ARPA-E GO Competition Challenge 3
Data Format

Terrence W.K. Mak   Carleton Coffrin   Christopher DeMarco   Ray Duthu
Stephen Elbert   Brent Eldridge   Tarek Elgindy   Scott Greene   Jesse Holzer
Nongchao Guo   Elaine Hale   Bernard Lesieutre   Colin McMillan
Hans Mittelmann   Richard O'Neill   Thomas Overbye   Bryan Palmintier
Farnaz Safdarian   Ahmad Tbaileh   Pascal Van Hentenryck   Arun Veeramany
Jessica Wert

Oct 11, 2022 - v1.0.4

Contents

1 Overview

This document contains a description of the data format for the ARPA-E GO Competition Challenge 3 and beyond. It consists of a number of sections.

I **Naming conventions and attributes:** the names of ***all*** the attributes for each electrical & market components used by the competition, classified by component types into tables;

II **JSON for static electrical components:** the static network description in JSON, e.g., static labels & data, a given system starting point, and non-time-varying parameters for each of the electrical components;

III **JSON for time-varying components:** a collection of attributes for each electrical & market components that describe the component status over time; the system status may represent, for instance, inputs to optimization problems, forecasts, or approximations to the optimization problems;

IV **JSON for contingencies:** a list of contingencies describing each of the contingent cases;

V **JSON for competition solutions:** a collection of attributes for each electrical & market components that describe the solutions for the competition;

VI **Data parsing:** the technical details for using the data format files specifically for the Challenge 3 formulation document.

It is important to emphasize that the values of some of these attributes can be time series instead of primitive data types (e.g., integers, floating numbers, and strings). In the data format, we will use arrays and inner JSON objects to represent time series. This is particularly important for multi-stage periods that are ubiquitous in power systems, and essential to solve optimization problems over a finite time horizon. To further simplify the data format, the format uses binary (0 & 1) values to indicate booleans (`false` & `true`). In addition, the format uses `uid` strings to globally label and identify components and their elements, primarily for ease of parsing and avoiding ambiguity.

2 Naming Conventions

This section summarizes the attributes for all the components used throughout the data format for quick references. Names, descriptions, types, and corresponding/associated formulation symbol (if exists) for each of the attributes are shown. Attributes that are required for the competition will be marked with a 'Y' in the 'Required' (Req) column. Additional optional attributes are marked with an 'N' in the Req column. Optional attributes are not guaranteed to exist. If they are present, competitors are free to use them to guide their development process, in particular for debugging. There are no guarantees that components of the same class have the same set of optional attributes. Ultimately, optional fields are outside the formulation and competitors should not assume their presence for their code to work; they may be relevant for future Challenges or post-Challenge analysis.

For ease of reference and convenience for describing the data formats, we also mark the nature of the attributes in the 'Section' (Sec) column. The corresponding symbol used in the formulation for each data item is contained in the 'Symbol' (Sym) column. For the input data format, attributes will be marked with:

- C — if it is a contingency parameter used in the reliability file section;
- S — if it is a static parameter and is expected to be presented in the static network file section;
- T — if it is a time series and is expected to be presented in the time series data file section, and;
- B — if it is a globally unique identifier/common attribute and it is necessary to be presented across both static and timeseries sections for identification/referencing purposes.

Similarly for the output data format, attribute will be marked with:

- C — if it is a contingency parameter;

- T — if it is a time series, and;
- B — if it is a globally unique identifier.

Attributes of inner JSON objects (inner attributes) are listed in separate tables & subsections for ease of reference. Inner attributes have a different scope and readers are advised to consult the detailed examples in subsequent sections during implmentation. As a reminder, the attribute list could be modified in the future. Competitors should always refer to the latest announcement for any updates.

2.1 General

Input Attributes The attributes for the general JSON object used by the input data format are listed below.

Global time attributes	Description (Type)	Req	Sec	Sym
`time_periods`	The number of time periods (Int)	Y	T	$\lvert T\rvert$
`interval_duration`	Time duration of the intervals, per time period in hours (Array of Float)	Y	T	d_t
`timestamp_start`	Period beginning timestamp for the first interval as string: YYYY-MM-DDThh:mm:ss at UTC (Timestamp)	N	S	
`timestamp_stop`	Period beginning timestamp for the interval following the last interval as string: YYYY-MM-DDThh:mm:ss at UTC (Timestamp)	N	S	
Qualitative descriptors	**Description (Type)**	**Req**	**Sec**	**Sym**
`season`	Season of the year the problem lies within (String: Winter, Spring, Summer, Fall)	N	S	
`electricity_demand`	How demand compares to other times of the year/season (String: Peak, High, Average, Low, Minimum)	N	S	
`vre_availability`	How variable renewable energy availability compares to other times of the year/season (String: High, Average, Low)	N	S	
`solar_availability`	How solar availability compares to other times of the year/season (String: High, Average, Low)	N	S	
`wind_availability`	How wind availability compares to other times of the year/season (String: High, Average, Low)	N	S	
`weather_temperature`	How outside temperature compares to other times of the year/season (String: Hottest, Warm, Average, Cool, Cold)	N	S	
`day_type`	What kind of weekday is represented (String: Weekday, Weekend, Holiday)	N	S	
`net_load`	How the net-load profile compares to other times of the year/season (String: Peak, High, Average, Low, Minimum, High-Up-Ramp, High-Down-Ramp)	N	S	
Normalization attributes	**Description (Type)**	**Req**	**Sec**	**Sym**
`base_norm_mva`	Base MVA normalization constant (Float)	Y	S	

2.2 Violation Costs & Parameters

Input Attributes The attributes for describing violation costs are listed below.

Global violation attributes	Description (Type)	Req	Sec	Sym
`p_bus_vio_cost`	Bus violation costs for active power violation in $/pu-h (Float)	Y	S	c^p
`q_bus_vio_cost`	Bus violation costs for reactive power violation in $/pu-h (Float)	Y	S	c^q
`s_vio_cost`	Branch violation costs for thermal violation in $/pu-h (Float)	Y	S	c^s

2.3 Bus

Input Attributes The attributes for buses used by the input data format are listed below.

Input attributes	Description (Type)	Req	Sec	Sym
`uid`	Bus unique identifier (String)	Y	B	
`vm_ub`	Voltage magnitude upper bound in p.u. (Float)	Y	S	v^{max}
`vm_lb`	Voltage magnitude lower bound in p.u. (Float)	Y	S	v^{min}
`active_reserve_uids`	List of active reserve zones (uids) that the bus participating (Array of String)	Y	S	
`reactive_reserve_uids`	List of reactive reserve zones (uids) that the bus participating (Array of String)	Y	S	
Location information	Description (Type)	Req	Sec	Sym
`area`	Bus control area (String)	N	S	
`zone`	Bus control zone (String)	N	S	
`longitude`	Bus location - longitude in decimal degree (Float)	N	S	
`latitude`	Bus location - latitude in decimal degree (Float)	N	S	
`city`	Bus city location (String)	N	S	
`county`	Bus county location (String)	N	S	
`state`	Bus state location (String)	N	S	
`country`	Bus country location (String)	N	S	
Operations information	Description (Type)	Req	Sec	Sym
`con_loss_factor`	Contingency participation loss factor (Float)	N	S	α
`base_nom_volt`	Bus nominal voltage (Float)	Y	S	
`type`	Bus type (String: PQ, PV, Slack, Not_used)	N	S	
`initial_status`	A JSON inner object storing data for initial time step (initial_status)	Y	S	
Initial condition attributes	[Inner Attributes] Within `initial_status`			
`vm`	Bus voltage magnitude in p.u. (Float)	Y	S	
`va`	Bus voltage angle in radian (Float)	Y	S	

Output Attributes The attributes for buses used by the output data format are listed below.

Output attributes	Description (Type)	Req	Sec	Sym
`uid`	Bus unique identifier (String)	Y	B	
`vm`	Voltage magnitude in p.u. (Array of Float)	Y	T	v
`va`	Voltage angle in radian (Array of Float)	Y	T	θ

2.4 Shunt

Input Attributes The attributes for bus shunts used by the input data format are listed below.

Input attributes	Description (Type)	Req	Sec	Sym
`uid`	Shunt unique identifier (String)	Y	B	
`bus`	Unique identifier for connecting bus (String)	Y	S	
`gs`	Shunt conductance for one step in p.u. (Float)	Y	S	g^{sh}
`bs`	Shunt susceptance for one step in p.u. (Float)	Y	S	b^{sh}
`step_ub`	Maximum step number (Int)	Y	S	$u^{sh,max}$
`step_lb`	Minimum step number (Int)	Y	S	$u^{sh,min}$
`initial_status`	A JSON inner object storing data for initial time step (initial_status)	Y	S	
Initial condition attributes	[Inner Attributes] Within `initial_status`			
`step`	Number of step (Int)	Y	S	u^{sh}

Output Attributes The attributes for shunts used by the output data format are listed below.

Output attributes	Description (Type)	Req	Sec	Sym
`uid`	Shunt unique identifier (String)	Y	B	
`step`	Number of step (Array of Int)	Y	T	u^{sh}

2.5 Dispatchable Devices: Simple Producing & Consuming Devices

Input Attributes The attributes for simple energy producing and consuming devices used by the input data format are listed below.

Device attributes	Description (Type)	Req	Sec	Sym
uid	Producing device unique identifier (String)	Y	B	
bus	Unique identifier for connecting bus (String)	Y	S	
device_type	Type of device (String: producer, consumer)	Y	S	
description	Detail description of the device (String) e.g. fuel type description: wind/solar/coal/gas	N	S	
vm_setpoint	Voltage magnitude setpoint in p.u. (Float)	N	S	
nameplate_capacity	Reference capacity in p.u. (Float)	N	S	
startup_cost	Device start up cost in $ (Float)	Y	S	c^{su}
startup_states	Array of downtime dependent start up states, where each states is an array with exactly two elements: 1) start up cost adjustments in $ (Float), 2) maximum down time in hr (Float) (Array of Float Float)	Y	S	c^{sus} $d^{dn,max}$
shutdown_cost	Device shut down cost in $ (Float)	Y	S	c^{sd}
startups_ub	Array of time interval startup data blocks, where each data block is an array with exactly three elements: 1) interval starting time in hr (Float), 2) interval ending time in hr (Float), and 3) maximum startups within the interval (Int) (Array of Float Float Int)	Y	S	$u^{su,max}$
energy_req_ub	Array of energy upper bound requirement data blocks, where each data block is an array with exactly three elements: 1) interval starting time in hr (Float), 2) interval ending time in hr (Float), and 3) maximum energy within the interval in p.u. (Float) (Array of Float Float Float)	Y	S	t^{max}, t^{min} e^{max}
energy_req_lb	Array of energy lower bound requirement data blocks, where each data block is an array with exactly three elements: 1) interval starting time in hr (Float), 2) interval ending time in hr (Float), and 3) minimum energy within the interval in p.u. (Float) (Array of Float Float Float)	Y	S	t^{max}, t^{min} e^{min}
on_cost	Device fixed operating cost in $ (Float)	Y	S	c^{on}
in_service_time_lb	Minimum uptime in service in hr (Float)	Y	S	$d^{up,min}$
down_time_lb	Minimum downtime in hr (Float)	Y	S	$d^{dn,min}$
p_ramp_up_ub	(Case: producer) Max production ramp up when operating in p.u./hr (Float)	Y	S	p^{ru}
	(Case: consumer) Max consumption ramp up when operating in p.u./hr (Float)	Y	S	p^{ru}
p_ramp_down_ub	(Case: producer) Max production ramp down when operating in p.u./hr (Float)	Y	S	p^{rd}
	(Case: consumer) Max consumption ramp down when operating in p.u./hr (Float)	Y	S	p^{rd}
p_startup_ramp_ub	(Case: producer) Max production ramp up when start up in p.u./hr (Float)	Y	S	$p^{ru,su}$
	(Case: consumer) Max consumption ramp up when start up in p.u./hr (Float)	Y	S	$p^{ru,su}$
p_shutdown_ramp_ub	(Case: producer) Max production ramp down when shut down in p.u./hr (Float)	Y	S	$p^{rd,sd}$
	(Case: consumer) Max consumption ramp down when shut down in p.u./hr (Float)	Y	S	$p^{rd,sd}$
initial_status	A JSON inner object storing data for initial time step (initial_status)	Y	S	
Flags for extra parameters	— Binary (0/1) to indicate extra parameters	Req	Sec	Sym
q_linear_cap	Device has additional reactive constraint (Binary)	Y	S	
q_bound_cap	Device has additional reactive bounds (Binary)	Y	S	

Device attributes	— Time varying operational attributes	Req	Sec	Sym
on_status_ub	On status indicator upper bound (Array of Binary)	Y	T	$u^{on,max}$
on_status_lb	On status indicator lower bound (Array of Binary)	Y	T	$u^{on,min}$
p_ub	(Case: producer) Upper bound of active dispatch in p.u. (Array of Float)	Y	T	p^{max}
	(Case: consumer) Upper bound of active demand in p.u. (Array of Float)	Y	T	p^{max}
p_lb	(Case: producer) Lower bound of active dispatch in p.u. (Array of Float)	Y	T	p^{min}
	(Case: consumer) Lower bound of active demand in p.u. (Array of Float)	Y	T	p^{min}
q_ub	(Case: producer) Upper bound of reactive dispatch in p.u.(Array of Float)	Y	T	q^{max}
	(Case: consumer) Upper bound of reactive demand in p.u. (Array of Float)	Y	T	q^{max}
q_lb	(Case: producer) Lower bound of reactive dispatch in p.u. (Array of Float)	Y	T	q^{min}
	(Case: consumer) Lower bound of reactive demand in p.u. (Array of Float)	Y	T	q^{min}
cost	A time series array of cost functions, where each function is represented as an array of cost block. A cost block is an array (pair) with exactly two elements: 1) marginal cost in $/p.u.-hr (Float), and 2) block size in p.u. (Float) (Array of Array of Array of Float Float)	Y	T	c^{en}, p^{max}

Reserve attributes	Description (Type)	Req	Sec	Sym
p_reg_res_up_ub	Maximum regulation reserve up in p.u. (Float)	Y	S	$p^{rgu,max}$
p_reg_res_down_ub	Maximum regulation reserve down in p.u. (Float)	Y	S	$p^{rgd,max}$
p_syn_res_ub	Maximum synchronized reserve in p.u. (Float)	Y	S	$p^{scr,max}$
p_nsyn_res_ub	Maximum non-synchronized reserve in p.u. (Float)	Y	S	$p^{nsc,max}$
p_ramp_res_up_online_ub	Maximum ramp up reserve when online in p.u. (Float)	Y	S	$p^{rru,on,max}$
p_ramp_res_down_online_ub	Maximum ramp down reserve when online in p.u. (Float)	Y	S	$p^{rrd,on,max}$
p_ramp_res_up_offline_ub	Maximum ramp up reserve when offline in p.u. (Float)	Y	S	$p^{rru,off,max}$
p_ramp_res_down_offline_ub	Maximum ramp down reserve when offline in p.u. (Float)	Y	S	$p^{rrd,off,max}$
Time varying reserve attributes	Description (Type)	Req	Sec	Sym
p_reg_res_up_cost	Costs for regulation reserve up in $/pu-h (Array of Float)	Y	T	c^{rgu}
p_reg_res_down_cost	Costs for regulation reserve down in $/pu-h (Array of Float)	Y	T	c^{rgd}
p_syn_res_cost	Costs for synchronized reserve in $/pu-h (Array of Float)	Y	T	c^{scr}
p_nsyn_res_cost	Costs for non-synchronized reserve in $/pu-h (Array of Float)	Y	T	c^{nsc}
p_ramp_res_up_online_cost	Costs for ramp up reserve when online in $/pu-h (Array of Float)	Y	T	$c^{rru,on}$
p_ramp_res_down_online_cost	Costs for ramp down reserve when online in $/pu-h (Array of Float)	Y	T	$c^{rrd,on}$
p_ramp_res_up_offline_cost	Costs for ramp up reserve when offline in $/pu-h (Array of Float)	Y	T	$c^{rru,off}$
p_ramp_res_down_offline_cost	Costs for ramp down reserve when offline in $/pu-h (Array of Float)	Y	T	$c^{rrd,off}$
q_res_up_cost	Costs for reactive reserve up in $/pu-h (Array of Float)	Y	T	c^{qru}
q_res_down_cost	Costs for reactive reserve down in $/pu-h (Array of Float)	Y	T	c^{qrd}

Initial condition attributes	[Inner Attributes] Within inner initial_status JSON object			
on_status	On status indicator for initial time step (Binary)	Y	S	$u^{on,0}$
p	(Case: producer) Active production for initial time step in p.u. (Float)	Y	S	p^0
	(Case: consumer) Active consumption for initial time step in p.u. (Float)	Y	S	p^0
q	(Case: producer) Reactive production for initial time step in p.u. (Float)	Y	S	q^0
	(Case: consumer) Reactive consumption for initial time step in p.u. (Float)	Y	S	q^0
accu_down_time	Accumulated down time in hr (Float)	Y	S	
accu_up_time	Accumulated up time in hr (Float)	Y	S	
Reactive cap. attributes	[Conditional Attributes] Exist if q_linear_cap is 1			
q_0	(Case: producer) Reactive production at zero active production in p.u. (Float)	Y	S	q^{p0}
	(Case: consumer) Reactive consumption at zero active consumption in p.u. (Float)	Y	S	q^{p0}
beta	Slope of active-reactive capability curve (Float)	Y	S	β
Reactive cap. attributes	[Conditional Attributes] Exist if q_bound_cap is 1			
q_0_ub	(Case: producer) Max reactive production at zero active production in p.u. (Float)	Y	S	$q^{max,p0}$
	(Case: consumer) Max reactive consumption at zero active consumption in p.u. (Float)	Y	S	$q^{max,p0}$
q_0_lb	(Case: producer) Min reactive production at zero active production in p.u. (Float)	Y	S	$q^{min,p0}$
	(Case: consumer) Min reactive consumption at zero active consumption in p.u. (Float)	Y	S	$q^{min,p0}$
beta_ub	Upper bound for slope of active-reactive capability curve (Float)	Y	S	β^{max}
beta_lb	Lower bound for slope of active-reactive capability curve (Float)	Y	S	β^{min}

Output Attributes The attributes for single mode generating units used by the output data format are listed below.

Output attributes	Description (Type)	Req	Sec	Sym
uid	Device unique identifier (String)	Y	B	
on_status	Connection status (Array of Binary)	Y	T	u^{on}
p_on	(Case: producer) Active production in p.u. (Array of Float)	Y	T	p^{on}
	(Case: consumer) Active consumption in p.u. (Array of Float)	Y	T	p^{on}
q	(Case: producer) Reactive production in p.u. (Array of Float)	Y	T	q
	(Case: consumer) Reactive consumption in p.u. (Array of Float)	Y	T	q
p_reg_res_up	Regulation up reserve in p.u. (Array of Float)	Y	T	p^{rgu}
p_reg_res_down	Regulation down reserve in p.u. (Array of Float)	Y	T	p^{rgd}
p_syn_res	Synchronized reserve in p.u. (Array of Float)	Y	T	p^{scr}
p_nsyn_res	Non-synchronized reserve in p.u. (Array of Float)	Y	T	p^{nsc}
p_ramp_res_up_online	Ramp up reserve when online in p.u. (Array of Float)	Y	T	$p^{rru,on}$
p_ramp_res_down_online	Ramp down reserve when online in p.u. (Array of Float)	Y	T	$p^{rrd,on}$
p_ramp_res_up_offline	Ramp up reserve when offline in p.u. (Array of Float)	Y	T	$p^{rru,off}$
p_ramp_res_down_offline	Ramp down reserve when offline in p.u. (Array of Float)	Y	T	$p^{rrd,off}$
q_res_up	Reactive reserve up in p.u. (Array of Float)	Y	T	q^{qru}
q_res_down	Reactive reserve down in p.u. (Array of Float)	Y	T	q^{qrd}

2.6 AC Transmission Line

Input Attributes The attributes for AC transmission lines used by the input data format are listed below.

Input attributes	Description (Type)	Req	Sec	Sym
uid	AC line unique identifier (String)	Y	B	
fr_bus	Unique identifier for connecting from bus (String)	Y	S	
to_bus	Unique identifier for connecting to bus (String)	Y	S	
r	Series resistance in p.u. (Float)	Y	S	r^{sr}
x	Series reactance in p.u. (Float)	Y	S	x^{sr}
b	Shunt susceptance in p.u. (Float)	Y	S	b^{ch}
mva_ub_nom	MVA limit, nominal rating in p.u. (Float)	Y	S	s^{max}
mva_ub_sht	MVA limit, short term rating in p.u. (Float)	N	S	
mva_ub_em	MVA limit, emergency rating in p.u. (Float)	Y	S	$s^{max,ctg}$
connection_cost	AC Line connection cost in $ (Float)	Y	S	c^{su}
disconnection_cost	AC line disconnection cost in $ (Float)	Y	S	c^{sd}
initial_status	A JSON inner object storing data for initial time step (initial_status)	Y	S	
additional_shunt	Branch has additional shunt components (Binary)	Y	S	
Additional shunt attributes	[Conditional Attributes] Exist if additional_shunt is 1			
g_fr	Conductance for shunt component at from bus in p.u. (Float)	Y	S	g^{fr}
b_fr	Susceptance for shunt component at from bus in p.u. (Float)	Y	S	b^{fr}
g_to	Conductance for shunt component at to bus in p.u. (Float)	Y	S	g^{to}
b_to	Susceptance for shunt component at to bus in p.u. (Float)	Y	S	b^{to}
Initial condition attributes	[Inner Attributes] Within initial_status			
on_status	Connection status (Binary)	Y	S	u^{on}

Output Attributes The attributes for AC transmission lines used by the output data format are listed below.

Output attributes	Description (Type)	Req	Sec	Sym
uid	AC line unique identifier (String)	Y	B	
on_status	Connection status (Array of Binary)	Y	T	u^{on}

2.7 Two Winding Transformer

Input Attributes The attributes for two winding transformers used by the input data format are listed below.

Input attributes	Description (Type)	Req	Sec	Sym
uid	Transformer unique identifier (String)	Y	B	
fr_bus	Unique identifier for connecting from/tap bus (String)	Y	S	
to_bus	Unique identifier for connecting to bus (String)	Y	S	
r	Series resistance in p.u. (Float)	Y	S	r^{sr}
x	Series reactance in p.u. (Float)	Y	S	x^{sr}
b	Shunt susceptance in p.u. (Float)	Y	S	b^{ch}
tm_ub	Upper bound for off-nominal tap ratio in p.u. (Float)	Y	S	τ^{max}
tm_lb	Lower bound for off-nominal tap ratio in p.u. (Float)	Y	S	τ^{min}
ta_ub	Upper bound for phase shifting angle in radian (Float)	Y	S	ϕ^{max}
ta_lb	Lower bound for phase shifting angle in radian (Float)	Y	S	ϕ^{min}
mva_ub_nom	MVA limit, nominal rating in p.u. (Float)	Y	S	s^{max}
mva_ub_sht	MVA limit, short term rating in p.u. (Float)	N	S	
mva_ub_em	MVA limit, emergency rating in p.u. (Float)	Y	S	$s^{max,ctg}$
connection_cost	Transformer connection cost in $ (Float)	Y	S	c^{su}
disconnection_cost	Transformer disconnection cost in $ (Float)	Y	S	c^{sd}
initial_status	A JSON inner object storing data for initial time step (initial_status)	Y	S	
additional_shunt	Transformer has additional shunt components (Binary)	Y	S	

Additional shunt attributes	[Conditional Attributes] Exist if `additional_shunt` is 1			
`g_fr`	Conductance for shunt component at from bus in p.u. (Float)	Y	S	g^{fr}
`b_fr`	Susceptance for shunt component at from bus in p.u. (Float)	Y	S	b^{fr}
`g_to`	Conductance for shunt component at to bus in p.u. (Float)	Y	S	g^{to}
`b_to`	Susceptance for shunt component at to bus in p.u. (Float)	Y	S	b^{to}
Initial condition attributes	[Inner Attributes] Within `initial_status`			
`on_status`	Connection status (Binary)	Y	S	u^{on}
`tm`	Off-nominal tap ratio in p.u. (Float)	Y	S	τ
`ta`	Phase shifting angle in radian (Float)	Y	S	ϕ

Output Attributes The attributes for two winding transformers used by the output data format are listed below.

Output attributes	Description (Type)	Req	Sec	Sym
`uid`	Transformer line unique identifier (String)	Y	B	
`tm`	Off-nominal taps ratio in p.u. (Array of Float)	Y	T	τ
`ta`	Phase shifting angle in radian (Array of Float)	Y	T	ϕ
`on_status`	Connection status (Array of Binary)	Y	T	u^{on}

2.8 DC Line

Input Attributes The attributes for DC lines used by the input data format are listed below.

Input attributes	Description (Type)	Req	Sec	Sym
`uid`	DC line unique identifier (String)	Y	B	
`fr_bus`	Unique identifier for connecting from bus (String)	Y	S	
`to_bus`	Unique identifier for connecting to bus (String)	Y	S	
`pdc_ub`	Maximum active power in p.u. (Float)	Y	S	$p^{dc,max}$
`qdc_fr_ub`	Maximum reactive power, from bus in p.u. (Float)	Y	S	$q^{dc,fr,max}$
`qdc_fr_lb`	Minimum reactive power, from bus in p.u. (Float)	Y	S	$q^{dc,fr,min}$
`qdc_to_ub`	Maximum reactive power, to bus in p.u. (Float)	Y	S	$q^{dc,to,max}$
`qdc_to_lb`	Minimum reactive power, to bus in p.u. (Float)	Y	S	$q^{dc,to,min}$
`initial_status`	A JSON inner object storing data for initial time step (initial_status)	Y	S	
Initial condition attributes	[Inner Attributes] Within `initial_status`			
`pdc_fr`	Active power flow in p.u. (Float)	Y	S	p^{fr}
`qdc_fr`	Reactive power flow, from bus in p.u. (Float)	Y	S	q^{fr}
`qdc_to`	Reactive power flow, to bus in p.u. (Float)	Y	S	q^{to}

Output Attributes The attributes for DC lines used by the output data format are listed below.

Output attributes	Description (Type)	Req	Sec	Sym
`uid`	DC line unique identifier (String)	Y	B	
`pdc_fr`	Active power flow in p.u. (Array of Float)	Y	T	p^{fr}
`qdc_fr`	Reactive power flow, from bus in p.u. (Array of Float)	Y	T	q^{fr}
`qdc_to`	Reactive power flow, to bus in p.u. (Array of Float)	Y	T	q^{to}

2.9 Active Zonal Reserve — Requirements & Violation Costs

Input Attributes The attributes for zonal reserves used by the data format are listed below. Names, descriptions, and types for each of the attributes are shown.

Input attributes	Description (Type)	Req	Sec	Sym
uid	Zone reserve unique identifier (String)	Y	B	
REG_UP	Regulation reserve up requirement fraction (Float)	Y	S	σ^{rgu}
REG_DOWN	Regulation reserve down requirement faction (Float)	Y	S	σ^{rgd}
SYN	Synchronized reserve requirement fraction (Float)	Y	S	σ^{scr}
NSYN	Non-synchronized reserve requirement fraction (Float)	Y	S	σ^{nsc}
RAMPING_RESERVE_UP	Ramping reserve up requirement (Array of Float)	Y	T	$p^{rru,min}$
RAMPING_RESERVE_DOWN	Ramping reserve down requirement (Array of Float)	Y	T	$p^{rrd,min}$
REG_UP_vio_cost	Regulation reserve up violation cost in $/pu-hr (Float)	Y	S	c^{rgu}
REG_DOWN_vio_cost	Regulation reserve down violation cost in $/pu-hr (Float)	Y	S	c^{rgd}
SYN_vio_cost	Synchronized reserve violation cost in $/pu-hr (Float)	Y	S	c^{scr}
NSYN_vio_cost	Non-synchronized reserve violation cost in $/pu-hr (Float)	Y	S	c^{nsc}
RAMPING_RESERVE_UP_vio_cost	Flexible-ramp up violation cost in $/pu-hr (Float)	Y	S	c^{rru}
RAMPING_RESERVE_DOWN_vio_cost	Flexible-ramp down violation cost in $/pu-hr (Float)	Y	S	c^{rrd}

2.10 Reactive Zonal Reserve — Requirements & Violation Costs

Input Attributes The attributes for zonal reserves used by the data format are listed below. Names, descriptions, and types for each of the attributes are shown.

Input attributes	Description (Type)	Req	Sec	Sym
uid	Region reserve unique identifier (String)	Y	B	
REACT_UP	Reactive reserve power up requirement (Array of Float)	Y	T	$q^{qru,min}$
REACT_DOWN	Reactive reserve power down requirement (Array of Float)	Y	T	$q^{qrd,min}$
REACT_UP_vio_cost	Reactive reserve power violation cost in $/pu-hr (Float)	Y	S	c^{qru}
REACT_DOWN_vio_cost	Reactive reserve power violation cost in $/pu-hr (Float)	Y	S	c^{qrd}

2.11 Contingency

Input Attributes The attributes for each of the contingency data JSON object are listed below.

Contingency attributes	Description (Type)	Req	Sec	Sym
uid	Contingency unique identifier (String)	Y	C	
components	A JSON array, where each element is an unique identifier (string) for the contingency component (Array of String)	Y	C	

3 Static Electrical Parameter

This section presents the latest data format in JSON for each network electrical component for the competition. This section first starts by presenting the static Network object — the top level JSON object containing all the **static data** (attributes marked with 'S'/'B' in Section 2) for each network component. JSON objects associated with each individual component will be presented afterwards.

3.1 Network

The Network object in JSON consists of:

- A "general" object
  The object contains data specified for the general attributes in Section 2.1.
- A "violation_cost" object
  The object contains specifications for various violation costs, defined in 3.2.

- A "bus" array
  The object contains a JSON array of buses, defined in 3.3.
- A "shunt" array
  The object contains a JSON array of shunts, defined in 3.4.
- A "simple_dispatchable_device" array
  The object contains a JSON array of simple producing & consuming devices, defined in 3.5.
- A "ac_line" array
  The object contains a JSON array of AC lines, defined in 3.6.
- A "two_winding_transformer" array
  The object contains a JSON array of two winding transformers, defined in 3.7.
- A "dc_line" array
  The object contains a JSON array of DC lines, defined in 3.8.
- A "active_zonal_reserve" array
  The object contains a JSON array of active regional/zonal reserve requirements, defined in 3.9.
- A "reactive_zonal_reserve" array
  The object contains a JSON array of reactive regional/zonal reserve requirements, defined in 3.9.
- A "development" object
  The object contains development data for ARPA-E Dataset Teams. This section is primarily reserved for development. Competitors are not required to process the JSON object. The object may or may not be presented.

Below shows a high-level schematic description for the network object and the general object in JSON:

```
"network": {
  "general" : { "base_norm_mva"   : 100.0,
                "timestamp_start" : "2012-01-10T05:00:00",
                "timestamp_stop"  : "2012-01-11T05:00:00" },
  "violation_cost": { ... },
  "bus": [ ... ],
  "shunt": [ ... ],
  "simple_dispatchable_device": [ ... ],
  "ac_line": [ ... ],
  "two_winding_transformer": [ ... ],
  "dc_line": [ ... ],
  "active_zonal_reserve": [ ... ],
  "reactive_zonal_reserve": [ ... ],
  "development" : { ... }
}
```

3.2 Violation Cost

Below shows a schematic example on the static violation cost parameters for describing the costs/penalties for violating the bus active flows, bus reactive flows, and branch (thermal) MVA limits.

```
{
 "p_bus_vio_cost" : 1000000.0,
 "q_bus_vio_cost" : 1000000.0,
 "s_vio_cost" : 500.0
}
```

3.3 Bus

A bus represents a location point where equipment connected together. It could be a physical bus bar of a substation, or a connection point between various equipment, or a virtual location/split-point resulting from component aggregation or model transformation. Attributes for buses are listed in Section 2.3. Below shows a schematic example on static parameters for buses.

```
{
  "uid" : "IDB0001",
  "vm_ub" : 1.1,
  "vm_lb" : 0.95,
  "active_reserve_uids" : ["IDRES0001", "IDRES0002"],
  "reactive_reserve_uids" : ["IDRES0010"],
  "area" : "MI",
  "zone" : "DET",
  "longitude" : -83.045753,
  "latitude" : 42.331429,
  "city" : "Detroit",
  "county" : "Wayne",
  "state" : "Michigan",
  "country" : "USA",
  "con_loss_factor" : 1.0,
  "base_nom_volt" : 100.0,
  "type" : "PQ",
  "initial_status" : {
     "vm" : 1.05,
     "va" : 0.0013
  }
},
...
```

3.4 Shunt

Bus shunts are bus components for modeling losses to ground, and usually will be defined as an admittance to ground. Attributes for shunts are listed in Section 2.4. Below shows a schematic example on the static parameters for shunts.

```
{
    "uid" : "IDSH0001",
    "bus" : "IDB0002",
    "gs"  : 0.0,
    "bs"  : 0.001,
    "step_ub" : 2,
    "step_lb" : 0,
    "initial_status" : {
         "step" : 1
    }
},
...
```

3.5 Dispatchable Devices: Simple Producing & Consuming Devices

Producing devices produce active power and reactive power, and support voltage at target voltage set-point. These devices can be classical synchronous generators, renewable generations, or supporting power from other lumped/aggregated areas. Consuming devices consume active power and reactive power. These devices can be household demands, industrial loads, pump storage, or supporting power to other lumped/aggregated areas. For the competition, a producing/consuming device is classified as a simple device if it does not have switchable modes. Attributes for simple producing & consuming devices are listed in Section 2.5. Below shows a schematic example on the static parameters for simple producing & consuming devices.


```
{
    "uid" : "IDG0001",
    "bus" : "IDB0002",
    "device_type" : "producer",
    "description" : "gas",
    "vm_setpoint" : 1.2,
    "startup_cost" : 24.0,
    "startup_states" : [ [-20.0, 3.0], [-10.0, 8.0], [0.0, 24.0] ],
    "shutdown_cost" : 5.0,
    "startups_ub" : [ [0.0, 1.0, 1], [1.0, 3.0, 1] ],
    "energy_req_ub" : [ [0.0, 1.0, 2.0], [1.0, 3.0, 5.0] ],
    "energy_req_lb" : [ [0.0, 1.0, 0.0], [1.0, 3.0, 0.0] ],
    "on_cost" : 0.1,
    "in_service_time_lb" : 5.0, "down_time_lb" : 10.0,
    "p_ramp_up_ub" : 10.0, "p_ramp_down_ub" : 10.0,
    "p_startup_ramp_ub" : 5.0, "p_shutdown_ramp_ub" : 5.0,
    "initial_status": {
        "on_status": 0,
        "p" : 0.0,
        "q" : 0.0,
        "accu_down_time" : 10.0,
        "accu_up_time" : 0.0
    },
    "q_linear_cap" : 0,
    "q_bound_cap" : 1,
    "p_reg_res_up_ub" : 1.0, "p_reg_res_down_ub" : 1.0,
    "p_syn_res_ub" : 2.0, "p_nsyn_res_ub" : 4.0,
    "p_ramp_res_up_online_ub" : 2.0, "p_ramp_res_down_online_ub" : 2.0,
    "p_ramp_res_up_offline_ub" : 1.0, "p_ramp_res_down_offline_ub" : 0.0,
    "q_0_ub" : 5.0,
    "q_0_lb" : 0.0,
    "beta_ub" : 2.0,
    "beta_lb" : -0.5
},
...
```


3.6 AC Line

An AC line connects network devices in two different points of the network. It has has two sides: *from* (fr) side and the *to* side. In general, power flows are bi-directional. Readers should always refer to the formulation document for the +ve and -ve signs requirements for indicating directions in their solutions, and consulate the competition team if questions arise. Attributes for lines are listed in Section 2.6. Below shows a schematic example on the static parameters for lines.

Static component data example:

```
{
```

```
    "uid" : "IDAC0001",
    "fr_bus" : "IDB0021",
    "to_bus" : "IDB0025",
    "r" : 0.001, "x" : 0.05,
    "b" : 0.0001,
    "mva_ub_nom" : 10.0, "mva_ub_sht" : 15.0, "mva_ub_em" : 100.0,
    "connection_cost" : 0.01,
    "disconnection_cost" : 0.01,
    "initial_status" : {
        "on_status" : 1
    }
    "additional_shunt" : 1,
    "g_fr" : 0.0, "b_fr" : 0.01,
    "g_to" : 0.0, "b_to" : 0.0
}
...
```

3.7 Two Winding Transformer

A two winding transformer connects devices located at two different voltage levels of the network. Transformers are usually equipped with taps on their winding to adjust the voltage transformation or the phase angles through the transformer. Similar to AC lines, power flows are also bi-directional for transformers. Readers should always refer to the formulation document for the +ve and -ve signs requirements for indicating directions in their solutions, and consulate the competition team if questions arise. Attributes for two winding transformers are listed in Section 2.7. Below shows a schematic example on the static parameters for transformers.

Static component data example:

```
{
    "uid" : "ID2T0001",
    "fr_bus" : "IDB0021", "to_bus" : "IDB0025",
    "r" : 0.005, "x" : 0.01,
    "b" : 0.0001,
    "tm_lb" : 0.98, "tm_ub" : 1.02,
    "ta_lb" : 0.0, "ta_ub" : 0.0,
    "mva_ub_nom" : 100.0, "mva_ub_sht" : 250.0, "mva_ub_em" : 500.0,
    "connection_cost" : 0.1,
    "disconnection_cost" : 0.1,
    "initial_status" : {
          "tm" : 1.0, "ta" : 0.0,
          "on_status" : 1
    }
    "additional_shunt" : 1,
    "g_fr" : 0.0, "b_fr" : 0.0,
    "g_to" : 0.0, "b_to" : 0.01
},
...
```

3.8 DC Line

A DC line connects network devices at two different points of the network, with an AC-DC converter on each side. Power flowing through the DC line are first converted from AC to DC by a rectifier, before delivering power back to the network in AC by an inverter. For simplicity, the competition models DC lines as two generators on each side with active power coupling. Similar to an AC line, DC line has two sides — *from* (fr) side and *to* side. Attributes for DC lines are listed in Section 2.8. Below shows a schematic example on the static parameters for DC lines.

```
{
   "uid" : "IDDC0001",
   "fr_bus" : "IDB0009",
   "to_bus" : "IDB0010",
   "pdc_ub" : 1.0,
   "qdc_fr_lb" : -1.0, "qdc_fr_ub" : 1.0,
   "qdc_to_lb" : -0.5, "qdc_to_ub" : 0.5,
   "initial_status" : {
      "pdc_fr" : 1.0,
      "qdc_fr" : 1.0, "qdc_to" : 0.5
   }
},
...
```

3.9 Zonal Reserve

A reserve zone is a partition of the grid containing a collection of resources with individual reserve requirements. In practice, each reserve zone is designed to handle contingencies and daily regulation activities within the area. In the competition, we consider two types of reserve zones — active reserve zones and reactive reserve zones. Below shows a schematic example on the static parameters for the active reserve zones.

```
{
  "uid" : "IDRES0001",
  "REG_UP" : 0.03,
  "REG_DOWN" : 0.03,
  "SYN"  :  0.30,
  "NSYN" :  0.70,
  "REG_UP_vio_cost" : 1244.0,
  "REG_DOWN_vio_cost" : 1244.0,
  "SYN_vio_cost" : 305.0,
  "NSYN_vio_cost" : 24.0,
  "RAMPING_RESERVE_UP_vio_cost" : 0.10,
  "RAMPING_RESERVE_DOWN_vio_cost" : 0.10
},
...
```

Below shows a schematic example on the static parameters for the reactive reserve zones.

```
{
  "uid" : "IDRES0010",
  "REACT_UP_vio_cost" : 24.0,
  "REACT_DOWN_vio_cost" : 24.0
},
...
```

4 Time series input data

This section presents the latest data format in JSON for the time series input data for the competition. The section first presents the global time-series JSON object — the top level JSON object for storing all the time series data (attributes marked with 'T'/'B' in Section 2) for each network component. JSON objects associated with each individual component will be presented afterwards.

4.1 Time series input

The time series JSON object consists of:

- A `general` object
  The object contains general data listed in Section 2.1. For time series, these include the number of time periods (`time_periods`) and the duration between each of the time point (`interval_duration`).
- A `"simple_dispatchable_device"` array
  The object contains a JSON array of simple producing & consuming devices, defined in 4.2.
- A `"active_zonal_reserve"` array
  The object contains a list of active regional/zonal reserve requirements, defined in 4.3.
- A `"reactive_zonal_reserve"` array
  The object contains a list of reactive regional/zonal reserve requirements, defined in 4.3.
- A `"development"` object
  The object contains development data for ARPA-E Dataset Teams. This section is primarily reserved for development and competitors are not required to process the JSON object. The object may or may not be presented.

```
"time_series_input": {
   "general" : {"time_periods": 3, "interval_duration": [1.0, 1.0, 1.0]},
   "simple_dispatchable_device" : [ ... ],
   "active_zonal_reserve" : [ ... ],
   "reactive_zonal_reserve" : [ ... ],
   "development" : { ... }
}
```

4.2 Dispatchable Devices: Simple Producing & Consuming Devices

Below shows a schematic example on the time varying parameters for simple producing & consuming devices.

```
{
    "uid": "IDG0001",
    "on_status_ub" : [1, 1, 1],
    "on_status_lb" : [0, 0, 1],
    "p_lb" : [0.0, 0.0, 0.0], "p_ub" : [0.0, 5.0, 10.0],
    "q_lb" : [0.0, 0.0, 0.0], "q_ub" : [0.0, 5.0, 10.0],
    "cost" : [
              [[0.0, 0.0]],
              [[0.1, 5.0]],
              [[0.1, 5.0], [0.3, 5.0]]
             ],
    "p_reg_res_up_cost" : [1244.0, 1244.0, 0.0],
    "p_reg_res_down_cost" : [1244.0, 1244.0, 0.0],
    "p_syn_res_cost"    : [305.0, 305.0, 0.0],
    "p_nsyn_res_cost" : [24.0, 24.0, 0.0],
    "p_ramp_res_up_online_cost" : [0.1, 0.1, 0.1],
    "p_ramp_res_down_online_cost" : [0.1, 0.1, 0.1],
    "p_ramp_res_up_offline_cost" : [0.1, 0.1, 0.1],
```

```
    "p_ramp_res_down_offline_cost" : [0.1, 0.1, 0.1],
    "q_res_up_cost" : [24.0, 24.0, 0.0],
    "q_res_down_cost" : [24.0, 24.0, 0.0]
},
...
```

4.3 Regional Reserve

Below shows a schematic example on the time-varying parameters for the active reserve zones.

```
{
  "uid" : "IDRES0001",
  "RAMPING_RESERVE_UP" : [1.00, 1.00, 2.00],
  "RAMPING_RESERVE_DOWN" : [1.00, 1.00, 0.50]
},
...
```

Below shows a schematic example on the time-varying parameters for the reactive reserve zones.

```
{
  "uid" : "IDRES0010",
  "REACT_UP" : [0.01, 0.01, 0.005],
  "REACT_DOWN" : [0.01, 0.01, 0.005]
},
...
```

5 Reliability Parameters

This section presents the latest data format in JSON for the reliability input data. The JSON format for contingencies consists of a JSON array, where each JSON object within the array contains data for each contingency.

5.1 Reliability

The JSON object consists of:

- A `contingency` array
  The object contains the contingency data listed in Section 2.11.

Below shows a schematic example on the contingency input data parameters in the JSON data format.

```
"contingency": [
    {"uid" : "CONT1", "components" : [ "IDAC0001" ]},
    {"uid" : "CONT2", "components" : [ "IDAC0003" ]},
    {"uid" : "CONT3", "components" : [ "ID2T0001" ]}
]
```

Competitors are reminded that the format above may be more general than what is required for the competition. Competitors should always follow the formulation document and additional guidelines from the competition team.

6 Solution Output Data

This section presents the latest data format in JSON for the communicating the solution data back to the ARPA-E competition team. The section first presents the top level JSON object — the object for storing all the solution/output data for each network component. JSON objects associated with each individual component will be presented afterwards.

6.1 Time Series Output

The time series JSON object consists of:

- A "bus" array
  The object contains a JSON array of buses, defined in 6.2.
- A "shunt" array
  The object contains a JSON array of shunts, defined in 6.3.
- A "simple_dispatchable_device" array
  The object contains a JSON array of simple producing & consuming devices, defined in 6.4.
- A "ac_line" array
  The object contains a JSON array of AC lines, defined in 6.5.
- A "two_winding_transformer" array
  The object contains a JSON array of two winding transformers, defined in 6.6.
- A "dc_line" array
  The object contains a JSON array of DC lines, defined in 6.7.

```
"time_series_output": {
   "bus": [ ... ],
   "shunt": [ ... ],
   "simple_dispatchable_device": [ ... ],
   "ac_line": [ ... ],
   "two_winding_transformer": [ ... ],
   "dc_line": [ ... ]
}
```

6.2 Bus

Attributes for buses are listed in Section 2.3. Below shows a schematic example on output parameters for buses.

```
{
  "uid" : "IDB0001",
  "vm" : [1.0, 0.99, 0.98],
  "va" : [0.0, 0.0125, 0.0233]
},
...
```

6.3 Shunt

Attributes for shunts are listed in Section 2.4. Below shows a schematic example on the output parameters for shunts.

```
{
    "uid" : "IDSH0001",
    "step" : [1, 1, 0]
},
...
```

6.4 Dispatchable Devices: Simple Producing & Consuming Devices

Attributes for simple producing & consuming devices are listed in Section 2.5. Below shows a schematic example on the output parameters for the simple devices.

```
{
    "uid" : "IDG0001",
    "on_status" : [1, 1, 1],
    "p_on" : [0.0, 2.0, 7.0],
    "q"    : [0.0, 1.0, 4.0],
    "p_reg_res_up" : [0.0, 0.5, 0.5],
    "p_reg_res_down" : [0.0, 0.5, 1.0],
    "p_syn_res" : [0.0, 0.5, 1.0],
    "p_nsyn_res" : [0.0, 0.5, 1.0],
    "p_ramp_res_up_online" : [0.0, 1.0, 0.5],
    "p_ramp_res_down_online" : [0.0, 1.0, 2.0],
    "p_ramp_res_up_offline" : [1.0, 0.0, 0.0],
    "p_ramp_res_down_offline" : [0.0, 0.0, 0.0],
    "q_res_up" : [0.0, 1.0, 2.0],
    "q_res_down" : [0.0, 1.0, 2.0]
},
...
```

6.5 AC Transmission Line

Attributes for AC transmission lines are listed in Section 2.6. Below shows a schematic example on the output parameters for lines.

```
{
    "uid" : "IDAC0001",
    "on_status" : [0, 1, 1]
},
...
```

6.6 Two Winding Transformer

Attributes for two winding transformers are listed in Section 2.7. Below shows a schematic example on the output parameters for transformers.

```
{
    "uid" : "ID2T0001",
    "tm"    : [1.0, 1.05, 1.0],
    "ta"    : [0.0, 0.0, 0.0],
    "on_status" : [1, 1, 0]
},
...
```

6.7 DC Line

Attributes for DC lines are listed in Section 2.8. Below shows a schematic example on the output parameters for DC lines.

```
{
   "uid" : "IDDC0001",
   "pdc_fr" : [0.5, 0.6, 0.7],
   "qdc_fr" : [0.0, 0.0, 0.1],
   "qdc_to" : [0.0, 0.0, -0.1]
},
...
```

7 Data Format Parsing & Details

This section presents data input parsing and solution writing instructions for extracting data and storing the solutions for the Challenge 3 competition.

7.1 Input Data File

Each input/problem data file is a JSON object that will include:

- the `network` data object (Section 3);
  The object containing static network information
- the `time_series_input` data object (Section 4)
  The object containing timeseries data
- the `reliability` data object (Section 5)
  The object containing contingency information

in JSON format. Below shows a high-level schematic structure for the input/problem data file:

```
{
    "network": {
    ...
    },
    "time_series_input": {
    ...
    },
    "reliability": {
        ...
    }
}
```

7.2 Output Data File

Each output/solution data file computed by competitors should be a JSON object consists of only the solution output data (Section 6).

It should only include:

- the `time_series_output` data object (Section 6)
  The object containing timeseries data

```
{
    "time_series_solution": {
        ...
    }
}
```

The Competition Team may request additional information (e.g., for debugging / communication purposes) from competitors and request information to be placed in additional fields in the solution file other than `solution_output`.

7.3 Numerical Accuracy & Precision

Programming packages/libraries handling JSON may have various different default/pre-set numerical precision when reading or writing floating point numbers. We recommend competitors to double check the default precision, and change the precision if needed when writing any floating point values to the output/solution file. We also recommend competitors to use at least double precision (64 bits) when reading non-integer values from JSON inputs.

8 Change Log

List of updates for: v1.0.4

- Section 2.1 (page 4/top): Rephrase the description for the General JSON object attributes. Previous format versions describe these attributes incorrectly as global attributes.
- Section 2.5 (page 7/top): Rephrase the description for the cost attribute for dispatchable devices. Previous format versions incorrectly describe the number of array levels for the attribute.

List of updates for: v1.0.3

- Section 2 (page 3/bottom): Clarification on optional attributes are added.
- Section 2 (page 4/top): Clarification on inner JSON objects are added.
- Section 5.1 (page 18/bottom): Clarification on reliability JSON object.
- Section 6.1 (page 19/top): Formal description on the time series output format are added.