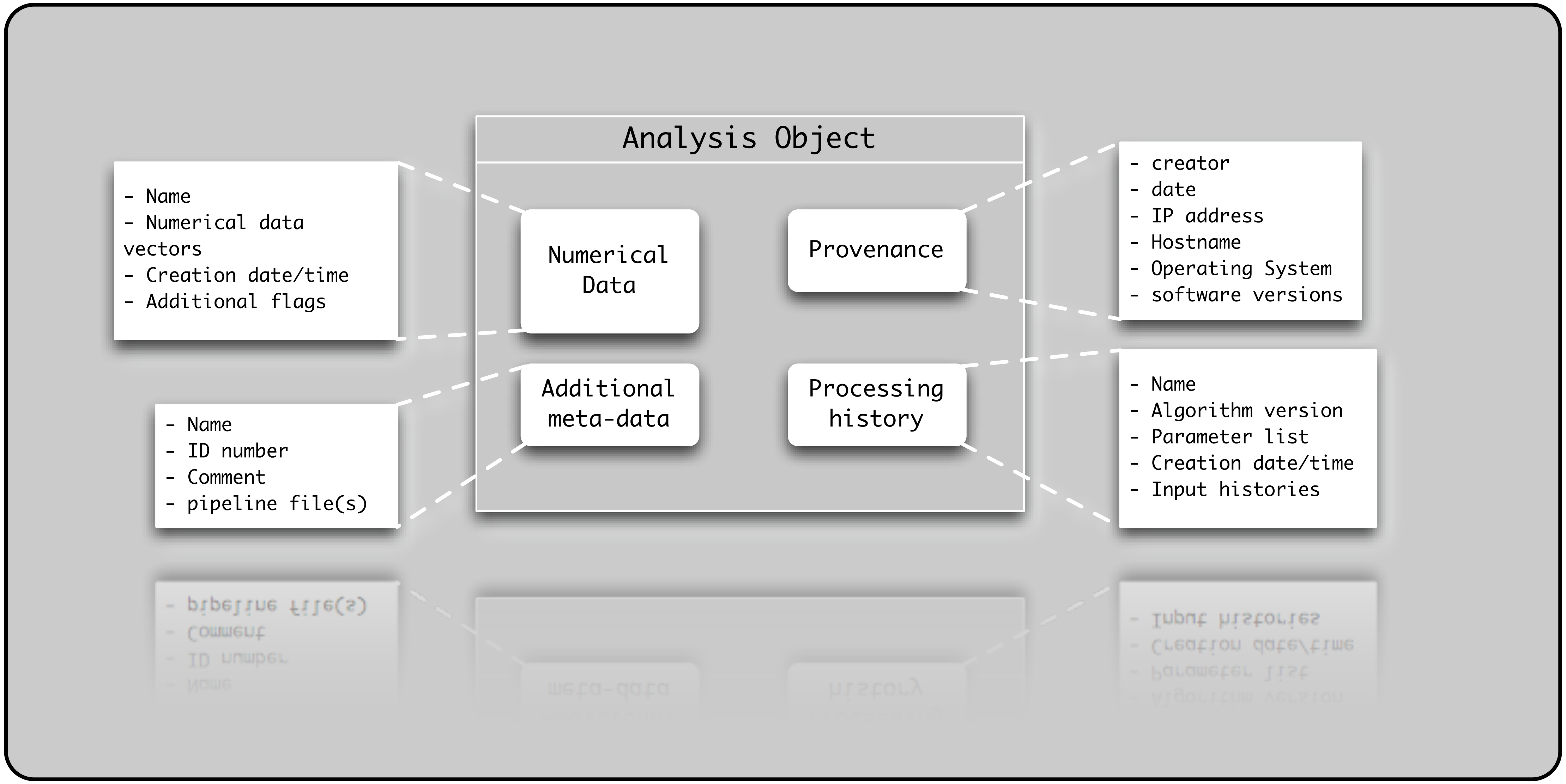
The First LTPDA Training Session - Topic 1

M Hewitson

```
+----------------------------------------------------+
|                                                    |
|                                                    |
|                        ****                        |
|                         **                         |
|                    -------------                   |
|                ////       /     \\\\               |
|             ///          /          \\\            |
|            |            /              |           |
|        ** |    +----+  /      +----+    | **       |
|        ***|    |    |//-------|    |    |***       |
|        ** |    +----+        /+----+    | **       |
|            |               /           |           |
|             \\\           /         ///            |
|                \\\\     //      ////               |
|                    -------------                   |
|                         **                         |
|                        ****                        |
|                                                    |
|          Welcome to LTPDA Toolbox                  |
|                                                    |
|                 Version: 1.9.3                     |
|                 Release: (R2008b)                  |
|                    Date: 19-10-08                  |
|                                                    |
+----------------------------------------------------+
```

Introducing Analysis Objects (AOs)

- An AO contains more than just data



Analysis Object

Numerical Data

- Name
- Numerical data vectors
- Creation date/time
- Additional flags

Provenance

- creator
- date
- IP address
- Hostname
- Operating System
- software versions

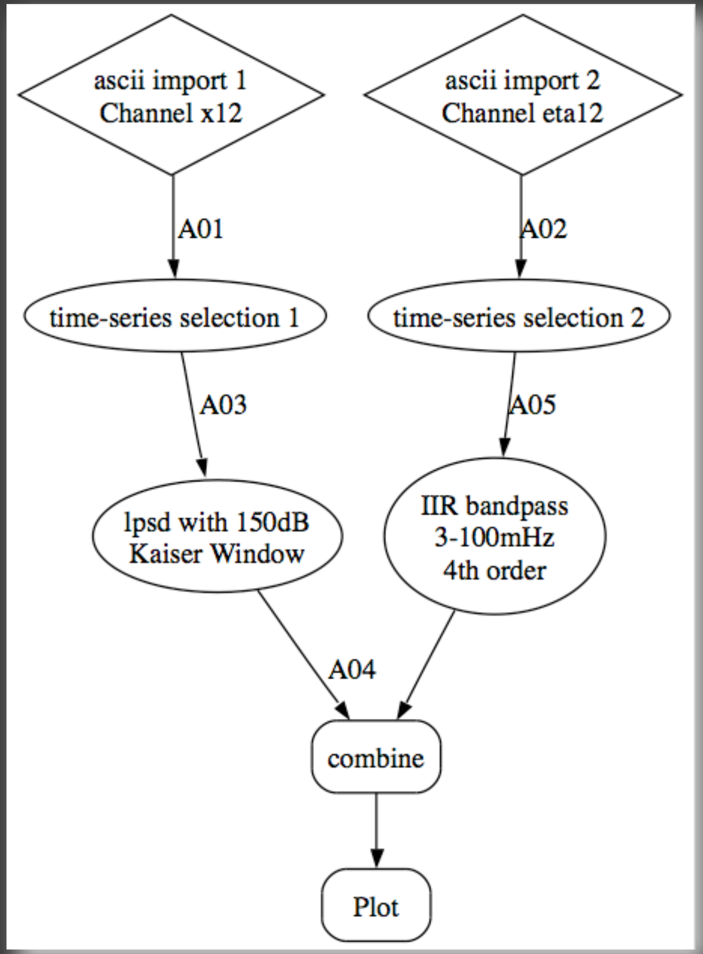
Additional meta-data

- Name
- ID number
- Comment
- pipeline file(s)

Processing history

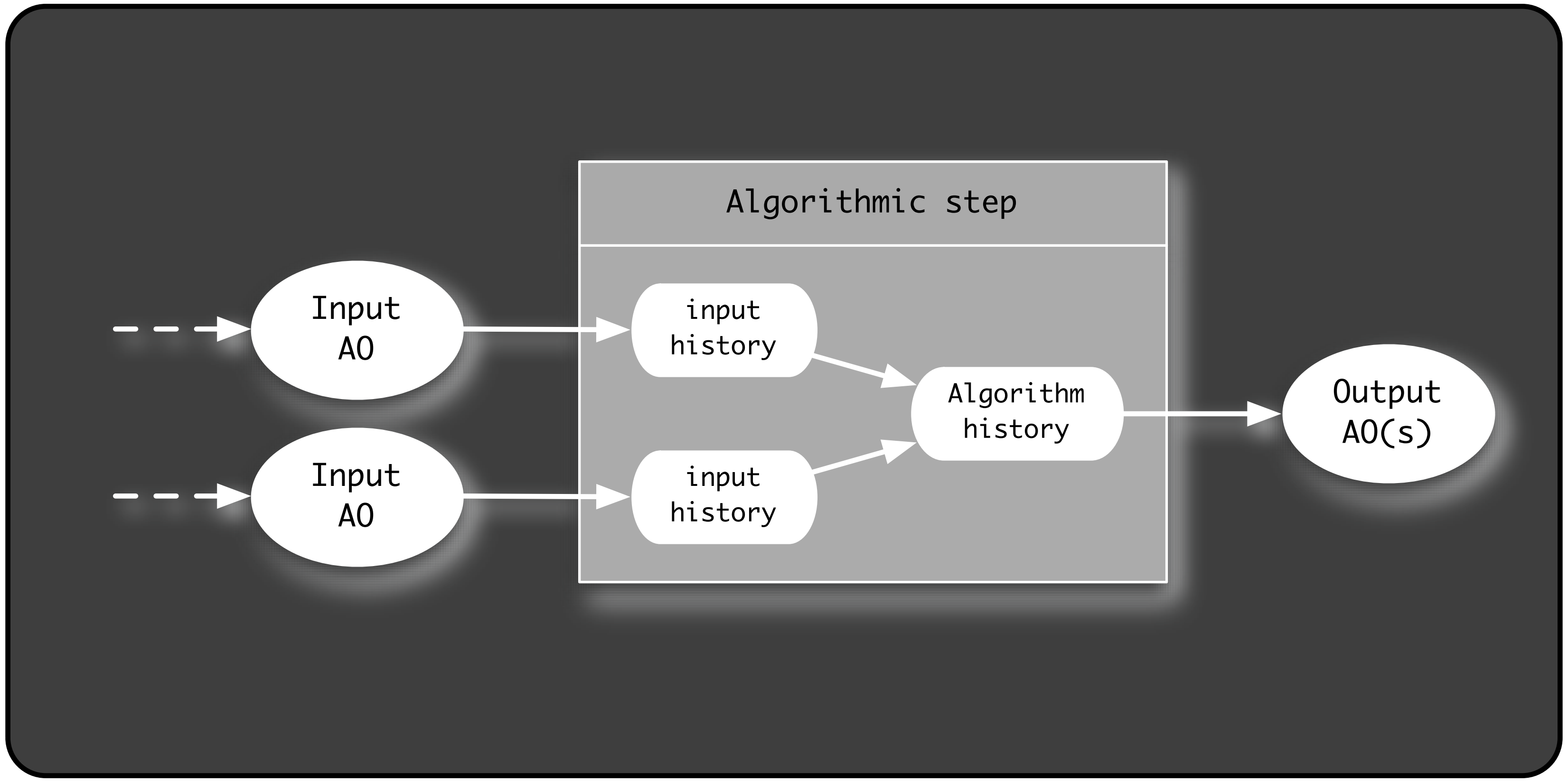
- Name
- Algorithm version
- Parameter list
- Creation date/time
- Input histories

History



traceable results

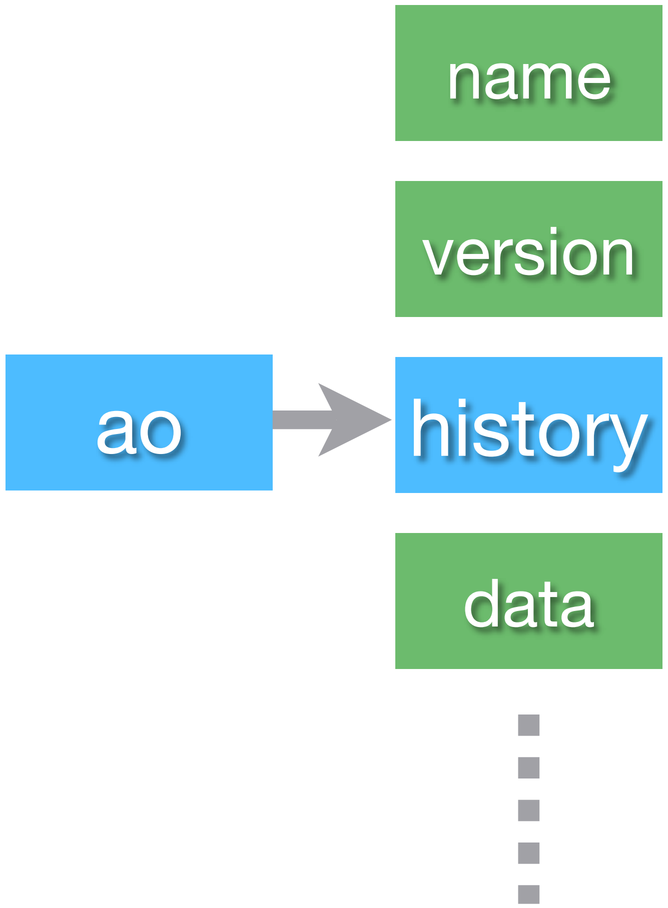
Intelligent algorithms



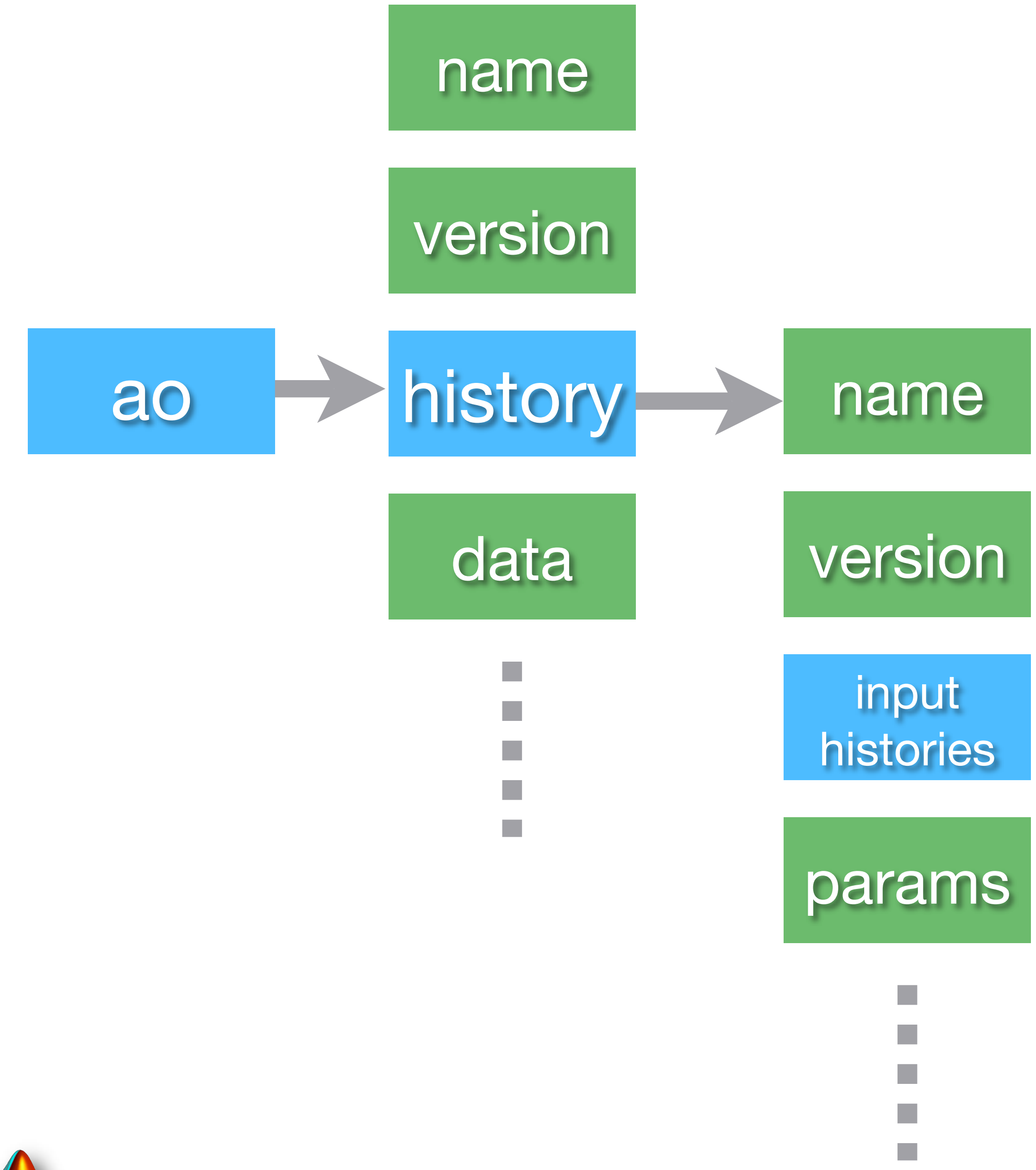
History tree

ao

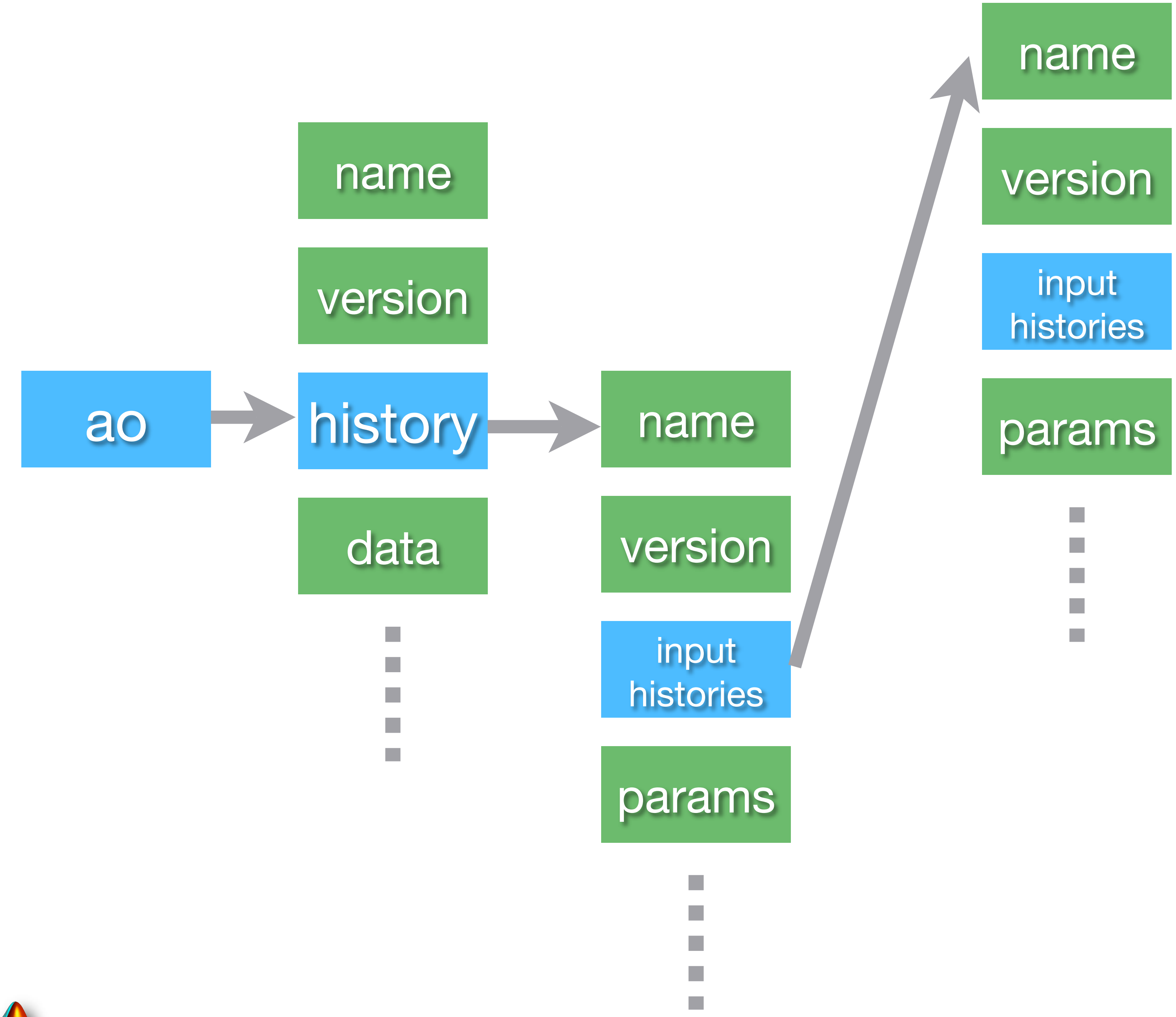
History tree



History tree



History tree



ao

name

version

history

data

name

version

input histories

params

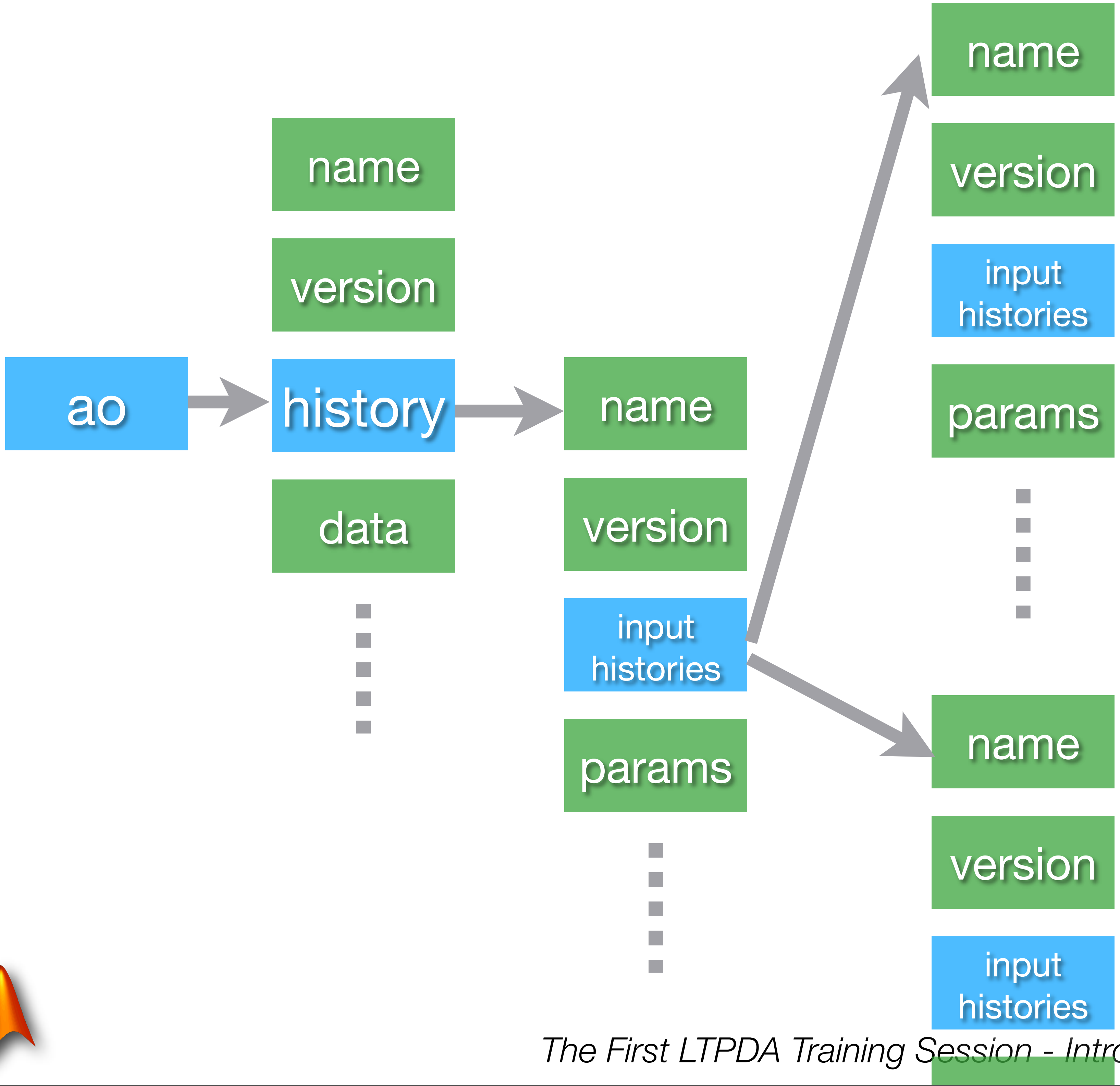
name

version

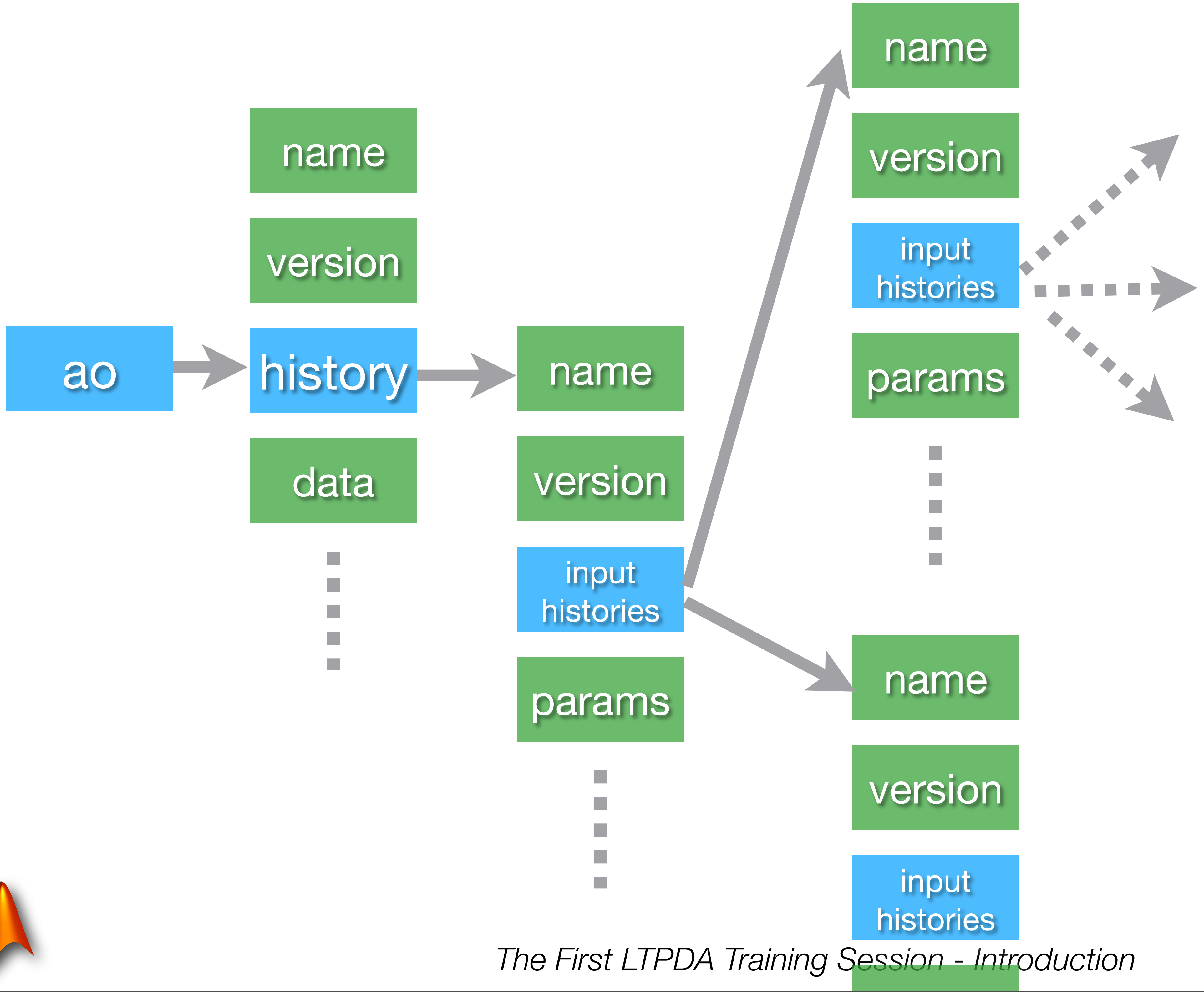
input histories

params

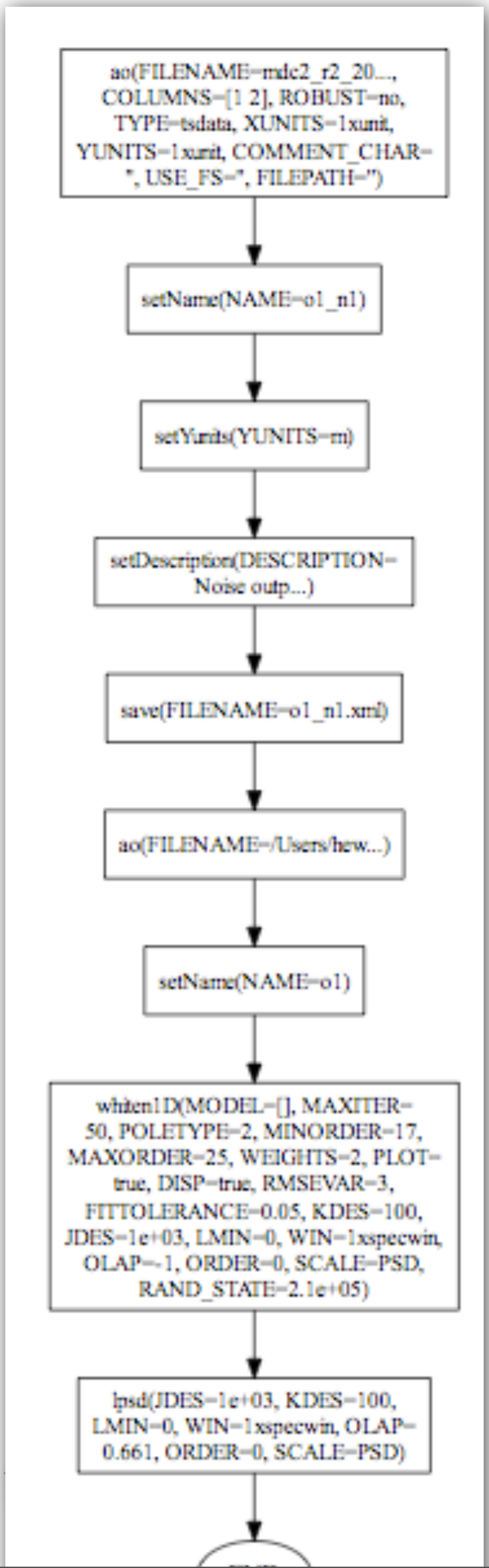
History tree

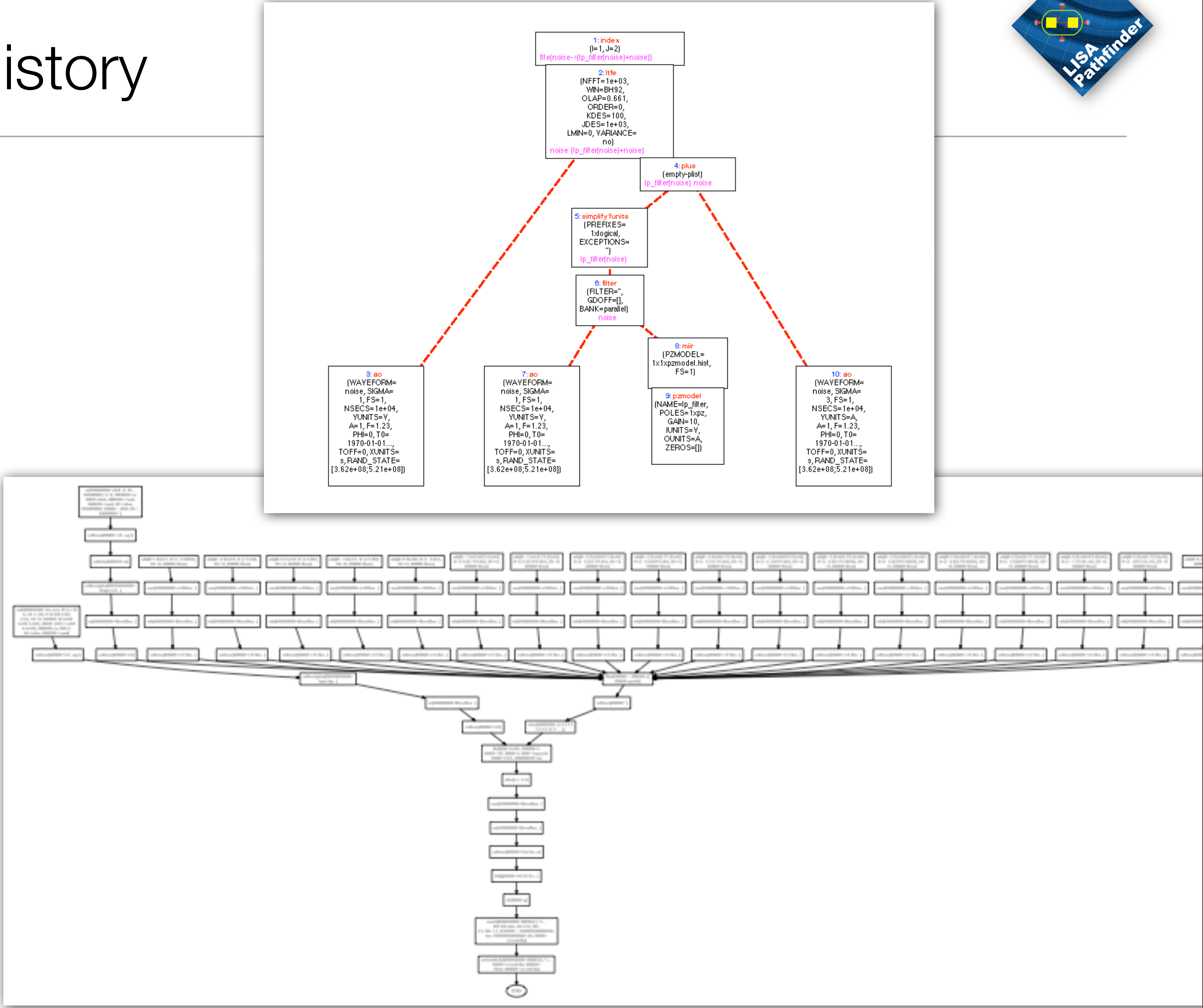


History tree



Viewing history


ao(FILENAME=mdc2_r2_20..., COLUMNS=[1 2], ROBUST=no, TYPE=tsdata, XUNITS=1xunit, YUNITS=1xunit, COMMENT_CHAR=", USE_FS=", FILEPATH=")
setName(NAME=o1_n1)
setYunits(YUNITS=m)
setDescription(DESCRIPTION=Noise outp...)
save(FILENAME=o1_n1.xml)
ao(FILENAME=/Users/hew...)
setName(NAME=o1)
whiten1D(MODEL=[], MAXITER=50, POLETYPE=2, MINORDER=17, MAXORDER=25, WEIGHTS=2, PLOT=true, DISP=true, RMSEVAR=3, FITTOLERANCE=0.05, KDES=100, JDES=1e+03, LMIN=0, WIN=1xspecwin, OLAP=-1, ORDER=0, SCALE=PSD, RAND_STATE=2.1e+05)
lpsd(JDES=1e+03, KDES=100, LMIN=0, WIN=1xspecwin, OLAP=0.661, ORDER=0, SCALE=PSD)


1: index
(I=1, J=2)
ltfe(noise->(lp_filter(noise)+noise))
2: ltfe
(NFFT=1e+03, WIN=BH92, OLAP=0.661, ORDER=0, KDES=100, JDES=1e+03, LMIN=0, VARIANCE=no)
noise (lp_filter(noise)+noise)
4: plus
(empty-plist)
lp_filter(noise) noise
5: simplifyYunits
(PREFIXES=1xlogical, EXCEPTIONS='')
lp_filter(noise)
6: filter
(FILTER='', GDOFF=[], BANK=parallel)
noise
8: miir
(PZMODEL=1x1xpzmodel.hist, FS=1)
9: pzmodel
(NAME=lp_filter, POLES=1xpz, GAIN=10, IUNITS=V, OUNITS=A, ZEROS=[])
3: ao
(WAVEFORM=noise, SIGMA=1, FS=1, NSECS=1e+04, YUNITS=V, A=1, F=1.23, PHI=0, T0=1970-01-01..., TOFF=0, XUNITS=s, RAND_STATE=[3.62e+08;5.21e+08])
7: ao
(WAVEFORM=noise, SIGMA=1, FS=1, NSECS=1e+04, YUNITS=V, A=1, F=1.23, PHI=0, T0=1970-01-01..., TOFF=0, XUNITS=s, RAND_STATE=[3.62e+08;5.21e+08])
10: ao
(WAVEFORM=noise, SIGMA=3, FS=1, NSECS=1e+04, YUNITS=A, A=1, F=1.23, PHI=0, T0=1970-01-01..., TOFF=0, XUNITS=s, RAND_STATE=[3.62e+08;5.21e+08])

Reliving history

```
obj.type(<file>)
```

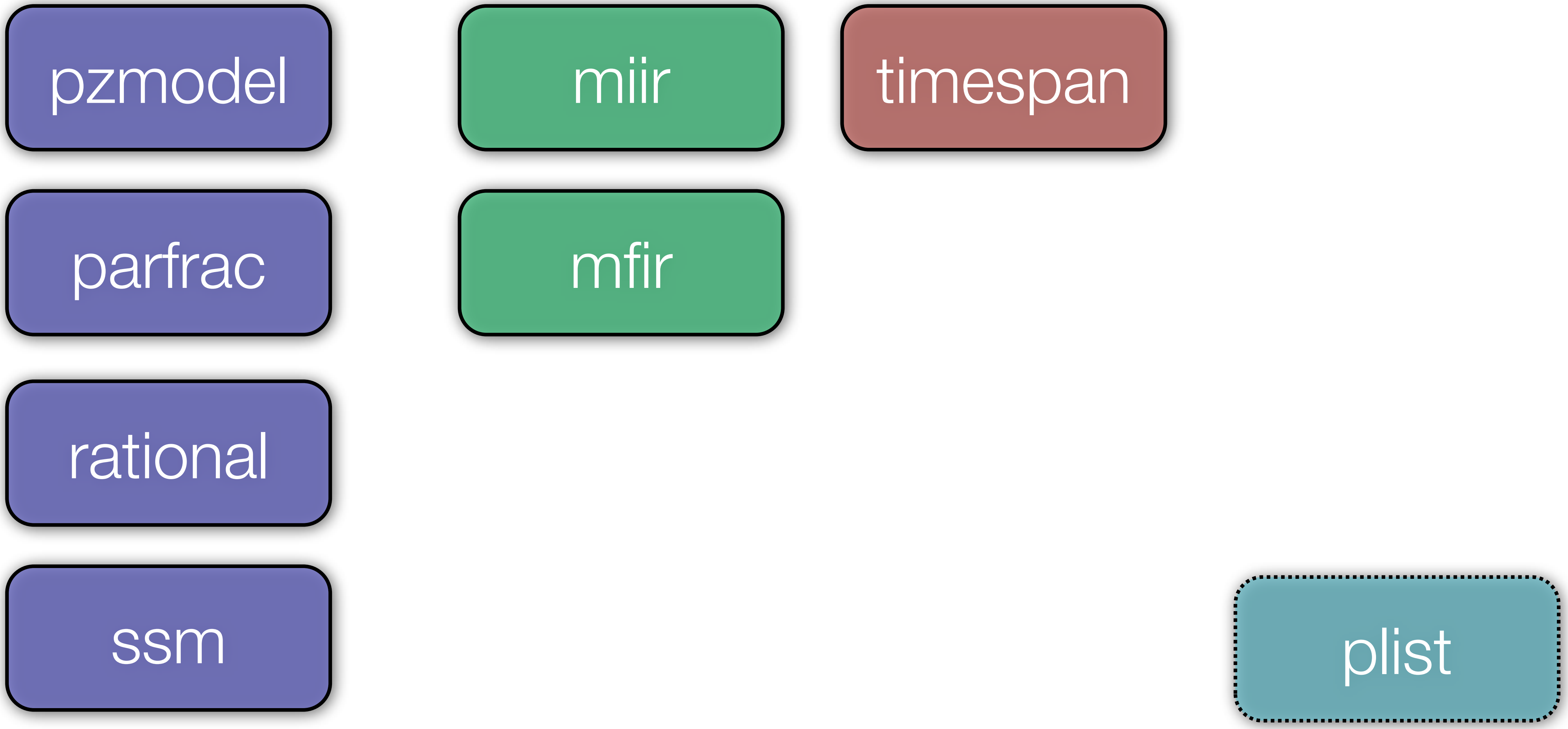
output commands needed to rebuild this object

```
robj = obj.rebuild
```

rebuild this object

LTPDA User Objects

- Not only AOs behave in this way:



pzmodel

parfrac

rational

ssm

miir

mfir

timespan

plist

Object-oriented programming for beginners

Object-oriented programming for beginners

Car

Object-oriented programming for beginners

Car

Properties:

Object-oriented programming for beginners

Car

Properties: color

Object-oriented programming for beginners

Car

Properties: color
engine size

Object-oriented programming for beginners

Car

Properties: color

engine size

cost

Object-oriented programming for beginners

Car

Properties: color

engine size

cost

Methods:

Object-oriented programming for beginners

Car

Properties: color
engine size
cost

Methods: start

Object-oriented programming for beginners

Car

Properties:	color
	engine size
	cost
Methods:	start
	stop

Object-oriented programming for beginners

Car

Properties:	color
	engine size
	cost
Methods:	start
	stop
	drive

Object-oriented programming for beginners

Car

Properties: color
engine size
cost

Methods: start
stop
drive

object = instance of a class

Object-oriented programming for beginners

Car

Properties: color

engine size

cost

Methods: start

stop

drive

object = instance of a class

```
car1 = Car(‘blue’)
car2 = Car(‘red’)
```

Object-oriented programming for beginners

Car

Properties: color

engine size

cost

Methods: start

stop

drive

object = instance of a class

```
car1 = Car('blue')
car2 = Car('red')
```

objects

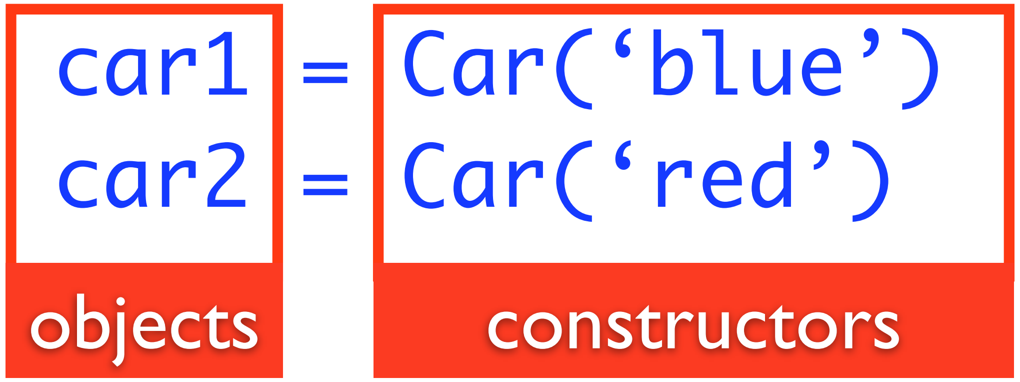
Object-oriented programming for beginners

Car

Properties: color
engine size
cost

Methods: start
stop
drive

object = instance of a class



```
car1 = Car('blue')
car2 = Car('red')
```

Object-oriented programming for beginners

Car

Properties: color
engine size
cost

Methods: start
stop
drive

object = instance of a class

```
car1 = Car('blue')
car2 = Car('red')
```

objects

constructors

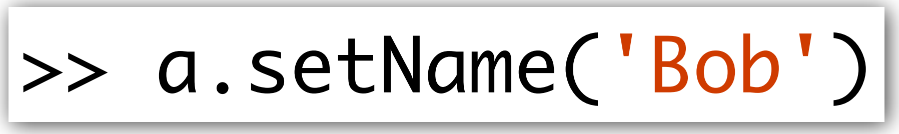
```
car1.start
car1.drive
stop(car1)
```

Exercise 1

- Now it’s your turn
- Open the MATLAB documentation
  - In the MATLAB terminal
    - >> doc
  - “Help -> Product Help>”
- Go to section:
  - LTPDA Toolbox
    - LTPDA Training Session 1
      - Topic 1
        - Making AOs
- And work your way through
  - “Exercise 1 - Your First Analysis Object”

Exercise 2 - setting object properties

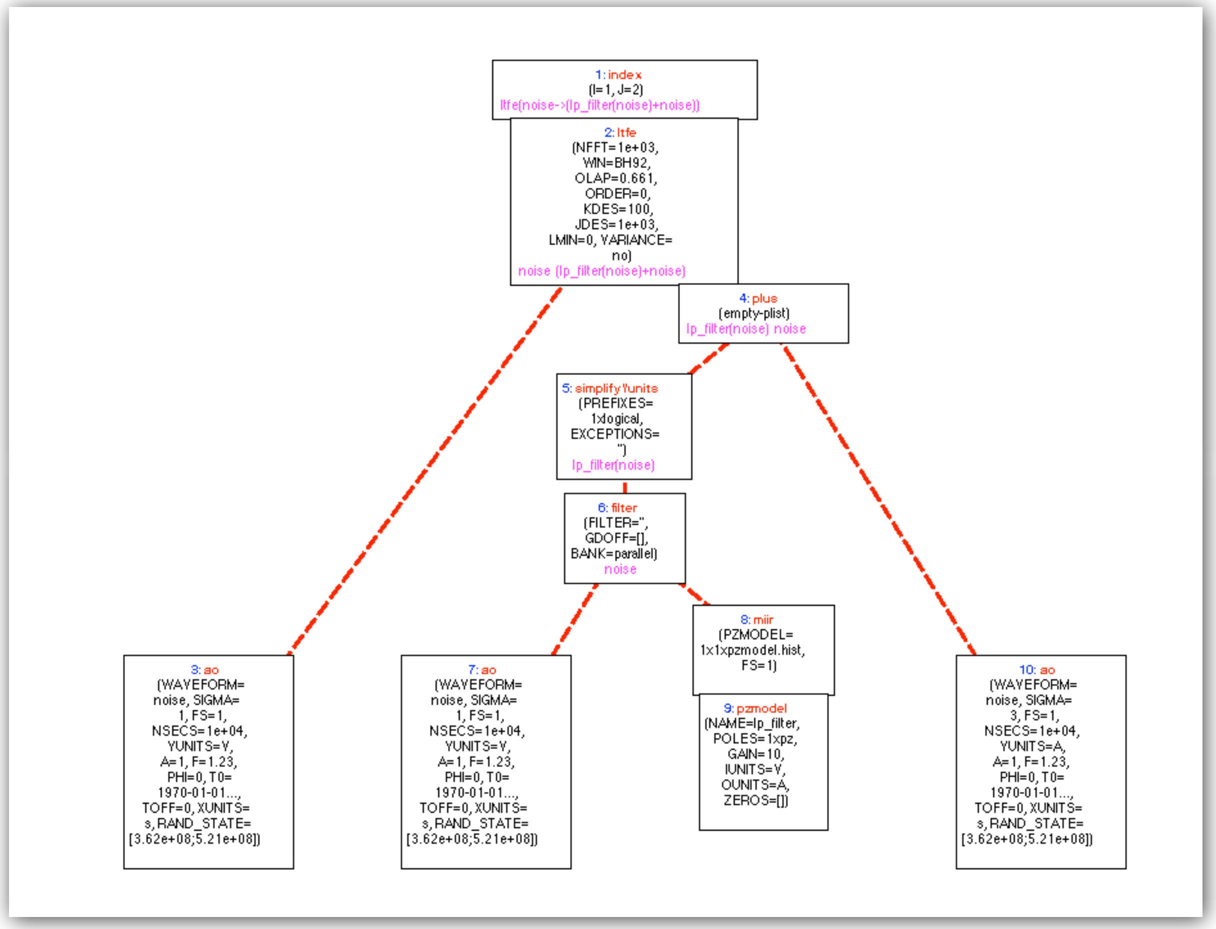
- AOs have properties, e.g.,
  - name
  - description
- Work through:
  - Exercise 2 - Setting properties of AOs



```
>> a.setName('Bob')
```

Exercise 3 - Viewing the history

- We saw earlier that AOs store their processing history.
- Here we learn how to view that history
- Work through the section
  - Exercise 3 - Viewing the history


1: index (I=1, J=2) ltfe(noise->(lp_filter(noise)+noise))
2: ltfe (NFFT=1e+03, WIN=BH92, OLAP=0.661, ORDER=0, KDES=100, JDES=1e+03, LMIN=0, VARIANCE= no) noise (lp_filter(noise)+noise)
4: plus (empty-plist) lp_filter(noise) noise
5: simplifyYunits (PREFIXES= 1xlogical, EXCEPTIONS= '') lp_filter(noise)
6: filter (FILTER='', GDOFF=[], BANK=parallel) noise
8: miir (PZMODEL= 1x1xpzmodel.hist, FS=1)
9: pzmodel (NAME=lp_filter, POLES=1xpz, GAIN=10, IUNITS=V, OUNITS=A, ZEROS=[])
3: ao (WAVEFORM= noise, SIGMA= 1, FS=1, NSECS=1e+04, YUNITS=V, A=1, F=1.23, PHI=0, T0= 1970-01-01..., TOFF=0, XUNITS= s, RAND_STATE= [3.62e+08;5.21e+08])
7: ao (WAVEFORM= noise, SIGMA= 1, FS=1, NSECS=1e+04, YUNITS=V, A=1, F=1.23, PHI=0, T0= 1970-01-01..., TOFF=0, XUNITS= s, RAND_STATE= [3.62e+08;5.21e+08])
10: ao (WAVEFORM= noise, SIGMA= 3, FS=1, NSECS=1e+04, YUNITS=A, A=1, F=1.23, PHI=0, T0= 1970-01-01..., TOFF=0, XUNITS= s, RAND_STATE= [3.62e+08;5.21e+08])

Exercise 4 - Making Time-series AOs

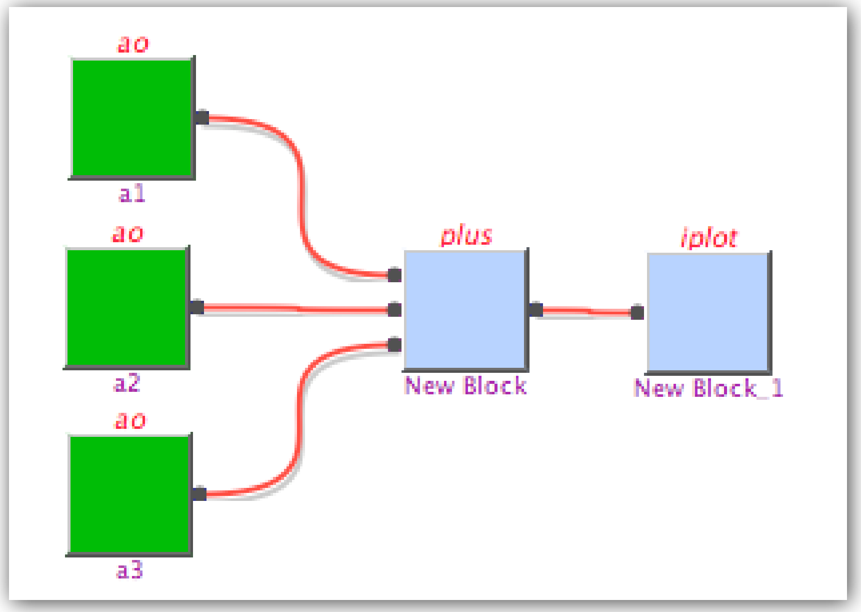
- AOs can contain different types of data
  - Time-series data are stored in a `tsdata` object
  - They also have properties:

tsdata	
t0	Absolute time-stamp of first sample
xunits	X-axis units
yunits	Y-axis units

- Work through section:
  - Topic 1
    - Making a time-series AO

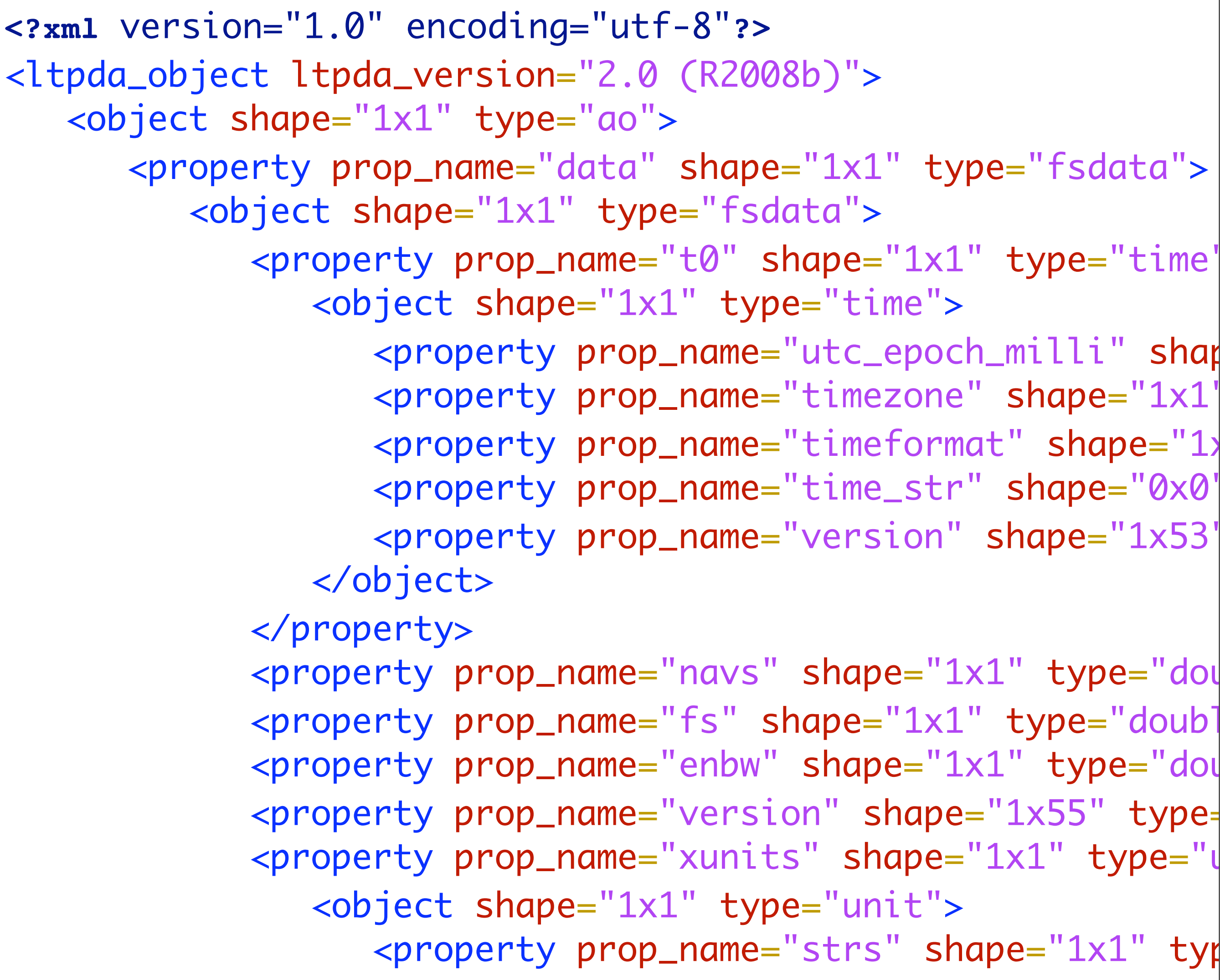
Basic Math

- You can operate on AOs using a large set of methods
  - In particular, many typical Math operations are available (overloaded)
    - Further details at: http://www.lisa.aei-hannover.de/ltpda/documents/files/operator_rules.pdf
- Work through the help section
  - Topic 1
    - Basic Math with AOs



Saving and loading AOs

- All LTPDA User Objects can be saved to (and loaded from), file in
  - XML format
  - binary MAT format
- Work through help section
  - Topic 1
    - Saving and loading AOs



```
<?xml version="1.0" encoding="utf-8"?>
<ltpda_object ltpda_version="2.0 (R2008b)">
  <object shape="1x1" type="ao">
    <property prop_name="data" shape="1x1" type="fsdata">
      <object shape="1x1" type="fsdata">
        <property prop_name="t0" shape="1x1" type="time"
          <object shape="1x1" type="time">
            <property prop_name="utc_epoch_milli" shap
            <property prop_name="timezone" shape="1x1"
            <property prop_name="timeformat" shape="1x
            <property prop_name="time_str" shape="0x0"
            <property prop_name="version" shape="1x53"
          </object>
        </property>
        <property prop_name="navs" shape="1x1" type="dou
        <property prop_name="fs" shape="1x1" type="doubl
        <property prop_name="enbw" shape="1x1" type="dou
        <property prop_name="version" shape="1x55" type=
        <property prop_name="xunits" shape="1x1" type="u
          <object shape="1x1" type="unit">
            <property prop_name="strs" shape="1x1" typ
```

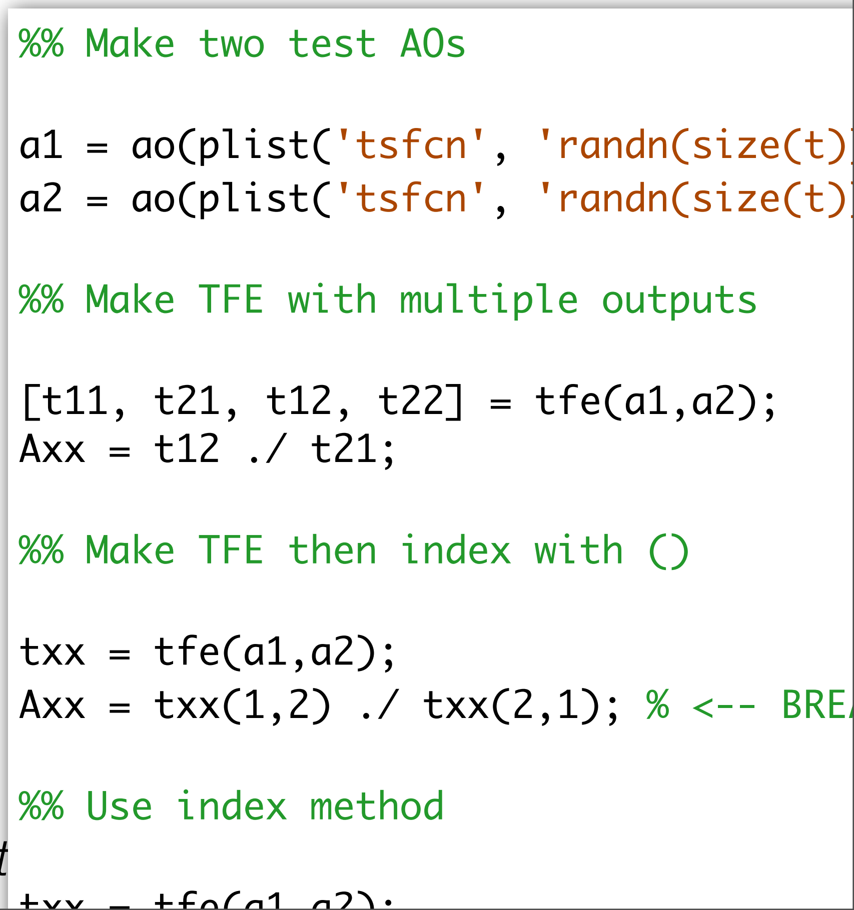
Reading data files

- You can construct AOs from existing ASCII (raw) data files
- Work through help section
    - Topic 1
        - Constructing AOs from data files

```
a = ao('topic1/simpleASCII.txt')
```

Writing LTPDA scripts

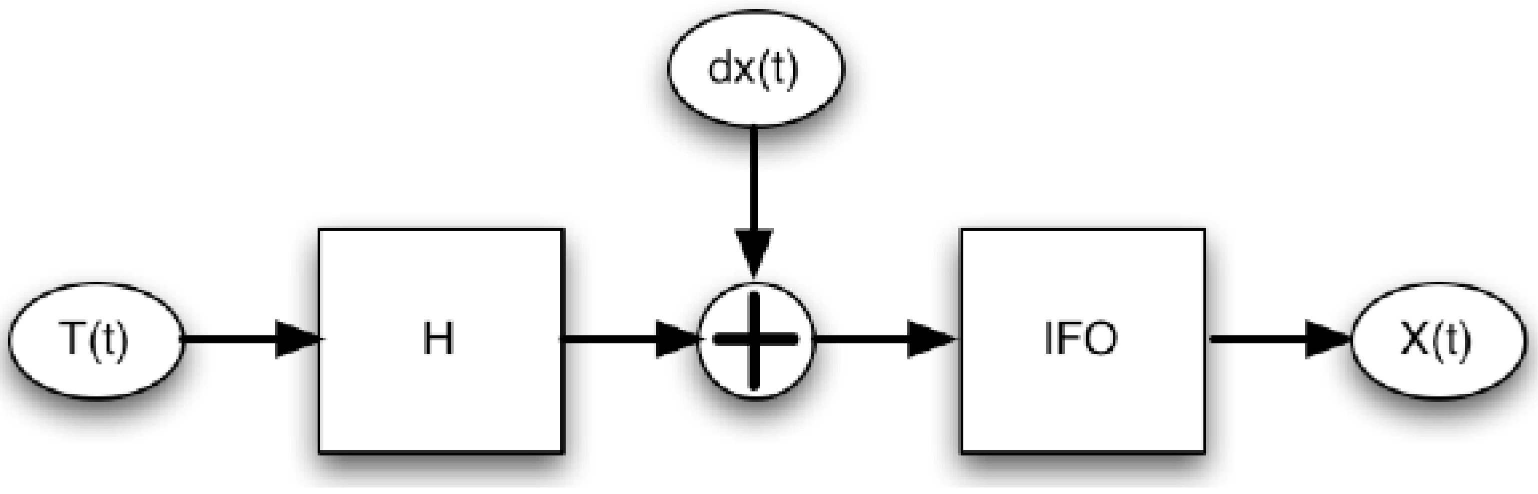
- So far we’ve done everything on the command-line or on the workbench
- Now we look at writing LTPDA scripts
  - There are some subtle differences to standard MATLAB scripts and some recommended practices
- Read through help section
  - Topic 1
    - Writing LTPDA scripts
- ** More details on the `tfe` method later



```
%% Make two test AOs

a1 = ao(plist('tsfcn', 'randn(size(t)
a2 = ao(plist('tsfcn', 'randn(size(t)

%% Make TFE with multiple outputs

[t11, t21, t12, t22] = tfe(a1,a2);
Axx = t12 ./ t21;

%% Make TFE then index with ()

txx = tfe(a1,a2);
Axx = txx(1,2) ./ txx(2,1); % <-- BRE

%% Use index method

txx = tfe(a1,a2);
```

IFO/Temperature example

- We have a data analysis exercise which will develop fully over the course of the training session
- This is the first part: reading and preparing the data
- Work through help section
  - Topic 1
    - IFO/Temperature Example - Introduction


dx(t)
T(t)
H
IFO
X(t)