
Load Pull Data Analysis

Release 1.0

Rutvij S, Adhiraj S, Ashby J, Hanna K, Subhash RB

May 02, 2021

CONTENTS:

1	loadPullDataAnalysis.dataXformation module	1
2	loadPullDataAnalysis.mdfParser module	4
	Index	5

LOADPULLDATAANALYSIS.DATAXFORMATION MODULE

`loadPullDataAnalysis.dataXformation.calcGComp(df: pandas.core.frame.DataFrame) → pandas.core.frame.DataFrame`

Finds the maximum value for Gain, then use it to calculate gComp.

Parameters `df` (*pandas.DataFrame*) – DataFrame with the Gain column

Returns The updated dataframe with gComp added as a column

Return type *pandas.DataFrame*

`loadPullDataAnalysis.dataXformation.dfFromPkl(filename: str) → pandas.core.frame.DataFrame`

Reading the pickle file and initialiazing it as a dataframe (df).

Parameters `filename` (*str*) – The pickle file generated from the parser

Returns The dataframe that is unpickled from the pickled file (generated by the parser)

Return type *pandas.DataFrame*

`loadPullDataAnalysis.dataXformation.dfWithCols(df: pandas.core.frame.DataFrame, ls: list) → pandas.core.frame.DataFrame`

Return the dataframe retaining only certain columns.

Parameters

- `df` (*pandas.DataFrame*) – DataFrame whose columns need to be subset.
- `ls` (*list[str]*) – List of columns to retain.

Returns Copy of the updated df with columns

Return type *pandas.DataFrame*

`loadPullDataAnalysis.dataXformation.filterColVal(df: pandas.core.frame.DataFrame, colName: str, value: float, filType: Optional[str] = None) → pandas.core.frame.DataFrame`

Filter a column of the DF using the value and operation provided.

Parameters

- `df` (*pandas.DataFrame*) – The original df
- `colName` (*str*) – Name of the column to be filtered
- `value` (*str*) – Value to compare against
- `fillType` (*pandas.DataFrame*) – The string value indicating what operation should be used to filter the column values (`>=` is geq, `>` gt, `<` lt, `<=` leq, `==` eq)

Returns Filtered DF.

Return type pandas.DataFrame

```
loadPullDataAnalysis.dataXformation.filterOnCompressionThreshold(df: pandas.core.frame.DataFrame,
                                                                    compVal: float)
                                                                    → pandas.core.frame.DataFrame
```

Filters a dataframe based on a given compression value.

Parameters **df** (pandas.DataFrame) – DataFrame with the gComp column

Returns The updated dataframe with points only beyond the compression value.

Return type pandas.DataFrame

```
loadPullDataAnalysis.dataXformation.interpolatedSlice(dfList: list, sliceVar: str,
                                                         sliceVal: float) → tuple
```

Slices a column of the DFs at a particular value and returns a DF with values for other columns interpolated at that value.

Parameters

- **dfList** (list [pd.DataFrame]) – A list of DataFrames with unique gamma values.
- **sliceVar** (str) – Column of the df that will be sliced
- **sliceVal** (float) – The numeric value to slice at

Returns selList, a list of potential plotting variables

Return type list[str]

Returns dfOfLoadsAtVarX the interpolated DF.

Return type pandas.DataFrame

```
loadPullDataAnalysis.dataXformation.pickVariable(sliceVar: str, df: pandas.core.frame.DataFrame)
                                                                    → dict
```

Returns the information for a variable in the DF. This is used to information potential valid slicing values.

Parameters

- **selVar** – The name of the column to be sliced.
- **df** (pandas.DataFrame) – DataFrame containing that column.

Returns A dictionary that contains maxVal, minVal, stepSize, defaultVal

Return type dict[str, float]

```
loadPullDataAnalysis.dataXformation.splitGammaTuple(df: pandas.core.frame.DataFrame)
                                                                    → pandas.core.frame.DataFrame
```

Split the Gamma Tuple into its component vals, one column for each member.

:param df:Dataframe with the gammaTuple as one of its cols. :type df: pandas.DataFrame :returns: The dataframe with split gamma values. :rtype: pandas.DataFrame

`loadPullDataAnalysis.dataXformation.splitOnUniqueGammaTuples` (*df*: *pandas.core.frame.DataFrame*)
→ list

Creates a list of DFs each one having a unique GammaTuple value and all the power indices for each.

Parameters *df* (*pandas.DataFrame*) – DataFrame with gammaTuple as one of its cols.

Returns A list of DataFrames with unique Gamma vals separated.

Return type list[pandas.DataFrame]

LOADPULLDATAANALYSIS.MDFPARSER MODULE

`loadPullDataAnalysis.mdfParser.calculateMetrics` (*df*: *pandas.core.frame.DataFrame*) → *pandas.core.frame.DataFrame*

Calculates load pull data from observed data. Adds columns for Pin, Pout, Gain, PAE, drain efficiency, and Load Gamma (r: real, jx: imaginary).

Parameters *df* (*pandas.DataFrame*) – The pandas data frame generated by the `parseMdf` function.

Returns A DataFrame with the columns of calculated metrics added to the DF.

Return type *pandas.DataFrame*

`loadPullDataAnalysis.mdfParser.exportFiles` (*df*: *pandas.core.frame.DataFrame*, *filepath*: *str*) → None

Exports the DataFrame to .csv and .pkl format for later use.

Parameters

- *df* (*pandas.DataFrame*) – The pandas data frame generated by the `calculateMetrics` or `unitConversions` function.
- *filepath* (*str*) – The name of the file to be saved + the location. path/filename, if the file is to be saved in the same directory, just provide filename.

Returns Nothing.

Return type None

`loadPullDataAnalysis.mdfParser.parseMdf` (*fileLoc*: *str*) → *pandas.core.frame.DataFrame*
Parses an MDF file generated by AWR Cadence (Microwave Office)

Parameters *fileLoc* (*str*) – The file location of the MDF file, if the file is in the same directory, then only the filename is required.

Returns The MDF file converted into a pandas DataFrame

Return type *pandas.DataFrame*

`loadPullDataAnalysis.mdfParser.unitConversions` (*df*: *pandas.core.frame.DataFrame*) → *pandas.core.frame.DataFrame*

Converts Gain to dB, and Pout/Pin to dBm

Parameters *df* (*pandas.DataFrame*) – The pandas data frame generated by the `calculateMetrics` function.

Returns A DataFrame with the units of Gain, Pin and Pout converted in place.

Return type *pandas.DataFrame*

INDEX

C

`calcGComp()` (in module `loadPullDataAnalysis.dataXformation`), 1
`calculateMetrics()` (in module `loadPullDataAnalysis.mdfParser`), 4

D

`dfFromPkl()` (in module `loadPullDataAnalysis.dataXformation`), 1
`dfWithCols()` (in module `loadPullDataAnalysis.dataXformation`), 1

E

`exportFiles()` (in module `loadPullDataAnalysis.mdfParser`), 4

F

`filterColVal()` (in module `loadPullDataAnalysis.dataXformation`), 1
`filterOnCompressionThreshold()` (in module `loadPullDataAnalysis.dataXformation`), 2

I

`interpolatedSlice()` (in module `loadPullDataAnalysis.dataXformation`), 2

L

`loadPullDataAnalysis.dataXformation`
module, 1
`loadPullDataAnalysis.mdfParser`
module, 4

M

module
 `loadPullDataAnalysis.dataXformation`,
 1
 `loadPullDataAnalysis.mdfParser`, 4

P

`parseMdf()` (in module `loadPullDataAnalysis.mdfParser`), 4

`pickVariable()` (in module `loadPullDataAnalysis.dataXformation`), 2

S

`splitGammaTuple()` (in module `loadPullDataAnalysis.dataXformation`), 2
`splitOnUniqueGammaTuples()` (in module `loadPullDataAnalysis.dataXformation`), 2

U

`unitConversions()` (in module `loadPullDataAnalysis.mdfParser`), 4