Standardized Array Value (SAV) Format of Measurement Data for State Estimation in Power Systems

By Ning Zhou, Gavin Trevorrow
Vestal, NY 13850, USA

06/13/2025
Ver 6.0

Contents

I. Summary of Contents

A. Objective:

The objective of this document is to define a **Standardized Array Value (SAV)** format for organizing measurement data used in power system state estimation (SE). This standardized format is designed to ensure consistency, accuracy, and ease of use across different simulation tools, software platforms, and research studies. The document also outlines how to interpret measurement accuracy levels and how to map scan indices to corresponding date and time values.

In addition to defining the SAV format, this document provides:

- Sample measurement data files for both dynamic simulations and converted IEEE test cases
- MATLAB code for solving state estimation problems using data in SAV format
- MATLAB scripts for converting measurement data from IEEE standard format into SAV format

B. Summary of Contents:

The document is organized into the following major components:

- **Standard Data Format – `SSE_X.Meas`:**
  Defines the structure of the measurement array, including the column indexing (`SSE_X.MeasCol`) and field definitions for various types of measurements.
- **Accuracy Level Mapping – `SSE_X.StdDev`:**
  Provides a standardized mapping from accuracy categories (negative integers) to corresponding standard deviation values.
- **Time Mapping – `SSE_X.Time`:**
  Explains how to convert scan indices to actual timestamps in UTC.
- **Sample Data Files:**
  Includes sample measurement datasets from both dynamic simulations (e.g., 3-machine 9-bus and 16-machine 68-bus systems) and IEEE standard cases (e.g., 14-bus, 57-bus, 118-bus, and 300-bus systems) formatted according to the SAV specification.
- **Sample MATLAB Code:**
  Provides implementation examples for:
    - Weighted Least Squares (WLS)-based steady-state estimation
    - Forecasting-aided state estimation considering time skew
    - Format conversion from IEEE to SAV

These resources are intended to facilitate reproducible and flexible testing of state estimation algorithms using standardized input formats.

II. Definition of the SAV Data Format

This chapter provides a detailed specification of the Standardized Array Value (SAV) format used for organizing measurement data in power system state estimation (SE). The SAV format is designed to be flexible and extensible, enabling it to accommodate a wide range of applications, from simple single-snapshot cases to complex multi-snapshot scenarios involving varying measurement types and accuracy levels. For basic use cases, only the first four fields in the `[SSE_X.Meas]` array are required to define a valid measurement dataset.

A. [SSE_X.MeasCol] Column Indexes of Measurement Data

[SSE_X.MeasCol] is a structure, which defines the column (col) indices of the fields defined in [SSE_X.Meas] of the above section. The following are examples on how to define [SSE_X.MeasCol].

Example A: a simple case of one snapshot of measurement, which has only 4 fields and 4 columns in [SSE_X.Meas].

- SSE_X.MeasCol.TYPE=[1]; % measurement (meas) type is stored in col 1 of SSE_X.Meas
- SSE_X.MeasCol.PID=[2]; % meas position ID is stored in column 2 of SSE_X.Meas
- SSE_X. MeasCol.VALUE=[3]; % 1 snapshot of meas is stored in column 3 of SSE_X.Meas
- SSE_X. MeasCol.STD=[4]; % meas STD is stored in column 4 of SSE_X.Meas

Example B: a case with 2 snapshots of measurements which has 5 fields and 8 columns in [SSE_X.Meas].

- SSE_X. MeasCol.NOTE=[0]; % a note to help reader understand the meaning of the row
- SSE_X.MeasCol.TYPE=[1]; % measurement type is stored in column 1 of SSE_X.Meas
- SSE_X. MeasCol.PID=[2]; % meas position ID is stored in column 2 of SSE_X.Meas
- SSE_X. MeasCol.STD=[3]; % meas STD is stored in column 3 of SSE_X.Meas
- SSE_X. MeasCol.VALUE=[4,5,6,7]; % 24snapshots of measurements are stored in columns 4-7.

B. [SSE_X.Meas] Measurement Data Array for SSE

[SSE_X.Meas] is an array for storing the measurement-related data for static state estimation (SSE). Here, [X] can be customized by the user to denote any specific case. For example, a user may set [X]=IEEEb57 makes a measurement array of [SSE_IEEEb57.Meas]. Each row in the [Meas] array represents a single measurement source (or channel). Each column of [Meas] store a field. The 13 potential fields of the measurements are defined as follows.

1. **Measurement Type (TYPE)**: Nature of the measurement, a negative integer.
    - 1: TSTEP, sampling time in seconds
    - -11: Pij, line real power measured at FromBus in pu
    - -12: Pji, line real power measured at ToBus in pu
    - -13: Pi, bus injection real power in pu
    - -14: Vang, bus voltage angle in radian
    - -15: Qij, line reactive power measured at FromBus in pu
    - -16: Qji, line reactive power measured at ToBus in pu
    - -17: Qi, bus injection reactive power in pu
    - -18: Vmag, bus voltage magnitude in pu
    - -101: Vreal, real part of bus voltage in pu
    - -102: Vimag, imaginary part of bus voltage in pu
    - -103: Iij_real, real part of line current at FromBus
    - -104: Iji_real, real part of line current at ToBus
    - -105: Iij_imag, imaginary part of line current at FromBus
    - -106: Iji_imag, imaginary part of line current at ToBus
    - -214: SE_Vang, estimated value of bus voltage angle in radian
    - -218: SE_Vmag, estimated value of bus voltage magnitude in pu
    - 14: Vang, true value of bus voltage angle in radian
    - 18: Vmag, true value of bus voltage magnitude in pu
2. **Position ID of the Measurement (PID)**: a unique identification number for the measurement, a positive integer.
    - For bus-related measurements, this column shall be the bus number where the measurement is taken from.
    - For branch-related measurements, this column shall be the branch number where the measurement is taken from (an integer).
3. **Metrics for Measurement Errors (STD):** A metric to quantify the measurement errors, a double precision number.
    - If the value is positive (>0), it is the standard deviation (STD) of the measurements.
    - If the value is zeros (=0), it indicates that the measurement is the true value from simulation.
    - If the value is a negative integer (<0 and integer), it represents a category of *accuracy levels*. The related measurement standard deviation is defined in "SSE_StdDev", which is defined in the next section.
4. **Measurement Value (VALUE)**: The actual measured value, a double precision number.
    - For angle measurements, it is in radians.
    - For other measurements, it is in per unit.

5. **Scan Time (TIME)**: Sampling time of the measurements in second, a double precision number.
6. **Measurement Feature (FEATURE)**: Feature of the measurement.
    - **1**: Actual measurement.
    - **2**: Pseudo measurement.
    - **3**: Zero-injection measurement.
7. **Skew Time (SKEW)**: Deviation from the static state estimation time, representing the time offset from the SSE time, a floating number, in seconds.
8. **Reliability of the measurement data (RELI):** define the reliability of the measurement data, an integer number
    - 1: normal data.
    - 0: missing data; the measurement is not available from the current scan.
    - -1: bad data; because the measurement value is out of the range of the measurement device.
9. **Lower limit of the measurements (LOWER):** Assigns the upper limit of the measurement device, a double precision number.
10. **Upper limits of the measurement (UPPER):** Assigns the upper limit of the measurement device, a double precision number.
11. **Measurement ID (MID)**: A unique identifier for each measurement, associated with its measurement device and measurement channel, a positive integer number, .
12. **Senor ID (SID)**: A unique positive identifier for the senor from which the measurement values are derived, a positive integer.
    - The Sensor ID is introduced to handle situations where multiple sensors measure the same variable. For example, there may be two bus voltage phasor measurements that measure the voltage at different nodes of the same bus. The Sensor ID can be used to group related measurements together. For instance, the magnitude and angle, or the real and imaginary parts of a voltage phasor, should have the same Sensor ID. Similarly, a pair of P and Q measurements should share the same Sensor ID.
    - Rule for Assigning Sensor ID: The Sensor ID is a unique positive integer identifier for each measurement set. This identifier is used to group related measurements together. The assignment of Sensor IDs should follow these rules:

        1. **Unique Identification**: Each Sensor ID must be unique across all measurement sources. If the measurements share the same source of sensor or calculation, they shall share the same Sensor ID.
        2. **Grouping by Measurement Set**:
            - **Complex Power Measurements**: Real power (P) and reactive power (Q) for a bus or branch share the same Sensor ID.
            - **Voltage Phasor Measurements**: The magnitude (Vmag) and angle (Vang) of a voltage phasor at a bus share the same Sensor ID.
            - **Current Phasor Measurements**: The real part (Ire) and imaginary part (Iim) of a current phasor in a branch share the same Sensor ID.
        3. **Sequential Assignment**: Sensor IDs should be assigned sequentially, starting from 1, to ensure no overlap and easy identification of new sets.
        4. **Consistent Usage**: Once a Sensor ID is assigned to a set of related measurements, it should consistently be used across all relevant data entries.

13. **NOTE:** In addition to the columns defined above, an optional cell called measurement "**Note"** can be included for each row in SSE_X.Meas to help a human reader understand the meaning of the measurement data. In the CSV file, this "Note" will be added as a side note at the end of the last effective column.

Note that for a simple case with one snapshot of measurement data, only the first 4 fields are needed to define a measurement data set [SSE_X.Meas] and each field only needs only one column. In contrast, for a case with 3 snapshots of measurements, field <VALUE> will need to have 3 columns and the column indices of the field is defined in a structure [SSE_X.MeasCol], which are detailed in the following section.

C. [SSE_X. StateTrue] True States Array for SSE

[SSE_X. StateTrue] Recorded true values of power system states. Its structure is similar to it in [SSE_X.Meas] as defined in the above subsections.

- SSE_X. StateTrue(1:NStates, VALUE): Bus voltage angles in radian
- SSE_X. StateTrue(NStates+1: 2*NStates, VALUE): Bus voltage magnitude in pu.

D. Assigning Standard Deviations for Accuracy Levels using [SSE_X.StdDev]

[SSE_X.StdDev] is an array variable that interprets the negative integer number of the "3. Metrics for Measurement Errors" defined in [SSE_X.Meas] into corresponding standard deviation values.

Each row in the **[SSE_X.StdDev]** array defines the accuracy level of one category of measurement device, with columns defined as follows:

- Column #1: the "3. Metrics for Measurement Errors" defined in **[SSE_X.Meas]**
    - Normally, it shall be a negative integer
- Column #2: the standard deviation of the corresponding measurements
    - Normally, it shall be a positive floating number

Here is an example of how you might define and interpret this variable:

Mapping

Accuracy Level	**Standard Deviation (σ)**
-1	0.001
-2	0.005
-3	0.01
-4	0.02
-5	0.05
-6	0.1

Accuracy Level	**Standard Deviation (σ)**
-7	0.2
-8	0.5
-9	1.0
-10	2.0

This table shows how different accuracy levels correspond to specific standard deviations. You can adjust the standard deviation values according to your specific requirements.

III. Sample Measurement Data Files in SAV Format

This chapter presents sample measurement datasets formatted according to the Standardized Array Value (SAV) specification. These datasets are derived from both dynamic simulations and benchmark IEEE test systems. The associated power flow models follow the MATPOWER format, while the measurement files conform to the SAV structure described in the previous chapter. These examples are intended to support validation, testing, and demonstration of state estimation algorithms under various system configurations and data scenarios.

<20_MeasurementSAV.zip> has the following measurement data files.

A. Measurement Data in SAV Format through Dynamic Simulation using PST

The objective of this section is to generate multi-snapshot measurement data for state estimation in the standard SAV format, using dynamic simulation conducted with Power System Toolbox (PST), a MATLAB based open-source code. This aims to support forecasting-aided state estimation while accounting for the time-skew problem.

1) Measurement data generated from the 3-machine 9-bus (3m9b) system

- Data3m9bMeasMult4SAV.csv: % Measurement data in SAV format (CSV file) for State estimation
- Data3m9bMeasMult4SAV.mat; % Measurement data in SAV format (matlab data file) for State estimation
- Data3m9bMeasSingle4SAV.csv: % Measurement data in SAV format (CSV file) for State estimation
- Data3m9bMeasSingle4SAV.mat; % Measurement data in SAV format (matlab data file) for State estimation
- Data3m9bCase_MAT.m % Bus [mpc.bus] and Branch [mpc.branch] models in MATPower (from PST format)
- Data3m9bCase_MAT.mat % Bus [mpc.bus] and Branch [mpc.branch] models in MATPower (from PST format)
- Data3m9bCase_PST.mat % Bus and Line model in PST format.

2) Measurement data generated from the 16-machine 68-bus (16m68b) system

- Data16m68bMeasMult4SAV.csv: % Measurement data in SAV format (CSV file) for State estimation
- Data16m68bMeasMult4SAV.mat; % Measurement data in SAV format (matlab data file) for State estimation
- Data16m68bMeasSingle4SAV.csv: % Measurement data in SAV format (CSV file) for State estimation
- Data16m68bMeasSingle4SAV.mat; % Measurement data in SAV format (matlab data file) for State estimation
- Data16m68bCase_MAT.m % Bus [mpc.bus] and Branch [mpc.branch] models in MATPower
- Data16m68bCase_MAT.mat % Bus [mpc.bus] and Branch [mpc.branch] models in MATPower
- Data16m68bCase_PST.mat % Bus and Line model in PST format.

B. Measurement Data in SAV Format converted from IEEE Standard Testing Cases

1) From IEEE 14-bus system (case14.m from MATPOWER)

- DataMeas014Bs001hSAV % Case #1 measurements from the IEEE 14 bus system in SAV format.
- DataMeas014Bs002hSAV % Case #2 measurements from the IEEE 14 bus system in SAV format.
- DataMeas014Bs003hSAV % Case #3 measurements from the IEEE 14 bus system in SAV format.

2) From IEEE 57-bus system (case57.m from MATPOWER)

- DataCSVMeas057Bs001SAV % Case #1 measurements from the IEEE 57 bus system in SAV format.
- DataCSVMeas057Bs002SAV % Case #2 measurements from the IEEE 57 bus system in SAV format.
- DataCSVMeas057Bs003SAV % Case #3 measurements from the IEEE 57 bus system in SAV format.

3) From IEEE 118-bus system (case118.m from MATPOWER)

- DataCSVMeas118Bs001SAV % Case #1 measurements from the IEEE 118 bus system in SAV format.
- DataCSVMeas118Bs002SAV % Case #2 measurements from the IEEE 118 bus system in SAV format.
- DataCSVMeas118Bs003SAV % Case #3 measurements from the IEEE 118 bus system in SAV format.

4) From IEEE 300-bus system (case300.m from MATPOWER)

- DataMeas300Bs001SAV % Case #1 measurements from the IEEE 300-bus system in SAV format.
- DataMeas300Bs002SAV % Case #2 measurements from the IEEE 300-bus system in SAV format.
- DataMeas300Bs003SAV % Case #3 measurements from the IEEE 300-bus system in SAV format.

IV. Sample MATLAB Code Files for State Estimation

This chapter provides sample MATLAB code for solving power system state estimation problems using measurement data in the SAV format. The included scripts implement the Weighted Least Squares (WLS) method for steady-state estimation, as well as advanced routines for forecasting-aided state estimation (FASE) that account for time-skew effects. These code examples are designed to support users in testing algorithms, validating simulation results, and exploring the impact of measurement timing on estimation accuracy.

<30_SSE_Code.zip> has the following sample MATLAB code.

A. Weight Least Squares Solution (WLS) for Steady State Estimation (SSE)

The following codes perform SSE using standard WLS solution procedure.

1) Matlab code for solving the SSE problems using the 3m9b system

- Run main codes: < testSSE_3m9b_Step04MATv01.m > using *.mat as inputs
- Run main codes: < testSSE_3m9b_Step04CSVv01.m > using *.csv as inputs

2) Matlab code for solving the problems from 16m68b system

- Run main codes: < testSSE_16m68b_Step04MATv01.m > using *.mat as inputs
- Run main codes: < testSSE_16m68b_Step04CSVv01.m > using *.csv as inputs

B. State Estimation Considering Time-Skew (TS) Problems (SEcTS)

The following codes perform forecasting aided state estimation (FASE) considering time-skew (TS) problems using the measurement data stored in both PST format and standard SAV format. The objective to evaluate the impact of time-skew problems on the estimation accuracy of state estimation.

1) Matlab code for solving the SEcTS using the 3m9b system

- Run main codes: < testFASE_SAV_MAT_3m9b_zn08_gt01> using *.mat as inputs
- Run main codes: < testFASE_SAV_CSV_3m9b_zn08_gt01 > using *.csv as inputs

2) Matlab code for solving the SEcTS using the 16m68b system

- Run main codes: <testFASE_SAV_MAT_16m68b_zn08_gt01 > using *.mat as inputs
- Run main codes: < testFASE_SAV_CSV_16m68b_zn08_gt01 > using *.csv as inputs

V. Converting IEEE-Formatted Measurement Data to SAV Format

This chapter introduces MATLAB scripts for converting measurement data from the IEEE standard format into the Standardized Array Value (SAV) format. These conversion tools facilitate interoperability between legacy datasets and the SAV-based processing framework. The provided scripts support both `.mat` and `.csv` input formats and generate corresponding `.mat` and `.csv` outputs in SAV structure. This conversion enables consistent use of standardized data across different estimation tools and simulation environments.

<40_IEEE2SAV.zip> has the following code for converting measurement data from IEEE format to SAV format.

A. Matlab code for converting *.mat in IEEE format to SAV format

Run main codes: < testSSE_IEEE2SAV05.m > using *.mat as inputs

- **Inputs:**
  - -MeasDataFileIEEE='B0057s_003.mat'; % the filename of the measurement data for SSE in IEEE format
  - -PowerFlowFileMAT='case57'; % power flow case (corresponding to the measurement) in MATpower format, "branch"
  - -Filename4Meas: % setup the file names for saving measurement data in SAV format
- **Outputs**
  - MeasSAV4SSE.Meas; % measurement data stored in SAV format
  - ' DataMeas057Bs003SAV.CSV'; % saved meausrement data in SAV format in a *.CSV file
  - ' DataMeas057Bs003SAV.mat'; % saved meausrement data in SAV format in a *.mat file

B. Matlab code for converting *.csv in IEEE format to SAV format

Run main codes: < testSSE_IEEE2SAV05csv > using *.csv as inputs

- **Inputs:**
  - -MeasDataFileIEEE='B0057s_data_001.csv'; % the filename of the measurement data for SSE in IEEE format
  - -MeasHeaderFileIEEE='B0057s_header_001.csv'; % the filename of the measurement data for SSE in IEEE format
  - -Filename4Meas: % setup the file names for saving measurement data in SAV format
- **Outputs**
  - MeasSAV4SSE.Meas; % measurement data stored in SAV format
  - ' DataCSVMeas057Bs003SAV.CSV'; % saved meausrement data in SAV format in a *.CSV file
  - ' DataCSVMeas057Bs003SAV.mat'; % saved meausrement data in SAV format in a *.mat file

C. Converted data files

- DataMeas014Bs001hSAV % Case #1 measurements from the IEEE 14 bus system in SAV format.
- DataMeas014Bs002hSAV % Case #2 measurements from the IEEE 14 bus system in SAV format.
- DataMeas014Bs003hSAV % Case #3 measurements from the IEEE 14 bus system in SAV format.
- DataCSVMeas057Bs001SAV % Case #1 measurements from the IEEE 57 bus system in SAV format.
- DataCSVMeas057Bs002SAV % Case #2 measurements from the IEEE 57 bus system in SAV format.
- DataCSVMeas057Bs003SAV % Case #3 measurements from the IEEE 57 bus system in SAV format.
- DataCSVMeas118Bs001SAV % Case #1 measurements from the IEEE 118 bus system in SAV format.
- DataCSVMeas118Bs002SAV % Case #2 measurements from the IEEE 118 bus system in SAV format.
- DataCSVMeas118Bs003SAV % Case #3 measurements from the IEEE 118 bus system in SAV format.
- DataMeas300Bs001SAV % Case #1 measurements from the IEEE 300-bus system in SAV format.
- DataMeas300Bs002SAV % Case #2 measurements from the IEEE 300-bus system in SAV format.
- DataMeas300Bs003SAV % Case #3 measurements from the IEEE 300-bus system in SAV format.

VI. MATLAB Code for State Estimation Using SAV Files Converted from IEEE Format

This chapter provides sample MATLAB code for performing state estimation using measurement data that has been converted from IEEE standard format into the SAV format. The included scripts apply the Weighted Least Squares (WLS) method to datasets from several IEEE benchmark systems (14-bus, 57-bus, 118-bus, and 300-bus). Separate routines are provided for handling both `.mat` and `.csv` inputs, allowing users to estimate system states and branch power flows using consistent and reproducible SAV-based workflows.

<50_SSE_Code4IEEE.zip> has the following codes to solve the state estimation problems from

- IEEE 14-bus system
- IEEE 57-bus system
- IEEE 118-bus system
- IEEE 300-bus system

A. State estimation using the Measurement Data in SAV format (in *.MAT)

Run main codes: < testSSE_57b_Step07MATv02.m> using *.mat as inputs

- **Example Inputs:**
  - FilenameMeas='DataMeas057Bs003SAV.mat';

- **Outputs:**
  - x0: estimated states, i.e., estimated bus voltage magnitudes and angles.
  - Sf: estimated branch power (Pij) at the from-bus of a branch
  - St: estimated branch power (Pji) at the to-bus of a branch

B. State estimation using the Measurement Data in SAV format (in *.CVS)

Run main codes: < testSSE_57b_Step07CSVv02.m > using *.csv as inputs

- **Example Inputs:**
  - mpc=case57; % load the power flow model in MAT format
  - FilenameMeasCSV='DataMeas057Bs003SAV.CSV'; % file names for measurements and true states,

- **Outputs:**
  - x0: estimated states, i.e., estimated bus voltage magnitudes and angles.
  - Sf: estimated branch power (Pij) at the from-bus of a branch
  - St: estimated branch power (Pji) at the to-bus of a branch