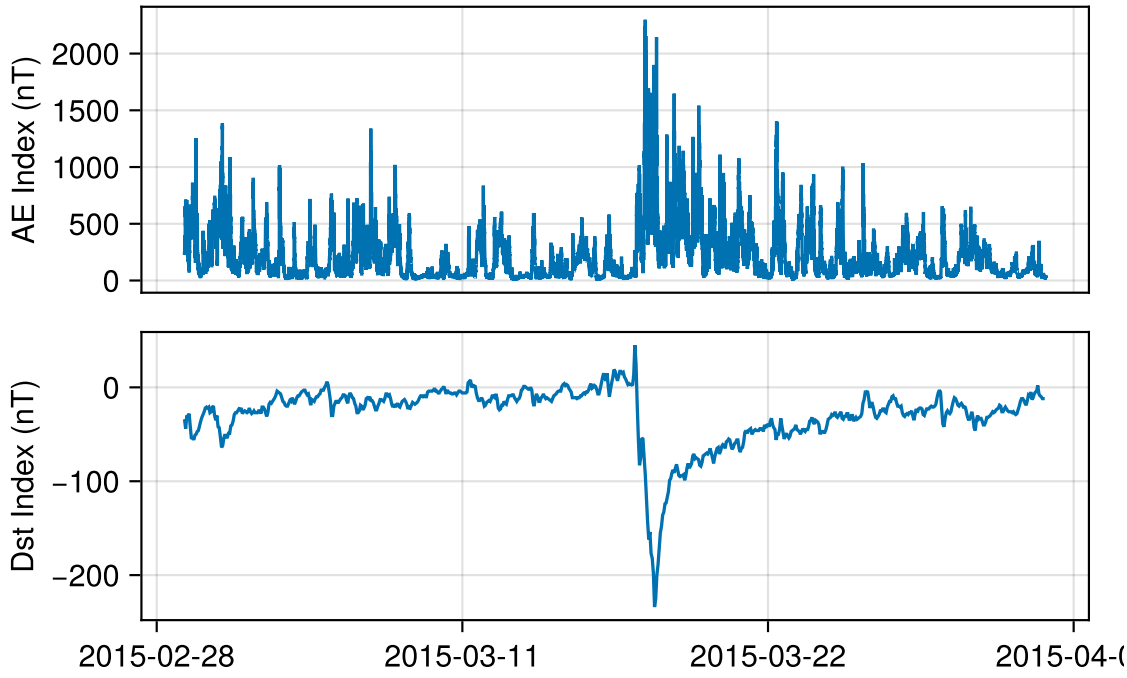
```
Parsing AE data...
Parsing Dst data...
```

```
744×1     TimeSeries.TimeArray{Any,     2,     Dates.DateTime,     Matrix{Any}}
2015-03-01T00:00:00 to 2015-03-31T23:00:00
┌─────────────────────┬────────┐
│                     │ DST    │
├─────────────────────┼────────┤
│ 2015-03-01T00:00:00 │ -34.0  │
│ 2015-03-01T01:00:00 │ -44.0  │
│ 2015-03-01T02:00:00 │ -30.0  │
│ 2015-03-01T03:00:00 │ -31.0  │
│ 2015-03-01T04:00:00 │ -28.0  │
│ 2015-03-01T05:00:00 │ -42.0  │
│ 2015-03-01T06:00:00 │ -55.0  │
│ 2015-03-01T07:00:00 │ -53.0  │
│          ⋮          │   ⋮    │
│ 2015-03-31T17:00:00 │   2.0  │
│ 2015-03-31T18:00:00 │  -7.0  │
│ 2015-03-31T19:00:00 │  -9.0  │
│ 2015-03-31T20:00:00 │ -10.0  │
│ 2015-03-31T21:00:00 │ -12.0  │
│ 2015-03-31T22:00:00 │ -12.0  │
│ 2015-03-31T23:00:00 │ -12.0  │
└─────────────────────┴────────┘
               729 rows omitted
```

Plot the data using CairoMakie.

```
Makie.convert_arguments(P::Type{<:Lines}, ta::TimeArray) = convert_arguments(P,
timestamp(ta), values(ta))

ylabel(ta::TimeArray) = meta(ta)["label"] * (isnothing(meta(ta)["unit"]) ||
meta(ta)["unit"] == "" ? "" : " (" * meta(ta)["unit"] * ")")

function tplot(tas; linkxaxes=true, figure=(;), kwargs...)
    f = Figure(; figure...)
    axs = []
    for (i, ta) in enumerate(tas)
        ax = Axis(f[i, 1]; ylabel=ylabel(ta))

        for p in propertynames(ta)
            lines!(ax, getproperty(ta, p); label=p)
```

```
        end

        # Hide redundant x labels
        linkxaxes && i != length(tas) && hidexdecorations!(ax, grid=false)
        push!(axs, ax)
    end
    linkxaxes && linkxaxes!(axs...)
    f
end

f = tplot([ae_data.AE, dst_data])
```



0.1.b Kp index

> In a second paragraph do the same as above but for Kp. What is Kp index? And how is it derived?

The Kp index is a global geomagnetic activity index that quantifies disturbances in Earth's magnetic field caused by solar activity. Kp is calculated from the K values or the geomagnetic recordings of 13 mid-latitude geomagnetic observatories.

We use a web service client to load the data (https://kp.gfz-potsdam.de/en/data) from json format into the TimeArray.

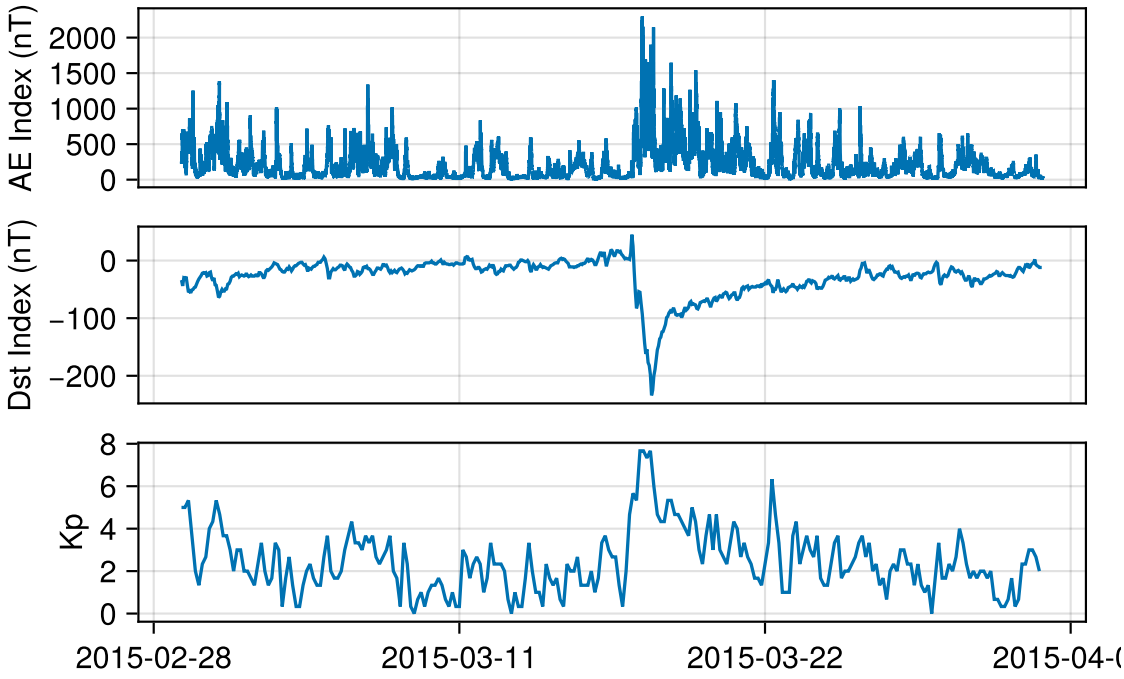
```
using JSON3

"""
Download 'Kp', 'ap', 'Ap', 'Cp', 'C9', 'Hp30', 'Hp60', 'ap30', 'ap60', 'SN',
'Fobs' or 'Fadj' index data from kp.gfz-potsdam.de

# Notes
- 'start_time' and 'end_time': date format 'YYYY-MM-DDThh:mm:ss'
"""
function load_Kp_data(start_time, end_time, index=:Kp)
    url = "https://kp.gfz-potsdam.de/app/json/?start=$(start_time)Z&end=$(end_
time)Z&index=$(index)"
    data = JSON3.read(download(url))
    datetime = DateTime.(data[:datetime], "yyyy-mm-ddTHH:MM:SSZ")
      TimeArray(datetime, Float64.(data[index]), [index], Dict("unit" => "",
"label" => string(index)))
end

kp_data = load_Kp_data(start_time, end_time)
```

```
248×1   TimeSeries.TimeArray{Float64,   1,   Dates.DateTime,   Vector{Float64}}
2015-03-01T00:00:00 to 2015-03-31T21:00:00
┌─────────────────────┬───────┐
│                     │ Kp    │
├─────────────────────┼───────┤
│ 2015-03-01T00:00:00 │   5.0 │
│ 2015-03-01T03:00:00 │   5.0 │
│ 2015-03-01T06:00:00 │ 5.333 │
│ 2015-03-01T09:00:00 │ 3.667 │
│ 2015-03-01T12:00:00 │   2.0 │
│ 2015-03-01T15:00:00 │ 1.333 │
│ 2015-03-01T18:00:00 │ 2.333 │
│ 2015-03-01T21:00:00 │ 2.667 │
│          ⋮          │   ⋮   │
│ 2015-03-31T03:00:00 │ 0.667 │
│ 2015-03-31T06:00:00 │ 2.333 │
│ 2015-03-31T09:00:00 │ 2.333 │
│ 2015-03-31T12:00:00 │   3.0 │
│ 2015-03-31T15:00:00 │   3.0 │
│ 2015-03-31T18:00:00 │ 2.667 │
│ 2015-03-31T21:00:00 │   2.0 │
└─────────────────────┴───────┘
               233 rows omitted
```

0.1.c Plot

[c] Plot all 3 indices in a single plot with common time, UT in usual units [2015-03-01 00:00]. If you cannot plot in this format, simply plot in decimal day or decimal hour.

```
f = tplot([ae_data.AE, dst_data, kp_data])
```



0.2 Galileo Magnetometer PDS

Obtain and plot the Galileo magnetometer flyby #8 data of Ganymede from the Planetary Data System (PDS). This is the NASA data repository of Planetary missions, the counterpart of SPDF (Space Physics Data Facility) used for Heliophysics missions. Note that planetary magnetospheric quantities are stored in both. UCLA is the central "node" for PDS with Prof. Walker the PI of that program. Use your own tools such as Excel (but not SPEDAS) to do this (you are welcome to use SPLASH if on Windows (download it from PDS)).

0.2.a Coordinate system

In a short paragraph explain: What is the coordinate system and how did you download and plot?

Dataset about Galileo (https://pds-ppi.igpp.ucla.edu/mission/Galileo) flyby of Ganymede is available in the Planetary Data System (PDS) https://pds-ppi.igpp.ucla.edu/collection/urn:nasa:

pds:galileo-mag-jup-calibrated:data-highres-ganymede [1]. Galileo calibrated MAG high-resolution magnetic field and trajectory data from the Ganymede 8 orbit Ganymede flyby cover 1997-05-07T15:36:00 to 1997-05-07T16:22:00.

The dataset contains data in the following coordinate systems:

- Ganymede Phi-Omega (GPHIO) coordinates https://pds-ppi.igpp.ucla.edu/item/urn:nasa:pds:galileo-mag-jup-calibrated:data-highres-ganymede:orb08_gan_gphio::1.0 [2].
- Ganymede-centered 'planetocentric' right-handed (GSPRH) coordinates https://pds-ppi.igpp.ucla.edu/item/urn:nasa:pds:galileo-mag-jup-calibrated:data-highres-ganymede:orb08_gan_gsprh::1.0
- Despun Spacecraft (IRC) coordinates https://pds-ppi.igpp.ucla.edu/item/urn:nasa:pds:galileo-mag-jup-calibrated:data-highres-ganymede:orb08_gan_irc::1.0
- System III [1965] (SYS3) coordinates https://pds-ppi.igpp.ucla.edu/item/urn:nasa:pds:galileo-mag-jup-calibrated:data-highres-ganymede:orb08_gan_sys3::1.0

The data is downloaded as CSV format, read and plot using Julia.

```
register(DataDep("ORB08_GAN_GPHIO",
    """
    Dataset: ORB08_GAN_GPHIO
    Website: https://pds-ppi.igpp.ucla.edu/mission/Galileo
    """,
      "https://pds-ppi.igpp.ucla.edu/ditdos/write?id=urn:nasa:pds:galileo-mag-
jup-calibrated:data-highres-ganymede:orb08_gan_gphio::1.0&f=csv"
))

GAN_file = readdir(datadep"ORB08_GAN_GPHIO", join=true)[1]

data = CSV.File(GAN_file) |> DataFrame
```

```
┌ Warning: Over-writing registration of the datadep
│   name = "ORB08_GAN_GPHIO"
└ @ DataDeps ~/.julia/packages/DataDeps/Y2lje/src/registration.jl:15
```

```
8187×8 DataFrame
  Row │ SPACECRAFT EVENT TIME    BX       BY       BZ       B-FIELD MAGNITUDE  ⋯
      │ Dates.DateTime           Float64  Float64  Float64  Float64            ⋯
──────┼──────────────────────────────────────────────────────────────────────────
    1 │ 1997-05-07T15:36:55.133    -8.36   -25.04   -85.24             89.23  ⋯
    2 │ 1997-05-07T15:36:55.467    -8.38   -25.16   -85.22              89.25
    3 │ 1997-05-07T15:36:55.800    -8.41   -25.09   -85.24              89.25
    4 │ 1997-05-07T15:36:56.133    -8.44   -25.08   -85.27              89.28
    5 │ 1997-05-07T15:36:56.467    -8.5    -25.16   -85.19             89.23  ⋯
    6 │ 1997-05-07T15:36:56.800    -8.47   -25.18   -85.18              89.23
    7 │ 1997-05-07T15:36:57.133    -8.6    -25.2    -85.18              89.25
```

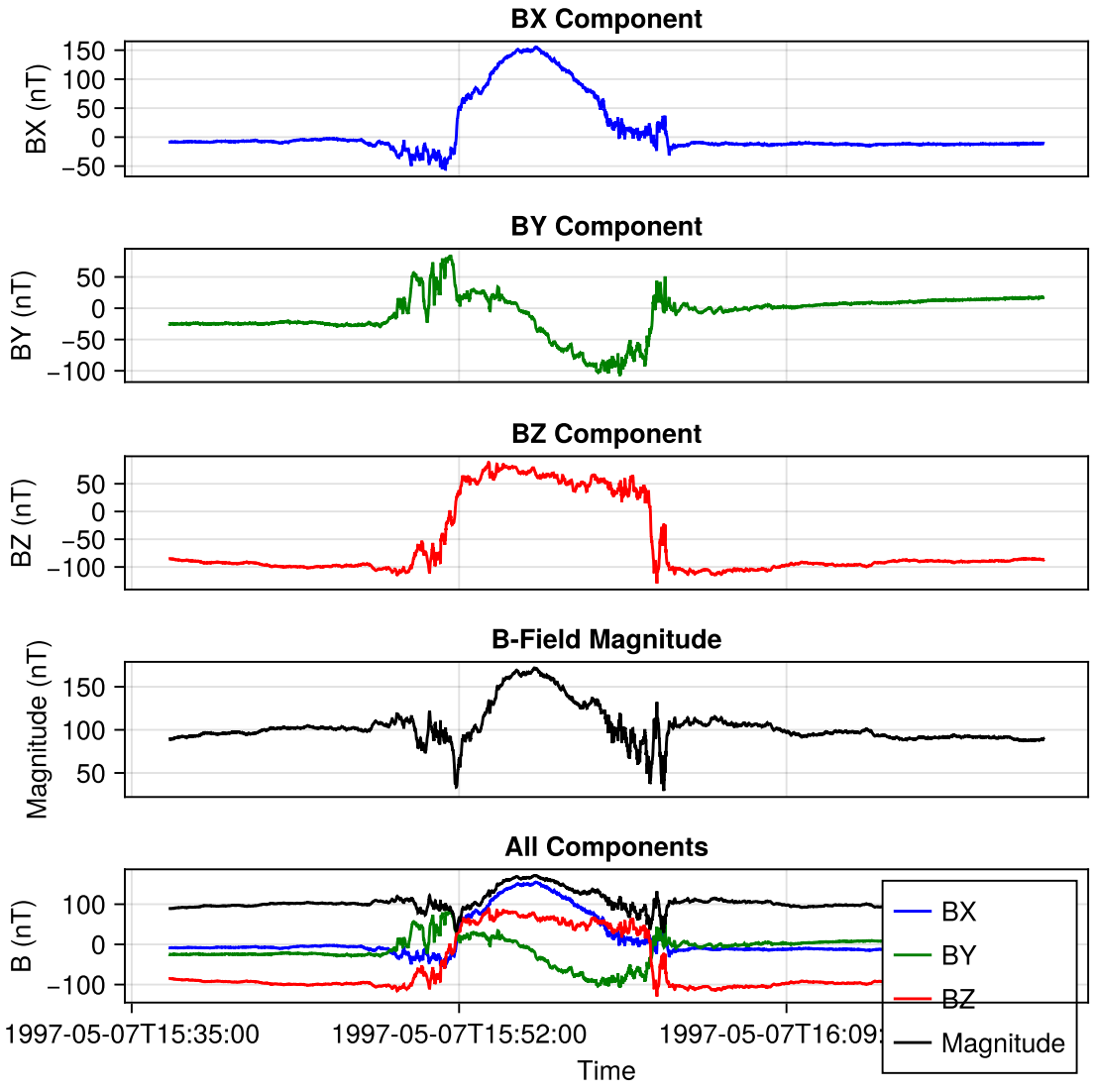
```
    8 | 1997-05-07T15:36:57.467    -8.47   -25.04   -85.12              89.13
  ⋮   |            ⋮                ⋮        ⋮        ⋮             ⋮          ⋮
 8181 | 1997-05-07T16:22:21.799   -10.97    17.63   -86.44             88.9   …
 8182 | 1997-05-07T16:22:22.132   -11.01    17.53   -86.46              88.9
 8183 | 1997-05-07T16:22:22.465   -10.98    17.45   -86.38              88.81
 8184 | 1997-05-07T16:22:22.799   -11.05    17.35   -86.41              88.82
 8185 | 1997-05-07T16:22:23.132   -11.07    17.39   -86.42               88.84
…
 8186 | 1997-05-07T16:22:23.465   -11.17    17.37   -86.27              88.71
 8187 | 1997-05-07T16:22:23.465   -11.17    17.37   -86.27              88.71
                                                  3 columns and 8172 rows omitted
```

0.2.b Plot

Plot XYZ components and magnitude of the field in four panels. Use a 5th panel in the same plot to show all four traces with different colors, say: Blue, Green, Red, Black for X,Y,Z,T.

```
# Create the plot with multiple panels
fig = Figure(; size=(600, 600))

time = data."SPACECRAFT EVENT TIME"

# Panel for BX
ax1 = Axis(fig[1, 1], title="BX Component", xlabel="Time", ylabel="BX (nT)")
lines!(ax1, time, data[!, "BX"], color=:blue)

# Panel for BY
ax2 = Axis(fig[2, 1], title="BY Component", xlabel="Time", ylabel="BY (nT)")
lines!(ax2, time, data[!, "BY"], color=:green)

# Panel for BZ
ax3 = Axis(fig[3, 1], title="BZ Component", xlabel="Time", ylabel="BZ (nT)")
lines!(ax3, time, data[!, "BZ"], color=:red)

# Panel for B-FIELD MAGNITUDE
ax4    =    Axis(fig[4,    1],    title="B-Field    Magnitude",    xlabel="Time",
ylabel="Magnitude (nT)")
lines!(ax4, time, data[!, "B-FIELD MAGNITUDE"], color=:black)

# Panel for all traces
ax5 = Axis(fig[5, 1], title="All Components", xlabel="Time", ylabel="B (nT)")
lines!(ax5, time, data[!, "BX"], color=:blue, label="BX")
lines!(ax5, time, data[!, "BY"], color=:green, label="BY")
lines!(ax5, time, data[!, "BZ"], color=:red, label="BZ")
lines!(ax5, time, data[!, "B-FIELD MAGNITUDE"], color=:black, label="Magnitude")
```

```
for ax in [ax1, ax2, ax3, ax4]
    hidexdecorations!(ax, grid=false)
end

linkxaxes!(ax1, ax2, ax3, ax4, ax5)
axislegend(ax5)

fig
```



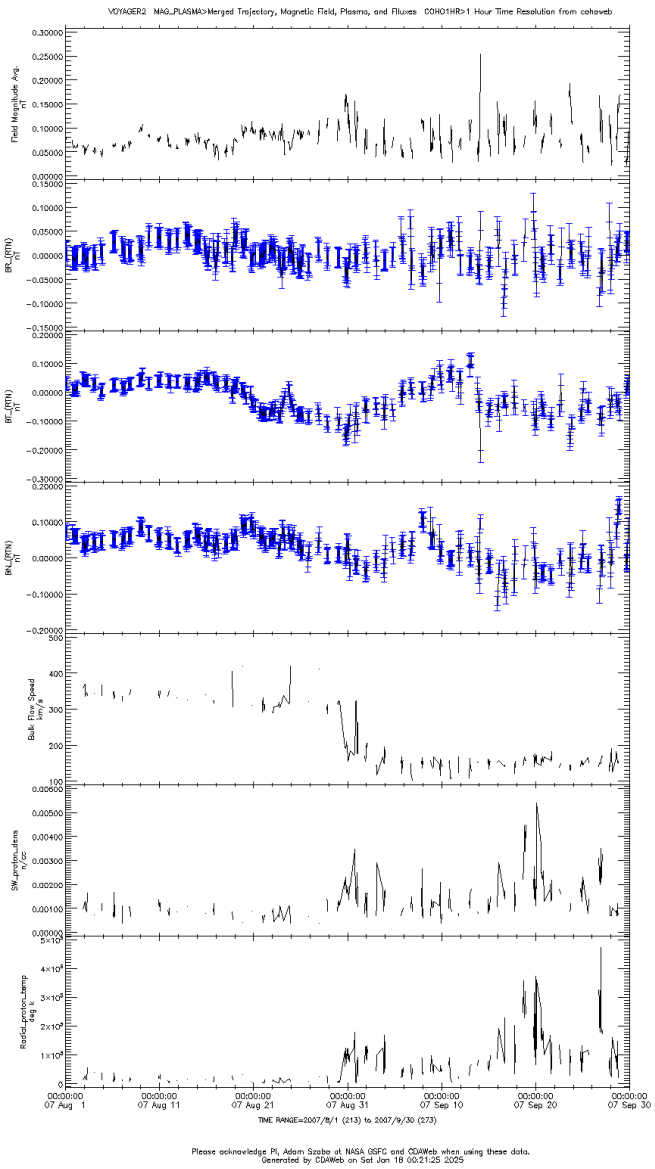
0.3 Voyager 2 Heliospheric termination shock

Voyager 2 crossed the Heliospheric termination shock and entered the heliosheath in Aug. 2007.

0.3.a Plot and analysis

[a] Plot the shock crossing in CDAWeb using their own plotting resources. In particular plot the total and RTN components of the magnetic field, flow speed, proton density N, and temperature T. In a short paragraph explain: what is the RTN system and how did you create the plot? In a second paragraph explain if the changes in the above parameters are consistent with shock crossings.

The RTN (Radial-Tangential-Normal) coordinate system is a heliocentric coordinate system used to describe the position in relation to the Sun fixed at a spacecraft (or the planet): The R axis is directed radially away from the Sun, the T axis is the cross product of the solar rotation axis and the R axis, and the N axis is the cross product of R and T.



We utilize the combined COHOWeb dataset (VOYAGER2_COHO1HR_MERGED_MAG_-PLASMA, https://cdaweb.gsfc.nasa.gov/misc/NotesV.html#VOYAGER2_COHO1HR_MERGED_MAG_PLASMA) [3]. The plot was generated using "CDAWeb Data Explorer" by selecting the "Plot Data" option and choosing the appropriate data variable. We select time range from 2007/08/01 00:00:00.000 to 2007/09/30 00:00:00.000.

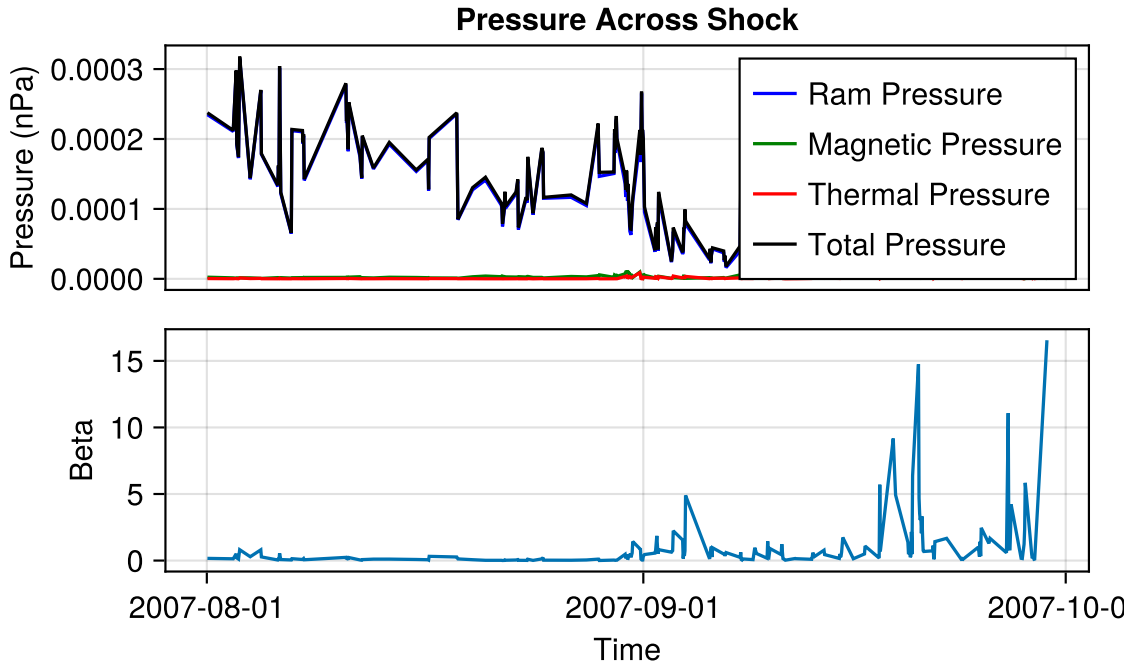
Around the end of August 31, we observe a significant decrease in bulk flow velocity accompanied by increases in proton density and temperature, corresponding to the shock crossing. Additionally, the magnetic field exhibits a slight increase. These changes are consistent with shock crossings, as shocks typically cause reductions in downstream flow velocity and simultaneous enhancements in plasma density, temperature, and magnetic field strength.

[b] What is the change in pressure, including dynamic pressure, across the shock (assume the crossing is along the shock normal)? To do this also ignore the electron thermal pressure (just use ion thermal pressure) and plot the quantity: N*V^2 + Pt, where Pt is the sum of magnetic and thermal pressures. Do this using your own tools (not SPEDAS) after downloading the data from SPDF.

```
using Unitful
using Unitful: μ0, k, mp

register(DataDep("VOYAGER2_COHO1HR_MERGED_MAG_PLASMA",
    """
    Dataset: VOYAGER2_COHO1HR_MERGED_MAG_PLASMA
      Website: https://cdaweb.gsfc.nasa.gov/misc/NotesV.html#VOYAGER2_COHO1HR_
MERGED_MAG_PLASMA
    """,
    "https://cdaweb.gsfc.nasa.gov/tmp/cweb_B6vad1PhkF/VOYAGER2_COHO1HR_MERGED_
MAG_PLASMA_1663093.csv"
))

# Read the CSV file
VIM_file = readdir(datadep"VOYAGER2_COHO1HR_MERGED_MAG_PLASMA", join=true)[1]
data = CSV.File(VIM_file; comment="#") |> DataFrame

allowmissing!(data)
data[!, "time"] = DateTime.(data."EPOCH_yyyy-mm-ddThh:mm:ss.sssZ", "yyyy-mm-
ddTHH:MM:SS.sssZ")
for c ∈ eachcol(data)
    replace!(c, -1.0e31 => missing)
end
dropmissing!(data)

N = data[!, "SW_PROTON_DENS_n/cc"] * u"cm^-3"
V = data[!, "BULK_FLOW_SPEED_km/s"] * u"km/s"
B = data[!, "FIELD_MAGNITUDE_AVG._nT"] * u"nT"
T = data[!, "RADIAL_PROTON_TEMP_deg_k"] * u"K"

# Calculate the pressure
pB = B .^ 2 / μ0 / 2 ./ 1u"nPa" .|> NoUnits
pT = k * N .* T ./ u"nPa" .|> NoUnits
pRam = mp * N .* V .^ 2 ./ u"nPa" .|> NoUnits
pressure = pRam .+ pB .+ pT
beta = pT ./ pB
time = data."time"

# Plot the data
fig = Figure()
ax = Axis(fig[1, 1], title="Pressure Across Shock", xlabel="Time",
```

```
ylabel="Pressure (nPa)")
lines!(time, pRam, color=:blue, label="Ram Pressure")
lines!(time, pB, color=:green, label="Magnetic Pressure")
lines!(time, pT, color=:red, label="Thermal Pressure")
lines!(time, pressure, color=:black, label="Total Pressure")
axislegend(ax)

ax2 = Axis(fig[2, 1], xlabel="Time", ylabel="Beta")
lines!(time, beta)

hidexdecorations!(ax, grid=false)
fig
```

```
┌ Warning: Over-writing registration of the datadep
│   name = "VOYAGER2_COHO1HR_MERGED_MAG_PLASMA"
└ @ DataDeps ~/.julia/packages/DataDeps/Y2lje/src/registration.jl:15
```



Bibliography

[1] K. Kivelson M.G., "Galileo Jupiter Magnetometer Highres Ganymede Data Collection." NASA Planetary Data System, 2024. doi: 10.17189/gch4-8w75.

[2] K. Kivelson M.G., "Galileo Jupiter MAG Highres Ganymede Data 1997-05-07." 2022.

[3] N. F. Ness, J. D. Richardson, L. F. Burlaga, and N. E. Papitashvili, “Voyager 2 Hourly Merged Magnetic Field and Plasma Data.” NASA Space Physics Data Facility, 2023. doi: 10.48322/VE5T-HF15.