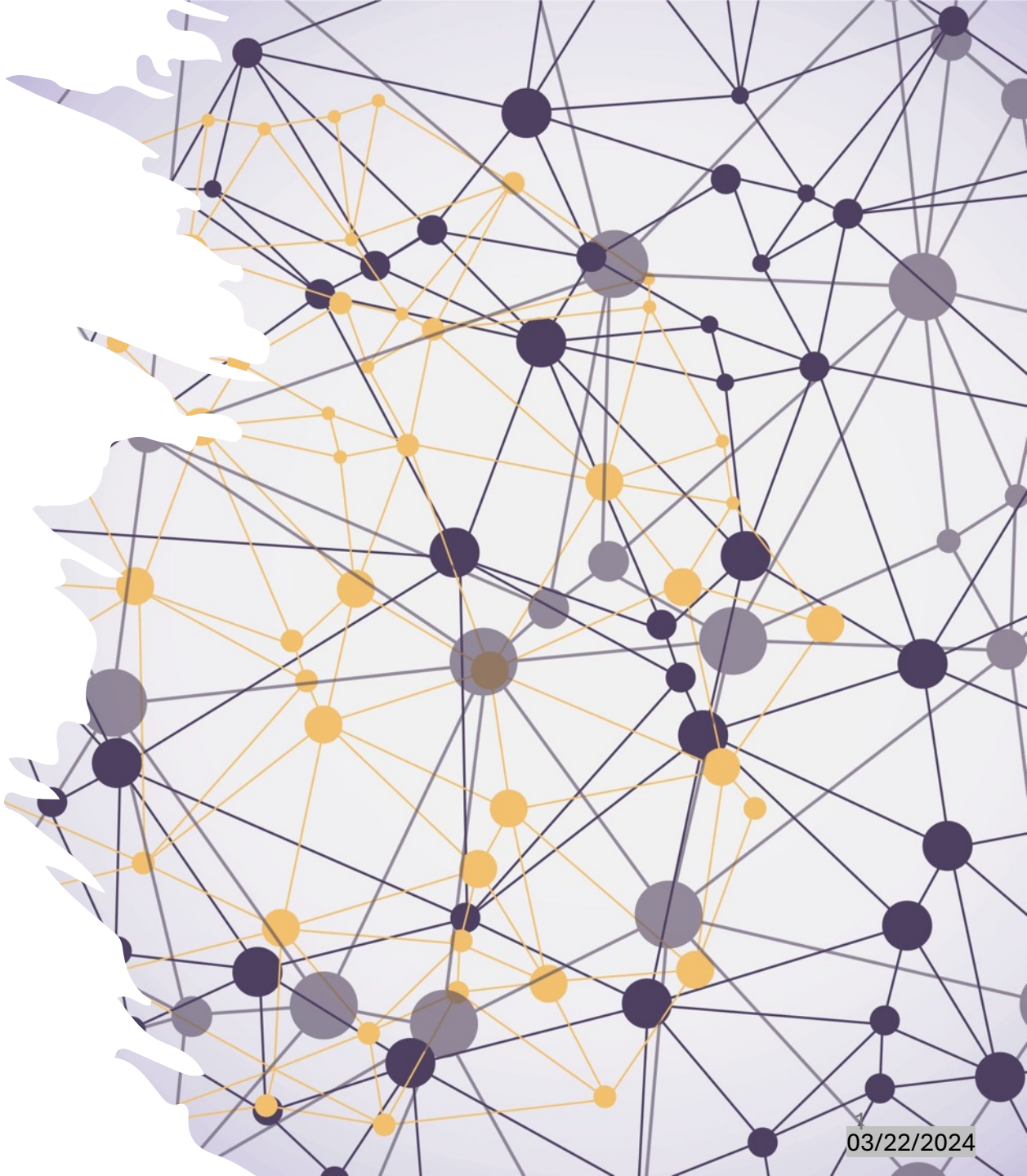
Working with High Frequency Wearables Data



Shravan Aras, Ph.D., University of Arizona

Center for Biomedical Informatics & Biostatistics

CYVERSE Care Evolution™ Center for Biomedical Informatics & Biostatistics

Team Acknowledgements

- **Yuan Jea Hew**
- School of Information
- Research : AI/ML pipelines

- **Gunjal Parekh**
- Management Information Systems

Data Acknowledgments

- Personalized Treatment Lab
  - Department of Psychology

- Dr. Zachary Cohen
- Nate Choukas
- Torsa Chattoraj

- Have generously donated the data for building this system

Outline

What is high frequency wearable data

- Raw sensor values
  - Accelerometer, Gyroscope, Lux meter, PPG optical signal,Skin Temperature .....
- Generally, 50-100Hz of sampling rate
- Millions of points per metric per day for a participant
- Lacks aggregate metrics
- Devices → Apple Sensorkit, Empatica Embrace, Ultra Human, Oura.

Where can it be used?

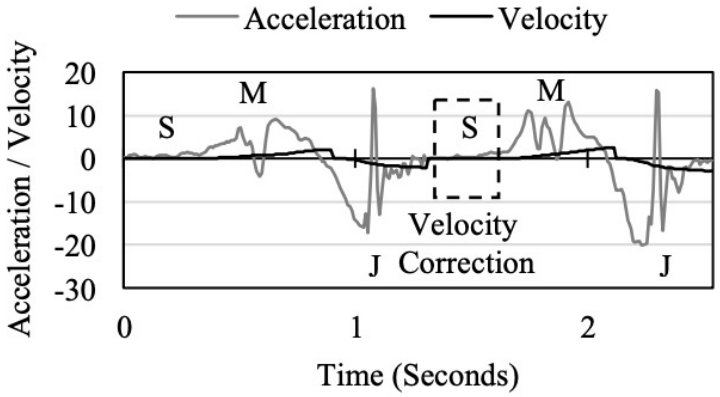


Figure 3. Acceleration and Velocity graph of 2 steps. S : Stationary, M : Start of step, J : Foot hits the ground and step ends.

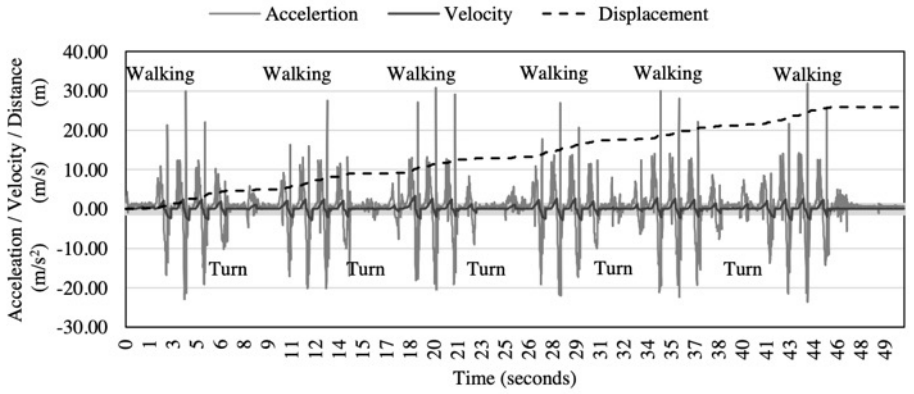


Figure 4. Automatic distance calculation using acceleration data from the ankle mounted accelerometer. (True distance = 25.56m, Calculated distance = 25.86m

Figure 1. Normalized daily finger temperature sample from preconception to delivery

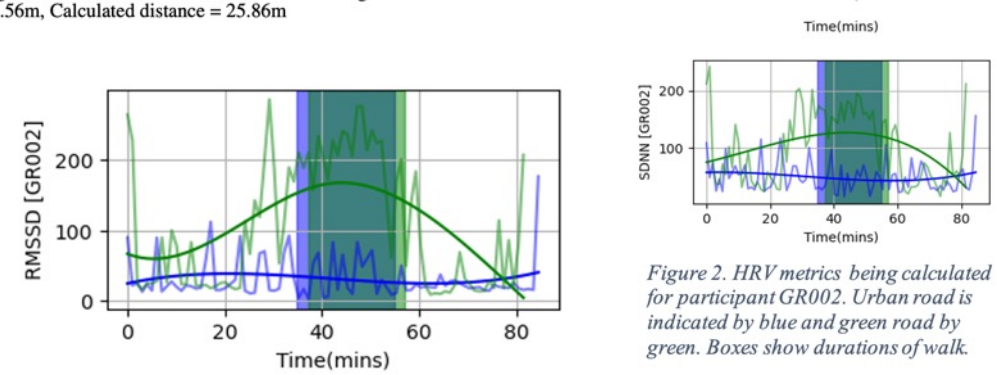


Figure 2. HRV metrics being calculated for participant GR002. Urban road is indicated by blue and green road by green. Boxes show durations of walk.

Basavaraj, C., Grant, A. D., Aras, S. G., & Erickson, E. N. (2024). Deep Learning Model Using Continuous Skin Temperature Data Predicts Labor Onset. *medRxiv*, 2024-02.

Gait Analysis, Fall Prediction, Jitter, Social Interactions,...

Aras, Shravan, et al. "MCATSS-End-to-End Mobile Cardiopulmonary Tolerance Score System." *Annual Conference of the PHM Society*. Vol. 11. No. 1. 2019.

Certain challenges to look out for

- Most of the data can be noise
- Can be overwhelming and have no idea where to start
  - Traditional potting tools fail to work
  - CSV won’t load in memory
  - Can’t open data set in excel to just “go through” it
- Sharing data and giving access is borderline insane
- Breaks and errors in data are hard to detect
- Monitoring participant compliance becomes guess work

What ends up happening

Collect till we are full → Then it just sits there

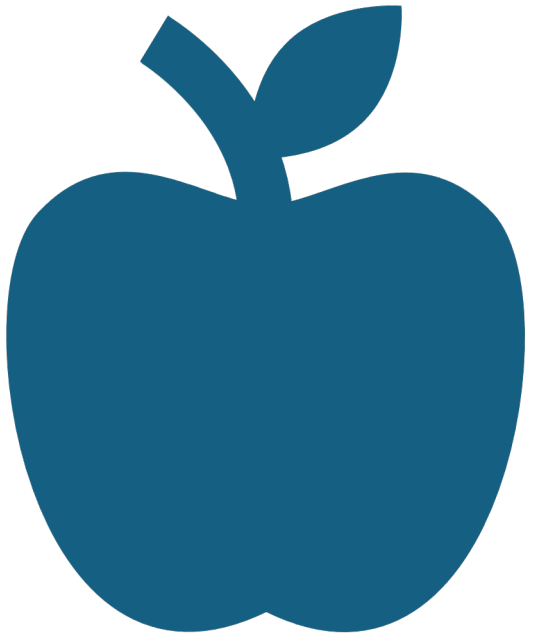
Data does NOT age like fine wine, more like an opened bottle of wine

Our Goal

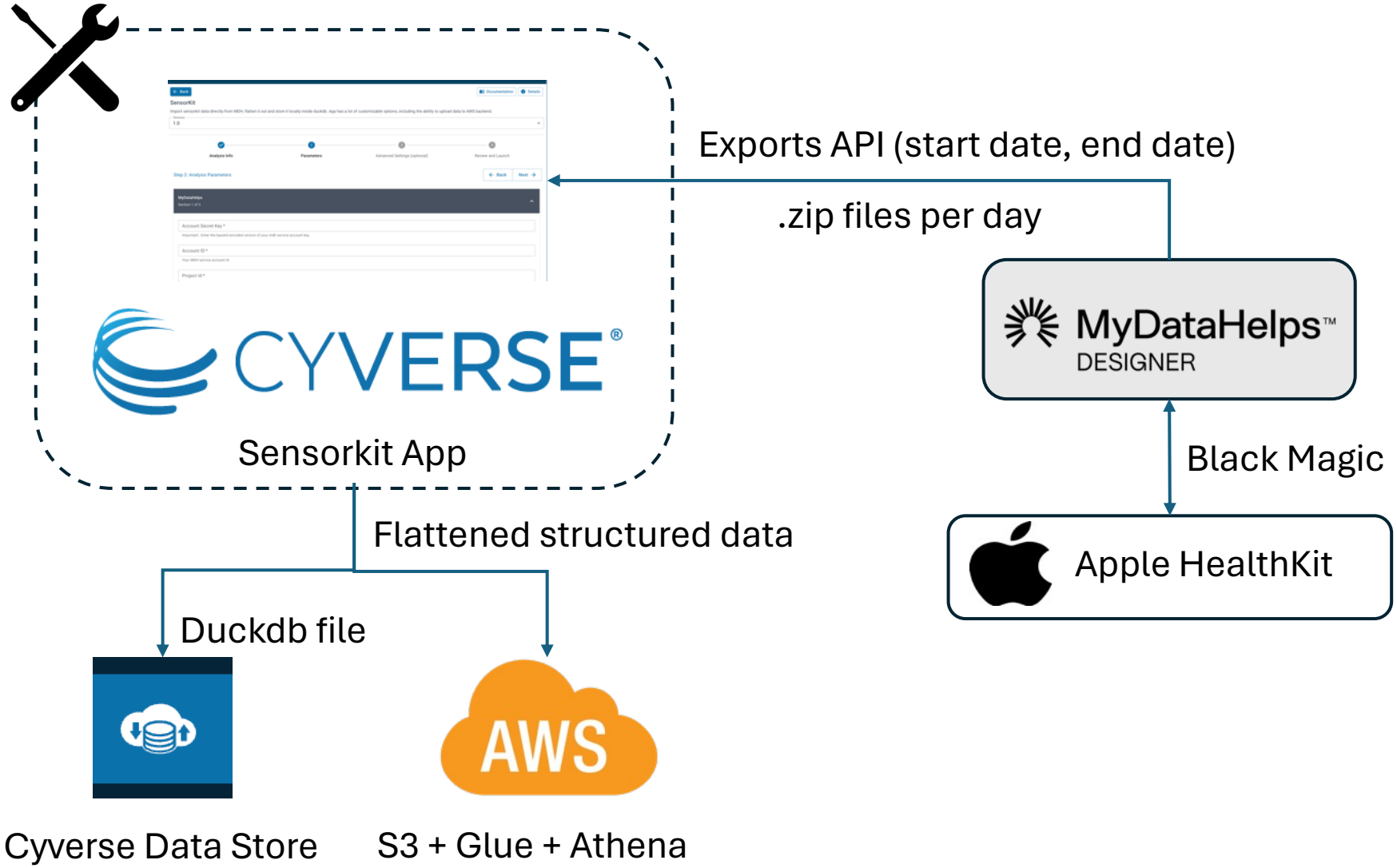
Make **querying, visualization, analysis** of high frequency data IN **appropriate time** using various data storage and ingestion practices TO **provide actionable, accurate and meaningful insights into it**.

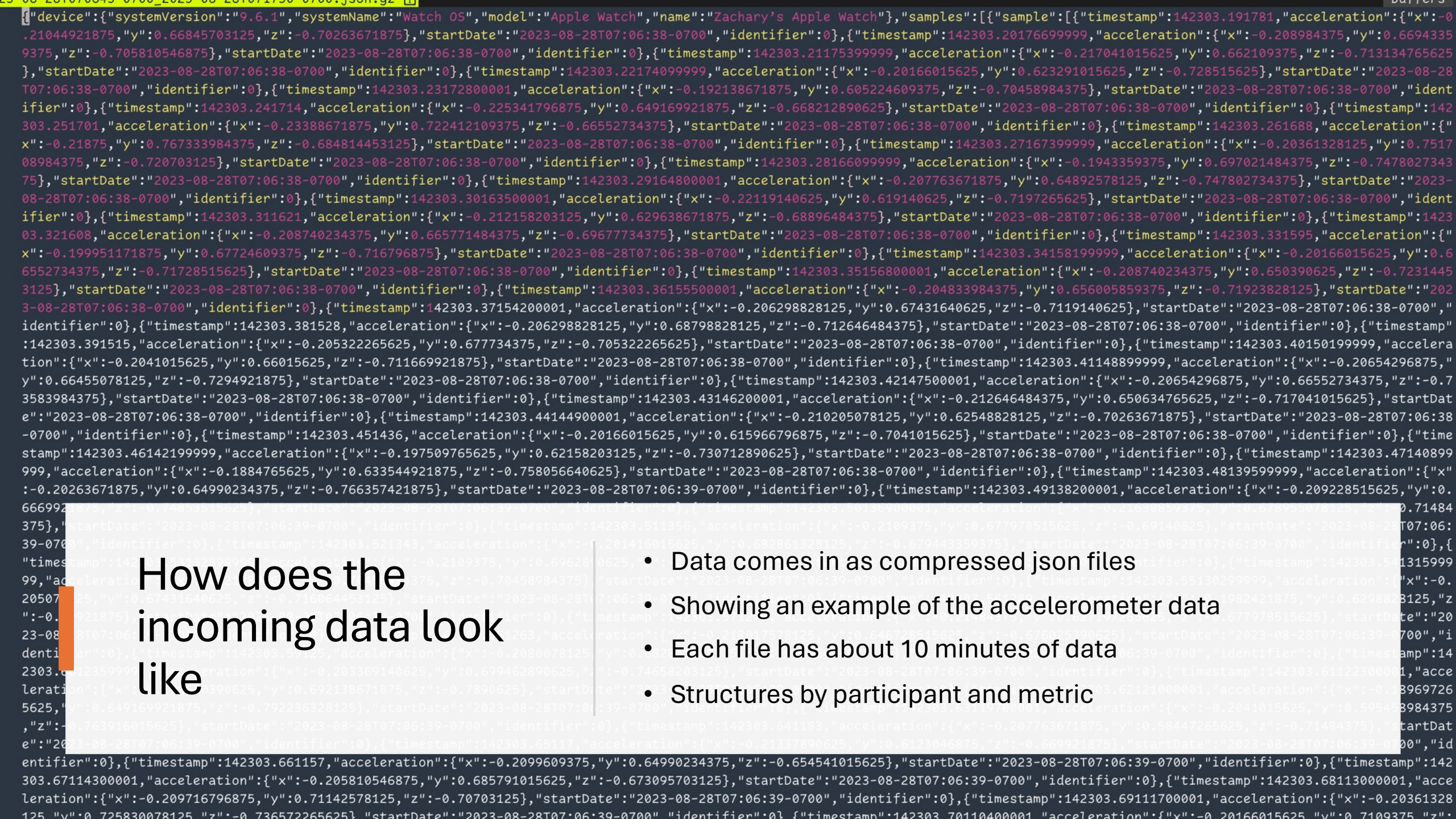
We will focus on Sensorkit today

- Collection of low-level metrics from iPhone and Apple Watch
  - *Accelerometer – (x, y, z)*
  - *Ambient light reading - (lux, placement, x, y)*
  - *Rotation Rate – (x, y, z)*
  - Speech Telephony
  - Visits
  - Message Usage
  - Keyboard Metrics
  - Device Usage



Overall Architecture




How does the incoming data look like
Data comes in as compressed json files
Showing an example of the accelerometer data
Each file has about 10 minutes of data
Structures by participant and metric

Some of the challenges with this

JSON was meant to make it easy for humans, but it can be hard …

Most analytically trained researchers are used to working with tabular and structured data

Hard to answer questions about the various fields, or number of total points, etc …

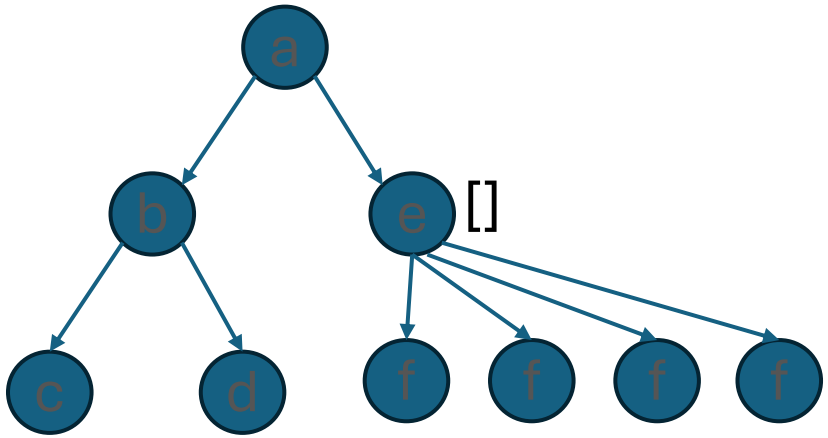
While NoSQL DB have gotten better, SQL is still the most widely used querying language.

Yes, SQL like solutions exist for NoSQL engines, but they have not become popular with research workflows.

Our Cave Man Solution

- Our Solution
  - Flatten the tree like JSON into structured view and store it
- What we had to do
  - **Schemer** – Samples JSON files and identifies schema. Looks out for changes in Schema by random sampling.
  - **Flattener** – Reclusively goes the JSON tree and flattens it to a structured view with some basic rules
  - **Filler** – Fills out the empty holes left by the flattener (can be turned off to save space), to give a cleaner tabular view
  - **Exporter** – Connects to storage locations / enginers – duckdb, S3 + Glue

Flattener



- Columns in tabular structure
  - A
  - A_b_c
  - A_b_d
    - Might need filling
  - A_e_f
    - Each element is a row

Tradeoffs

- Causes redundancies
- Increase storage size (we deserialize it)

BUT

- Much easier to query
- Can be parallized
- We understand structured data better

Optimizing the flattener

When we started

Took 58.72 seconds per 10-minute file

Now

1.52 second per 10-minute file

How?

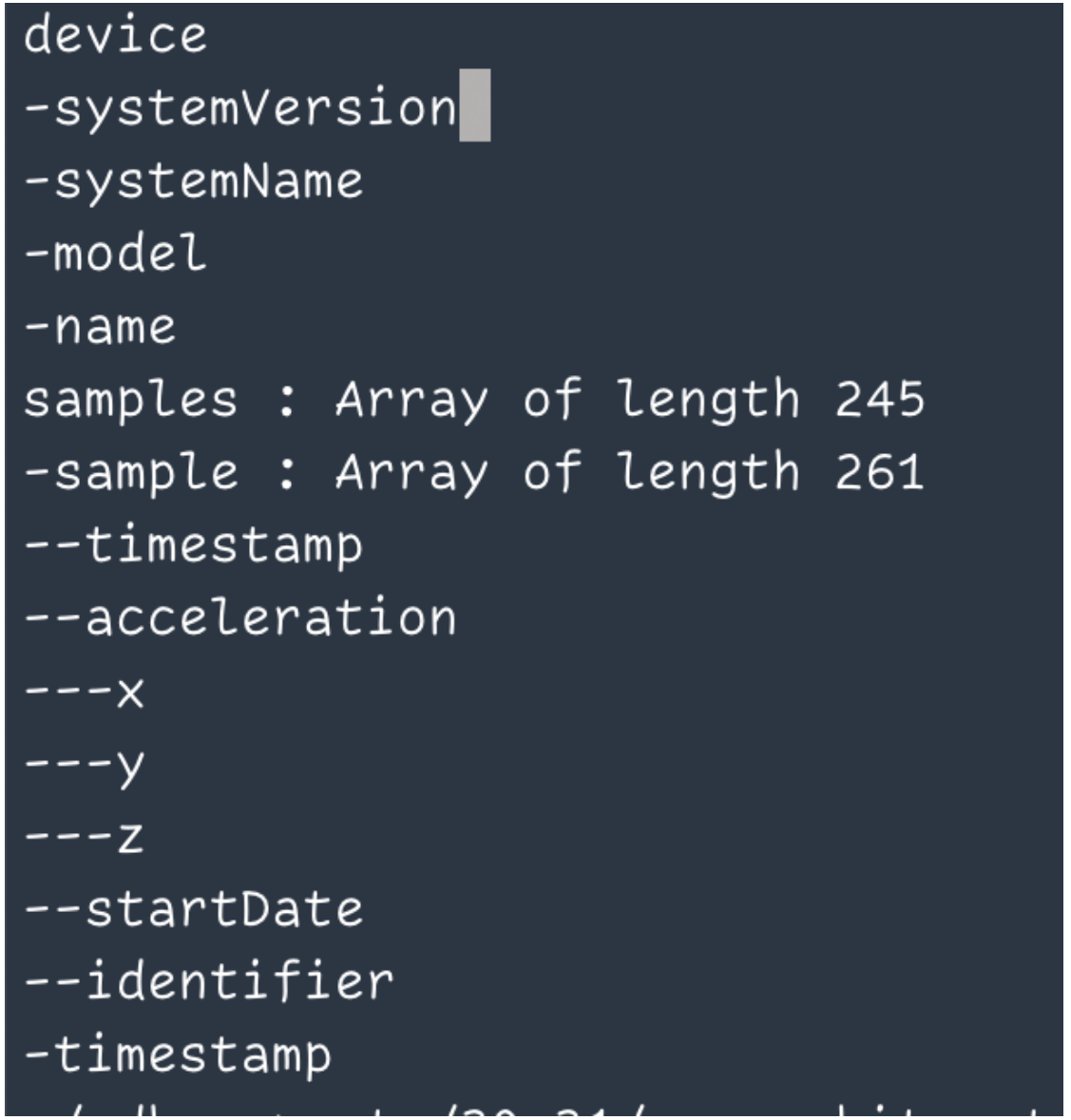
Substitute heavy non-primitive data types like pandas frame for primitive dictionary types.

Build recursion to work on columnar manner than record / row manner.

Schemer

- Experimented with having a fluent schema –
  - Let the engines infer the types and structure when ingesting
  - Went horribly wrong

- So, we created schemer
- *Sensorfabric.json.Raw.prettyPrintSchemer*



Schemer

Global schema file

```
{
    "participantidentifier" : "object",
    "device_systemVersion" : "object",
    "device_systemName" : "object",
    "device_model" : "object",
    "device_name" : "object",
    "samples_sample_timestamp" : "float64",
    "samples_sample_acceleration_x" : "float64",
    "samples_sample_acceleration_y" : "float64",
    "samples_sample_acceleration_z" : "float64",
    "samples_sample_startDate" : "object",
    "samples_sample_identifier" : "int64",
    "samples_timestamp" : "object",
    "samples_sample_rotationRate_x" : "float64",
    "samples_sample_rotationRate_y" : "float64",
    "samples_sample_rotationRate_z" : "float64",
    "samples_sample_lux" : "int64",
    "samples_sample_chromaticity_x" : "float64",
    "samples_sample_chromaticity_y" : "float64",
    "samples_sample_placement" : "object"
}
```

Key Features of using Cyverse

Our app makes it easy to automatically ingest from MyDataHelps
Does schema discovery, break down into tables, flattening, validation
Stores data locally in DuckDB.
You can share your data with other Cyverse users through DS
Partition duckdb files by dates, participants, study period, etc.
No hefty cloud costs.
Suitable for most small to medium studies

Where to find code

- https://github.com/UArizonaCB2/cyverse-sensorkit-helper.git
- https://github.com/UArizonaCB2/sensorfabric-py
- https://github.com/UArizonaCB2/sensorkit-mdh

Thank You

- shravanaras@arizona.edu

- Shout out to -
  - Tina
  - Nirav
  - Basyl
  - All Cyverse Team!