# TableToLongForm
## Working with Modules

*Jimmy Oh*

Department of Statistics
University of Auckland

## 1 Introduction

TableToLongForm is partially modular and can be extended in some ways with external *modules*. This is done by *registering* external modules with TableToLongForm and then instructing TableToLongForm to use the new modules via the following optional arguments: `IdentPrimary`, `IdentAuxiliary`, `ParePreRow` and `ParePreCol`.

**Ident Primary** The Primary Ident algorithm, of which one is chosen. They should take a single argument, `matFull`. They should return an `IdentResult`.

Default: `IdentPrimary = "combound"`

**Ident Auxiliary** Auxiliary Ident algorithms, of which any combination, in any order, can be chosen. They are called after the Primary algorithm, to refine the `IdentResult`. They should take two arguments, `matFull` and `IdentResult`. They should return an `IdentResult`.

Default: `IdentAuxiliary = "sequence"`

**Pare Pre Row** Pre-requisite algorithms that tidy up the Row Labels for correct operation of the Main Parentage algorithm. Any combination of these algorithms, in any order, can be chosen. The current implementation of TableToLongForm has no Pre Row algorithms. They should take two arguments, `matData` and `matRowLabel`. They should return a named list containing two elements, `matData` and `matRowLabel`.

Default: `ParePreRow = NULL`

**Pare Pre Col** Pre-requisite algorithms that tidy up the Column Labels for correct operation of the Main Parentage algorithm. Any combination of these algorithms, in any order, can be chosen. They should take two arguments, `matData` and `matColLabel`. They should return a named list containing two elements, `matData` and `matColLabel`.

Default: `ParePreCol = c("mismatch", "misalign", "multirow")`

## 2 Vocabulary

| | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| 1 | | | Column 1 | Column 2 | Column 3 | Column 4 |
| 2 | Row Parent1 | Row Child1 | 10 | 20 | 30 | 40 |
| 3 | | Row Child2 | 11 | 21 | 31 | 41 |
| 4 | Row Parent2 | Row Child1 | 12 | 22 | 32 | 42 |
| 5 | | Row Child2 | 13 | 23 | 33 | 43 |

**matFull** The entire Table.

**matRowLabel** Blue region.

**matColLabel** Green region.

**matData** Red region.

**IdentResult** A list containing these two elements:

**label** - a vector of the rows or columns where the labels are found.

**data** - a vector of the rows or columns where the data are found.

For this Table:

```
IdentResult = list(rows = list(label = 1,   data = 2:5),
                   cols = list(label = 1:2, data = 3:6))
```

# 3 Workflow

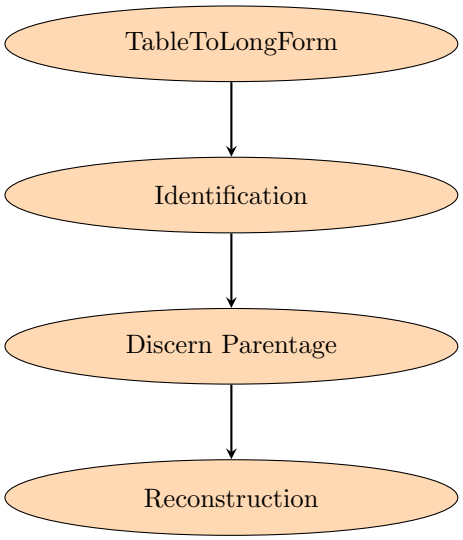


Figure 1: The basic workflow of TableToLongForm. There are 3 stages in the conversion process:

1. Identification - identify where in the Table the data is found and where the accompanying labels are, while ignoring any extraneous information we do not want.
2. Parentage - understand the hierarchical structure (the *parentage*) of the row and column labels.
3. Reconstruction - use the information from the first two stages to reconstruct the Table as a Dataframe.

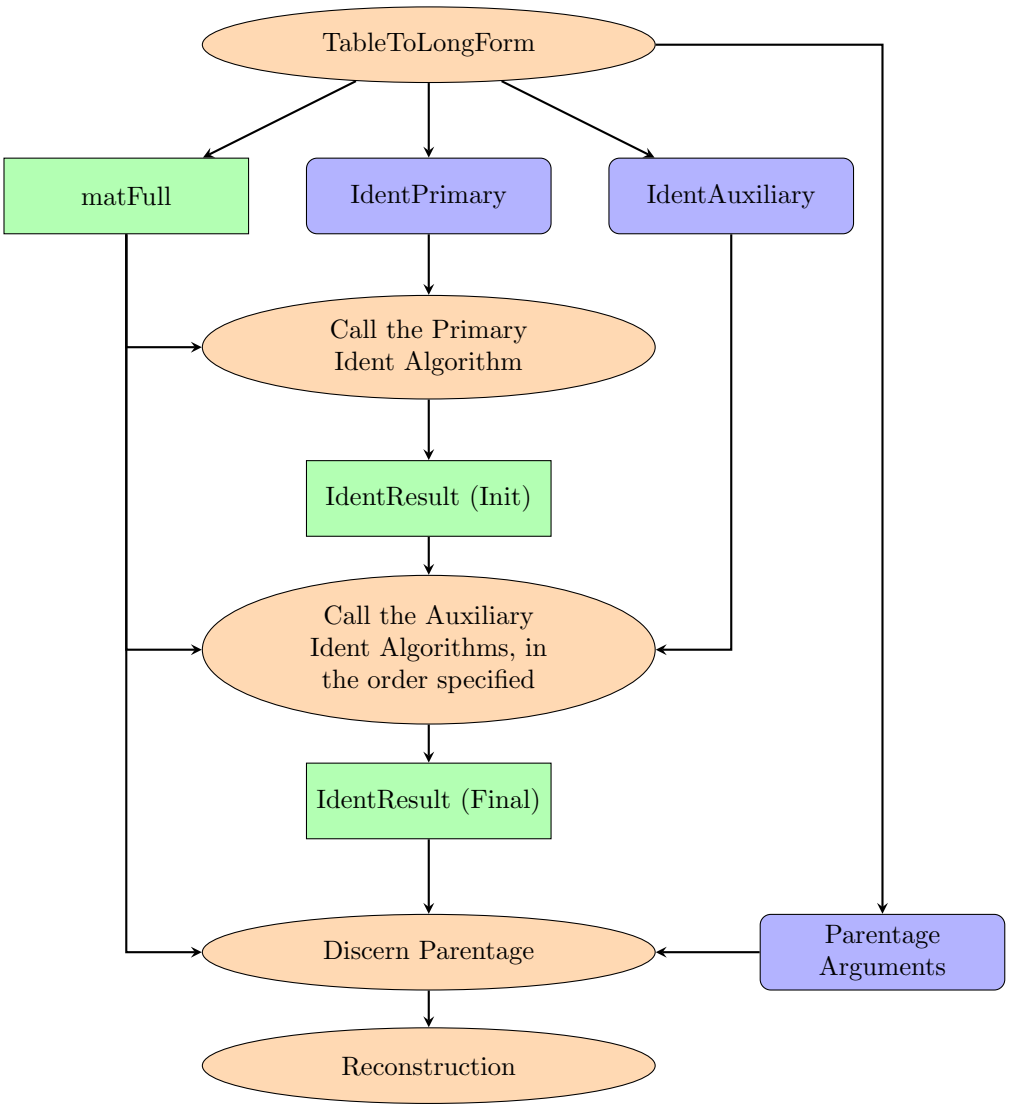


Figure 2: We break down the Identification stage in more detail. The Table argument is referred to as matFull internally. The IdentPrimary and IdentAuxiliary arguments specify what algorithms will be called at the respective steps of the workflow.

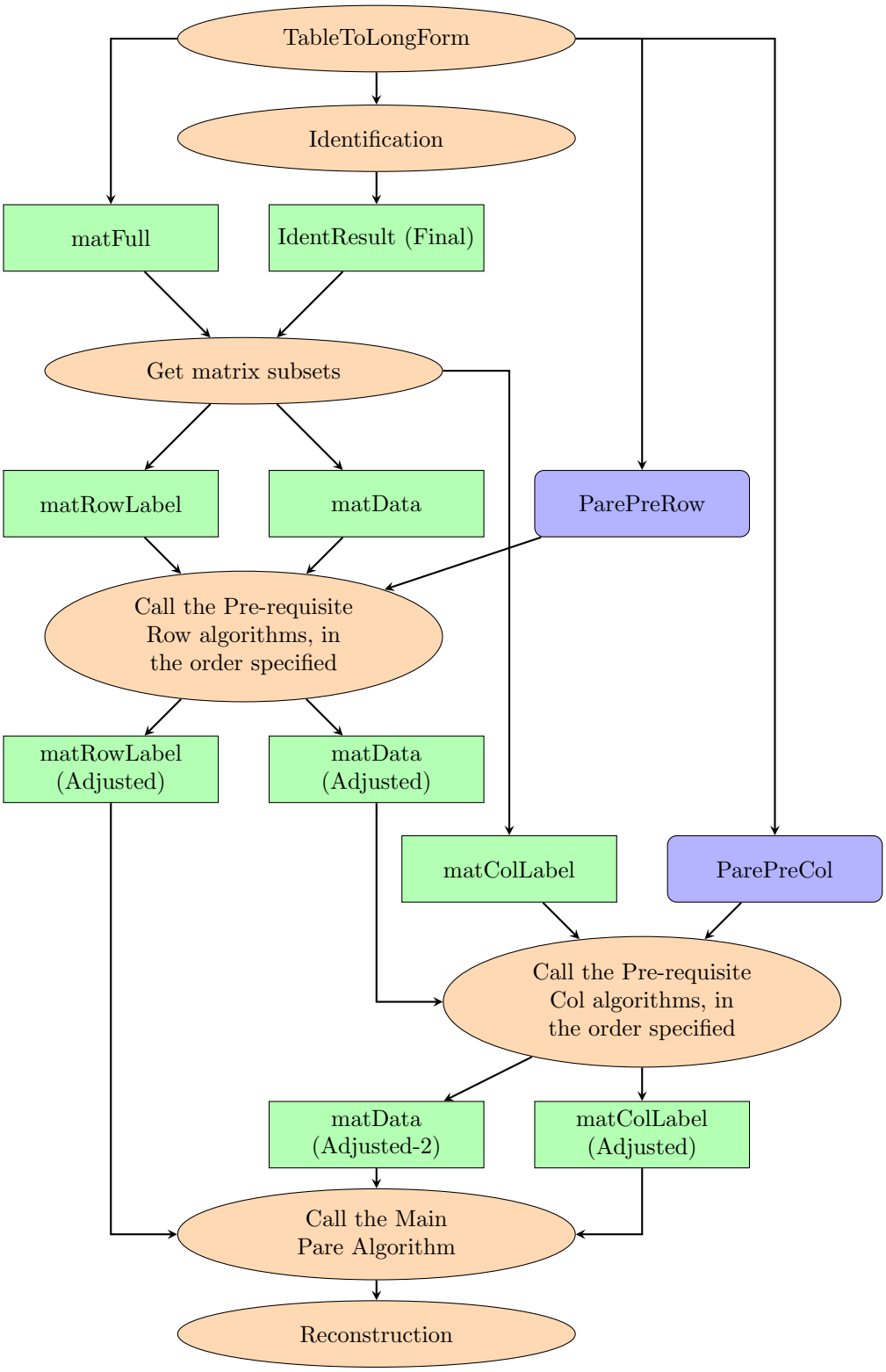


Figure 3: We break down the Discern Parentage stage in more detail. We use the IdentResult from the Identification stage to obtain subsets of matFull that correspond to just the labels or the data. We can then use these subsets to discern the parentage of the labels. The ParePreRow and ParePreCol arguments specify what algorithms will be called at the respective steps of the workflow, and they make adjustments to the matrix subsets so that the Main Parentage algorithm can function correctly.

# 4 Registering a New Module

This is done with a call to `TTLFaliasAdd`, which has the following arguments:

**Type** e.g. IdentPrimary

**Fname** the name of the Function/Algorithm

**Falias** the alias for the Function/Algorithm, which is used for the call to `TableToLongForm`

**Author** (optional) name of the author of the algorithm

**Description** (optional) a short description of the purpose of the algorithm

For example, the following is used to register the default Ident Primary algorithm.

```
TTLFaliasAdd("IdentPrimary", "IdentbyMostCommonBoundary", "combound",
             "Base Algorithm", "Default IdentPrimary algorithm")
```

The `.R` file that contains the function(s) should also contain this registration line. Then during an R session, one can load the TableToLongForm package, then source the module's `.R` file to register the module's algorithm(s).

One can check if this is successful by then calling `TTLFaliasList`. The output with no additional modules is as follows (line-breaks added):

```
> TTLFaliasList()
==Type: IdentPrimary==
Name: IdentbyMostCommonBoundary
Alias: combound
Author: Base Algorithm
Description: Default IdentPrimary algorithm

==Type: IdentAuxiliary==
Name: IdentbySequence
Alias: sequence
Author: Base Algorithm
Description: Search for fully numeric row labels (e.g. Years)
  that were misidentified as data

==Type: ParePreCol==
Name: ParePreColMismatch
Alias: mismatch
Author: Base Algorithm
Description: Correct for column labels not matched correctly
  over data (label in a different column to data)

Name: ParePreColMisaligned
Alias: misalign
Author: Base Algorithm
Description: Correct for column labels not aligned correctly over
  data (parents not positioned on the far-left, relative to their
  children in the row below)

Name: ParePreColMultirow
Alias: multirow
Author: Base Algorithm
Description: Merge long column labels that were physically split
  over multiple rows back into a single label
```

If your new algorithms were successfully registered, they should appear on this list, and the aliases for the new algorithms can be used during a call to `TableToLongForm`.