

# Introduction to the *TPP* package for analyzing Thermal Proteome Profiling data: 2D-TPP experiments

***Dorothee Childs, Nils Kurzawa***

European Molecular Biology Laboratory (EMBL),  
Heidelberg, Germany  
`dorothee.childs@embl.de`

***TPP* version 3.6.0 (Last revision 2016-10-24)**

## Abstract

Thermal Proteome Profiling (TPP) combines the cellular thermal shift assay concept [1] with mass spectrometry based proteome-wide protein quantitation [2]. Thereby, drug-target interactions can be inferred from changes in the thermal stability of a protein upon drug binding, or upon downstream cellular regulatory events, in an unbiased manner.

The package *TPP* facilitates this process by providing executable workflows that conduct all necessary data analysis steps. Recent advances in the field have lead to the development of so called 2D Thermal Proteome Profiling (2D-TPP) experiments [3]. Recent advances in the field have lead to the development of so called 2D Thermal Proteome Profiling (2D-TPP) experiments [3]. Similar as for the TPP-TR and the TPP-CCR analysis, the function `analyze2DTPP` executes the whole workflow from data import through normalization and curve fitting to statistical analysis. Nevertheless, all of these steps can also be invoked separately by the user. The corresponding functions can be recognized by their suffix `tpp2d`.

Here, we first show how to start the whole analysis using `analyze2DTPP`. Afterwards, we demonstrate how to carry out single steps individually.

For details about the analysis of 1D TR- or CCR experiments [2, 4], please refer to the vignette `TPP_introduction_1D`.

## Contents

1	Installation . . . . .	2
1.1	Special note for Windows users . . . . .	2
2	Analyzing 2D-TPP experiments. . . . .	3
2.1	Overview . . . . .	3
2.2	Performing the analysis . . . . .	3
2.3	Quality control analyses . . . . .	15
2.4	Spline fits of treatment effects over temperature . . . . .	16

# 1 Installation

---

To install the package, type the following commands into the *R* console

```
source("http://bioconductor.org/biocLite.R")
biocLite("TPP")
```

The installed package can be loaded by

```
library("TPP")
```

## 1.1 Special note for Windows users

The *TPP* package uses the *openxlsx* package to produce Excel output [5]. *openxlsx* requires a zip application to be installed on your system and to be included in the path. On Windows, such a zip application is not installed by default, but is available, for example, via [Rtools](#). Without the zip application, you can still use the 'TPP' package and access its results via the dataframes produced by the main functions.

## 2 Analyzing 2D-TPP experiments

### 2.1 Overview

Before you can start your analysis, you need to specify information about your experiments:

The mandatory information comprises a unique experiment name, as well as the isobaric labels and corresponding temperature values for each experiment. The package retrieves this information from a configuration table that you need to specify before starting the analysis. This table can either be a data frame that you define in your R session, or a spreadsheet in .xlsx or .csv format. In a similar manner, the measurements themselves can either be provided as a list of data frames, or imported directly from files during runtime.

We demonstrate the functionality of the package using the dataset `Panobinostat_2DTPP_smallExampleData`. It contains an illustrative subset of a larger dataset which was obtained by 2D-TPP experiments on HepG2 cells treated with the histone deacetylase (HDAC) inhibitor panobinostat in the treatment groups and with vehicle in the control groups. The experiments were performed for different temperatures. The raw MS data were processed with the Python package `isobarQuant`, which provides protein fold changes relative to the protein abundance at the lowest temperature as input for the TPP package [3].

### 2.2 Performing the analysis

First of all, we load an example data set:

```
data(panobinostat_2DTPP_smallExample, package = "TPP")
```

Using this command we load two objects:

1. `Panobinostat_2DTPP_smallExampleData`: a list of data frames that contain the measurements to be analyzed,
2. `hdac2D_config`: a configuration table with details about each experiment.

```
config_tpp2d <- panobinostat_2DTPP_config
data_tpp2d <- panobinostat_2DTPP_data
```

```
config_tpp2d
```

##	Compound	Experiment	Temperature	126	127L	127H	128L	128H	129L	129H
## 1	Panobinostat	X020466	42.0	5	1	0.143	0.02	0	-	-
## 2	Panobinostat	X020466	44.1	-	-	-	-	-	5	1
## 3	Panobinostat	X020467	46.2	5	1	0.143	0.02	0	-	-
## 4	Panobinostat	X020467	48.1	-	-	-	-	-	5	1
## 5	Panobinostat	X020468	50.4	5	1	0.143	0.02	0	-	-
## 6	Panobinostat	X020468	51.9	-	-	-	-	-	5	1
## 7	Panobinostat	X020469	54.0	5	1	0.143	0.02	0	-	-
## 8	Panobinostat	X020469	56.1	-	-	-	-	-	5	1
## 9	Panobinostat	X020470	58.2	5	1	0.143	0.02	0	-	-
## 10	Panobinostat	X020470	60.1	-	-	-	-	-	5	1
## 11	Panobinostat	X020471	62.4	5	1	0.143	0.02	0	-	-
## 12	Panobinostat	X020471	63.9	-	-	-	-	-	5	1

## TPP

```
##      130L 130H 131L RefCol Path
## 1      -    -    -    128H
## 2 0.143 0.02    0    131L
## 3      -    -    -    128H
## 4 0.143 0.02    0    131L
## 5      -    -    -    128H
## 6 0.143 0.02    0    131L
## 7      -    -    -    128H
## 8 0.143 0.02    0    131L
## 9      -    -    -    128H
## 10 0.143 0.02    0    131L
## 11     -    -    -    128H
## 12 0.143 0.02    0    131L

data_tpp2d %>% str(1)

## List of 6
## $ X020466:'data.frame': 484 obs. of  15 variables:
## $ X020467:'data.frame': 478 obs. of  15 variables:
## $ X020468:'data.frame': 448 obs. of  15 variables:
## $ X020469:'data.frame': 372 obs. of  15 variables:
## $ X020470:'data.frame': 306 obs. of  15 variables:
## $ X020471:'data.frame': 261 obs. of  15 variables:
```

The data object `Panobinostat_2DTPP_smallExampleData` is organized as a list of data frames which contain the experimental raw data of an 2D-TPP experiment. The names of the list elements correspond to the different multiplexed experiments. Each experimental dataset contains the following columns:

```
data_tpp2d$X020466 %>% colnames

## [1] "clustername"          "representative"
## [3] "msexperiment_id"      "qupm"
## [5] "qusm"                 "sumionarea_protein_126"
## [7] "sumionarea_protein_127L" "sumionarea_protein_127H"
## [9] "sumionarea_protein_128L" "sumionarea_protein_128H"
## [11] "sumionarea_protein_129L" "sumionarea_protein_129H"
## [13] "sumionarea_protein_130L" "sumionarea_protein_130H"
## [15] "sumionarea_protein_131L"
```

In order to perform the complete workflow we can now simply use:

```
tpp2dResults <- analyze2DTPP(configTable = config_tpp2d,
                             data = data_tpp2d,
                             compFc = TRUE,
                             idVar = "representative",
                             intensityStr = "sumionarea_protein_",
                             nonZeroCols = "qusm",
                             addCol = "clustername",
                             methods = "doseResponse",
                             createReport = "none")

## Warning in function_list[[k]](value): NAs introduced by coercion
```



## TPP

[illegible]

## TPP

```
## Mean :1.0035          Mean :1.0105
## 3rd Qu.:1.0727        3rd Qu.:1.0632
## Max. :4.6565          Max. :5.8855
##
## norm_rel_fc_protein_1_unmodified norm_rel_fc_protein_5_unmodified
## Min. : 0.2422          Min. : 0.2512
## 1st Qu.: 0.9344        1st Qu.: 0.9337
## Median : 1.0000        Median : 1.0000
## Mean : 1.0163          Mean : 1.0259
## 3rd Qu.: 1.0654        3rd Qu.: 1.0589
## Max. :10.0240          Max. :17.0405
##
## norm_rel_fc_protein_0_normalized_to_lowest_conc
## Min. :1
## 1st Qu.:1
## Median :1
## Mean :1
## 3rd Qu.:1
## Max. :1
##
## norm_rel_fc_protein_0.02_normalized_to_lowest_conc
## Min. :0.1767
## 1st Qu.:0.9192
## Median :1.0000
## Mean :1.0035
## 3rd Qu.:1.0727
## Max. :4.6565
##
## norm_rel_fc_protein_0.143_normalized_to_lowest_conc
## Min. :0.2612
## 1st Qu.:0.9364
## Median :1.0000
## Mean :1.0105
## 3rd Qu.:1.0632
## Max. :5.8855
##
## norm_rel_fc_protein_1_normalized_to_lowest_conc
## Min. : 0.2422
## 1st Qu.: 0.9344
## Median : 1.0000
## Mean : 1.0163
## 3rd Qu.: 1.0654
## Max. :10.0240
##
## norm_rel_fc_protein_5_normalized_to_lowest_conc
## Min. : 0.2512
## 1st Qu.: 0.9337
## Median : 1.0000
## Mean : 1.0259
## 3rd Qu.: 1.0589
## Max. :17.0405
```

## TPP

```
##
## norm_rel_fc_protein_0_transformed norm_rel_fc_protein_0.02_transformed
## Min. :0.000 Min. : -0.884
## 1st Qu.:0.000 1st Qu.: -0.154
## Median :1.000 Median : 0.297
## Mean :0.621 Mean : 0.302
## 3rd Qu.:1.000 3rd Qu.: 0.614
## Max. :1.000 Max. : 2.542
## NA's :4421 NA's :4421
## norm_rel_fc_protein_0.143_transformed norm_rel_fc_protein_1_transformed
## Min. : -1.201 Min. : -0.961
## 1st Qu.: 0.086 1st Qu.: 0.095
## Median : 0.376 Median : 0.313
## Mean : 0.400 Mean : 0.400
## 3rd Qu.: 0.662 3rd Qu.: 0.652
## Max. : 3.294 Max. : 2.925
## NA's :4421 NA's :4421
## norm_rel_fc_protein_5_transformed pEC50 slope
## Min. :0.000 Min. :5.728 Min. : -50.000
## 1st Qu.:0.000 1st Qu.:6.696 1st Qu.: -10.804
## Median :0.000 Median :7.778 Median : -1.000
## Mean :0.379 Mean :7.346 Mean : -8.302
## 3rd Qu.:1.000 3rd Qu.:8.126 3rd Qu.: 1.159
## Max. :1.000 Max. :8.126 Max. : 50.000
## NA's :4421 NA's :4421 NA's :4421
## R_sq plot compound_effect meets_FC_requirement
## Min. : -0.068 NA's:4656 destabilized: 146 Mode:logical
## 1st Qu.: 0.545 stabilized : 89 FALSE:4537
## Median : 0.723 NA's :4421 TRUE :119
## Mean : 0.675
## 3rd Qu.: 0.881
## Max. : 1.000
## NA's :4421
## passed_filter pEC50_outside_conc_range model_converged
## Mode:logical Mode:logical Mode:logical
## FALSE:4601 FALSE:111 TRUE:235
## TRUE :55 TRUE :124 NA's:4421
## NA's :4421
##
##
##
## pEC50_quality_check sufficient_data_for_fit protein_identified_in
## 5.72818301656452: 12 Mode:logical Mode:logical
## 6.07074587494624: 6 TRUE:235 TRUE:4656
## 7.44099730847312: 6 NA's:4421
## 6.75587159170968: 2
## 5.83469502048232: 1
## (Other) : 84
## NA's :4545
## representative qupm qusm clustername
## IPI00000001.2: 12 Min. : 1.000 Min. : 1.00 A2M : 12
```



```
## IPI00000005.1: 12 1st Qu.: 3.000 1st Qu.: 5.00 ABHD10 : 12
## IPI00000690.1: 12 Median : 7.000 Median : 11.00 ACAA1 : 12
## IPI00000811.2: 12 Mean : 9.149 Mean : 19.57 AC01 : 12
## IPI00000875.7: 12 3rd Qu.:12.000 3rd Qu.: 23.00 AC02 : 12
## IPI00001914.1: 12 Max. :87.000 Max. :263.00 ACTC1 : 12
## (Other) :4584 (Other):4584
## sumionarea_protein_5 sumionarea_protein_1 sumionarea_protein_0.143
## Min. :2.063e+05 Min. :3.819e+05 Min. :3.579e+05
## 1st Qu.:7.696e+07 1st Qu.:7.604e+07 1st Qu.:8.079e+07
## Median :2.511e+08 Median :2.512e+08 Median :2.591e+08
## Mean :7.182e+08 Mean :7.542e+08 Mean :7.554e+08
## 3rd Qu.:7.382e+08 3rd Qu.:7.682e+08 3rd Qu.:7.857e+08
## Max. :2.125e+10 Max. :2.138e+10 Max. :1.924e+10
##
## sumionarea_protein_0.02 sumionarea_protein_0 temperature experiment
## Min. :4.335e+05 Min. :2.925e+05 Min. :42.0 X020466:968
## 1st Qu.:8.401e+07 1st Qu.:7.345e+07 1st Qu.:46.2 X020467:950
## Median :2.739e+08 Median :2.574e+08 Median :50.4 X020468:894
## Mean :8.100e+08 Mean :8.599e+08 Mean :51.6 X020469:738
## 3rd Qu.:8.331e+08 3rd Qu.:8.554e+08 3rd Qu.:56.1 X020470:600
## Max. :2.249e+10 Max. :2.644e+10 Max. :63.9 X020471:506
##
## rel_fc_protein_5 rel_fc_protein_1 rel_fc_protein_0.143 rel_fc_protein_0.02
## Min. : 0.3487 Min. :0.2985 Min. :0.3887 Min. : 0.1882
## 1st Qu.: 0.7894 1st Qu.:0.8231 1st Qu.:0.8156 1st Qu.: 0.8413
## Median : 0.8964 Median :0.9197 Median :0.9415 Median : 0.9601
## Mean : 0.9935 Mean :0.9753 Mean :1.0187 Mean : 1.0974
## 3rd Qu.: 1.0878 3rd Qu.:1.0588 3rd Qu.:1.1447 3rd Qu.: 1.2027
## Max. :17.1835 Max. :8.6463 Max. :6.2354 Max. :10.0917
##
## rel_fc_protein_0
## Min. :1
## 1st Qu.:1
## Median :1
## Mean :1
## 3rd Qu.:1
## Max. :1
##
```

Moreover, we can also invoke the single functions of the workflow manually. Therefore, we start with importing the data. Using the import function the data is subsequently imported and stored in a single dataframe containing all the required data columns and those that the user likes to take along through the analysis to be displayed together with the results of this workflow.

## TPP

[illegible]

[illegible]

```
head(data2d)
```

##	representative	qupm	qusm	clustername	sumionarea_protein_5
## 1	IPI00000001.2	15	25	STAU1	1193994914
## 2	IPI00000001.2	15	25	STAU1	1272771185
## 3	IPI00000001.2	13	22	STAU1	1482437522

## TPP

```
## 4 IPI00000001.2 13 22 STAU1 1157290962
## 5 IPI00000001.2 15 24 STAU1 396823892
## 6 IPI00000001.2 15 24 STAU1 345169960
## sumionarea_protein_1 sumionarea_protein_0.143 sumionarea_protein_0.02
## 1 1337957734 1375948494 1956350223
## 2 1473572092 1273285951 1669312103
## 3 1513181000 1284434575 1487032006
## 4 1050288621 1110810226 1128507681
## 5 458022616 453860821 412257039
## 6 350182409 352193788 344410388
## sumionarea_protein_0 temperature experiment unique_ID
## 1 1801848318 42.0 X020466 X020466_42_IPI00000001.2
## 2 1404292404 44.1 X020466 X020466_44.1_IPI00000001.2
## 3 1422365645 46.2 X020467 X020467_46.2_IPI00000001.2
## 4 999666282 48.1 X020467 X020467_48.1_IPI00000001.2
## 5 439399665 50.4 X020468 X020468_50.4_IPI00000001.2
## 6 309019704 51.9 X020468 X020468_51.9_IPI00000001.2

attr(,"data2d","importSettings")

## $proteinIdCol
## [1] "representative"
##
## $uniqueIdCol
## [1] "unique_ID"
##
## $addCol
## [1] "clustername"
##
## $intensityStr
## [1] "sumionarea_protein_"
##
## $qualColName
## [1] "qupm"
##
## $nonZeroCols
## [1] "qusm"
##
## $fcStr
## NULL
```

If we haven't computed fold changes from the raw "sumionarea" data, as it is the case in this example, we can invoke the function `tpp2dComputeFoldChanges` in order to do so:

```
fcData2d <- tpp2dComputeFoldChanges(data = data2d)
```

Thereon the function adds additional columns to our dataframe containing corresponding fold changes:

```
head(fcData2d)

## representative qupm qusm clustername sumionarea_protein_5
## 1 IPI00000001.2 15 25 STAU1 1193994914
```

## TPP

```
## 2 IPI00000001.2 15 25 STAU1 1272771185
## 3 IPI00000001.2 13 22 STAU1 1482437522
## 4 IPI00000001.2 13 22 STAU1 1157290962
## 5 IPI00000001.2 15 24 STAU1 396823892
## 6 IPI00000001.2 15 24 STAU1 345169960
## sumionarea_protein_1 sumionarea_protein_0.143 sumionarea_protein_0.02
## 1 1337957734 1375948494 1956350223
## 2 1473572092 1273285951 1669312103
## 3 1513181000 1284434575 1487032006
## 4 1050288621 1110810226 1128507681
## 5 458022616 453860821 412257039
## 6 350182409 352193788 344410388
## sumionarea_protein_0 temperature experiment unique_ID
## 1 1801848318 42.0 X020466 X020466_42_IPI00000001.2
## 2 1404292404 44.1 X020466 X020466_44.1_IPI00000001.2
## 3 1422365645 46.2 X020467 X020467_46.2_IPI00000001.2
## 4 999666282 48.1 X020467 X020467_48.1_IPI00000001.2
## 5 439399665 50.4 X020468 X020468_50.4_IPI00000001.2
## 6 309019704 51.9 X020468 X020468_51.9_IPI00000001.2
## rel_fc_5 rel_fc_1 rel_fc_0.143 rel_fc_0.02 rel_fc_0
## 1 0.6626501 0.7425474 0.7636317 1.0857463 1
## 2 0.9063434 1.0493342 0.9067100 1.1887212 1
## 3 1.0422338 1.0638481 0.9030270 1.0454640 1
## 4 1.1576773 1.0506392 1.1111810 1.1288844 1
## 5 0.9031047 1.0423827 1.0329112 0.9382279 1
## 6 1.1169837 1.1332041 1.1397130 1.1145257 1
```

We can then normalize the data by performing a median normalization on the fold changes, in order to account for experiment specific noise.

```
normData2d <- tpp2dNormalize(data = fcData2d)
head(normData2d)

## representative qupm qusm clustername sumionarea_protein_5
## 1 IPI00000001.2 15 25 STAU1 1193994914
## 2 IPI00000001.2 15 25 STAU1 1272771185
## 3 IPI00000001.2 13 22 STAU1 1482437522
## 4 IPI00000001.2 13 22 STAU1 1157290962
## 5 IPI00000001.2 15 24 STAU1 396823892
## 6 IPI00000001.2 15 24 STAU1 345169960
## sumionarea_protein_1 sumionarea_protein_0.143 sumionarea_protein_0.02
## 1 1337957734 1375948494 1956350223
## 2 1473572092 1273285951 1669312103
## 3 1513181000 1284434575 1487032006
## 4 1050288621 1110810226 1128507681
## 5 458022616 453860821 412257039
## 6 350182409 352193788 344410388
## sumionarea_protein_0 temperature experiment unique_ID
## 1 1801848318 42.0 X020466 X020466_42_IPI00000001.2
## 2 1404292404 44.1 X020466 X020466_44.1_IPI00000001.2
## 3 1422365645 46.2 X020467 X020467_46.2_IPI00000001.2
## 4 999666282 48.1 X020467 X020467_48.1_IPI00000001.2
```

## TPP

```
## 5      439399665      50.4      X020468 X020468_50.4_IPI00000001.2
## 6      309019704      51.9      X020468 X020468_51.9_IPI00000001.2
##      rel_fc_5 rel_fc_1 rel_fc_0.143 rel_fc_0.02 rel_fc_0 norm_rel_fc_5
## 1 0.6626501 0.7425474 0.7636317 1.0857463      1      1.107187
## 2 0.9063434 1.0493342 0.9067100 1.1887212      1      1.114453
## 3 1.0422338 1.0638481 0.9030270 1.0454640      1      1.187727
## 4 1.1576773 1.0506392 1.1111810 1.1288844      1      1.249516
## 5 0.9031047 1.0423827 1.0329112 0.9382279      1      1.123552
## 6 1.1169837 1.1332041 1.1397130 1.1145257      1      1.171933
##      norm_rel_fc_1 norm_rel_fc_0.143 norm_rel_fc_0.02 norm_rel_fc_0
## 1      1.059331      1.105019      1.244416      1
## 2      1.164559      1.022695      1.195813      1
## 3      1.229422      1.078735      1.211522      1
## 4      1.147406      1.108487      1.329434      1
## 5      1.268366      1.267164      1.256324      1
## 6      1.176446      1.158041      1.163566      1
```

To run the TPP-CCR main function on our 2D-TPP data we now invoke:

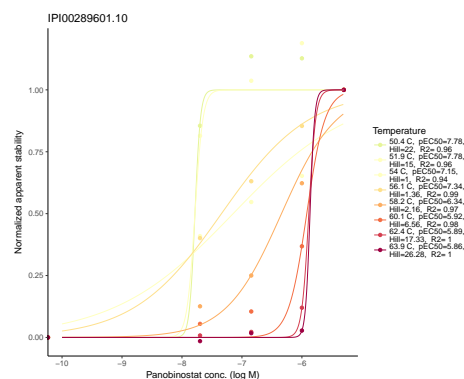
```
ccr2dResults <- tpp2dCurveFit(data = normData2d)
```

Now we can plot the curves for any of the proteins for which at least one CCR curve could be fitted. In this case we choose HDAC2:

```
drPlots <- tpp2dCreateDRplots(data = ccr2dResults, type = "good")
```

```
# Find IPI id for HDAC2 (in column representative):
IPI_id_HDAC2 <- unique(filter(ccr2dResults, clustername == "HDAC2")$representative)

# Show corresponding plot:
drPlots[[IPI_id_HDAC2]]
```



And we can also plot the single curves for each of the proteins with:

```
drPlotsByTemperature <- tpp2dCreateDRplots(data = ccr2dResults, type = "single")
drPlotsByTemperature[[IPI_id_HDAC2]][["54"]]
```

## 2.3 Quality control analyses

In order to access the quality of the experimental 2D-TPP data set acquired in a specific cell line, we recommend to compare the data with vehicle TR experiments (at least two replicates) of the same cell line. For the analysis of this data we supply a QC-workflow that enables comparison of treatment and non-treatment samples with reference data.

In order to start this workflow the first thing we need to do, is to generate a cell line specific TR reference object. We also need to specify the result path where this object should be stored:

```
resultPath = file.path(getwd(), 'Panobinostat_Vignette_Example_2D')
if (!file.exists(resultPath)) dir.create(resultPath, recursive = TRUE)

trConfig <- file.path(system.file("example_data", package="TPP"),
                      "2D_example_data/panobinostat_ex_config.csv")

tpp2dCreateTPPTRreference(trConfigTable = trConfig,
                        resultPath = resultPath,
                        outputName = "desired_file_name",
                        createFCboxplots = FALSE)
```

For the purpose of explaining this workflow, we will use a reference data set of a HepG2 cell line supplied with this package. Originating from this object we can now perform various quality control steps. First of all by setting the *createFCboxplots* flag to true, we can generate box plot melting curves of the reference data which are first of all informative of the quality of the reference data and illustrate melting behavior of all proteins without any treatment.

Calling the function will generate a couple of output files in the indicated output directory.

- The `tppRefData.RData` file is the most important one. This is the file that has to be referenced by indication of a system path to this file when calling functions to generate the 2D-TPP spline plots and perform an F test. When loaded in *R* the object `tppRefData` represents a list with the following elements:
  - `tppCfgTable`: the TPP-TR configurable which was used for generating this object
  - `sumResTable` a list of two elements:
    - `detail`: the exact result data from the TR analysis and
    - `summary`: a summary of the analyzed TR data comprising the median and standard deviation values of the measurements at the different temperatures (encoded by the isobaric labels)
      - `temperatures`: a table listing the temperatures which were used in the TR experiment in the different replicates
      - `lbsByTemp`: a table matching each temperature to an isobaric label
- An excel file which summarizes the data present in `tppRefData` on different sheets
- Textfiles representing the sheets of the excel file as plain text
- `normalizedData.RData` containing the TPP-TR data after normalization
- `resultTable.RData` containing the TPP-TR analysis result table

## TPP

Secondly, we can generate plots which visualize the melting point temperatures of the 2D-TPP data in comparison to the TR reference data. Here we demonstrate this function on a subset of the proteins:

```
# set the system path for the HepG2 TR reference data set:
trRef <- file.path(system.file("data", package="TPP"), "TPPTR_reference_results_HepG2.RData")

plotData <- ccr2dResults %>% filter(clustername %in% IPI_id_HDAC2)

pEC50QC_HDAC1 <- tpp2dPlotQCpEC50(resultTable = plotData,
                                   resultPath = resultPath,
                                   trRef = trRef,
                                   idVar = "representative")

print(pEC50QC_HDAC1)

## named list()
```

We have therefore used the *ccr2dResults* data frame which we previously generated by invoking the TPP-CCR routine and the the respective configTable.

Moreover, we can generate plots that visualize the distributions of fold changes over the different treatment concentrations and temperatures and how the normalization affected them (of course only if we previously performed a normalization). The function automatically also visualizes various other characteristics of the data, such as how proteins behave in neighboring temperatures which are multiplexed. It can be invoked as follows:

```
tpp2dPlotQChist(configFile = config_tpp2d,
                 resultTable = ccr2dResults,
                 resultPath = resultPath,
                 trRef = trRef,
                 idVar = "representative")

dir(resultPath)

## [1] "qc_Histograms"
```

## 2.4 Spline fits of treatment effects over temperature

In order to access whether the drug treatment has a significant impact on altering the thermal stability of specific proteins a function was implemented which illustrates the course of stability of a certain protein over different temperatures based on a reference data set. A natural cubic spline fitted to the reference data is then used to infer the relative stability curves of proteins with different concentrations of treatment which are in turn fitted by natural cubic splines. The cubic spline with  $n$  degrees of freedom on  $[a, b]$  obeys:

- $S(x) \in C^2[a, b]$
- $a = t_0 < t_1 < \dots < t_n = b$



and:

$$S(x) = \begin{cases} S_0(x) = a_0x^3 + b_0x^2 + c_0x + d_0, & t_0 \leq x \leq t_1 \\ S_1(x) = a_1x^3 + b_1x^2 + c_1x + d_1, & t_1 \leq x \leq t_2 \\ \cdot \\ \cdot \\ \cdot \\ S_{n-1}(x) = a_{n-1}x^3 + b_{n-1}x^2 + c_{n-1}x + d_{n-1}, & t_{n-1} \leq x \leq t_n \end{cases} \quad \mathbf{1}$$

a *natural cubic spline* additionally constrains that it's function has to be linear beyond the boundary knots with constrains that both the first and the last section of the cubic spline has to be linear.

The function to perform this analysis can be invoked by:

```
analysisResults <- tpp2dSplineFitAndTest(data = normData2d,
                                         dataRef = trRef,
                                         refIDVar = "Protein_ID",
                                         refFcStr = "norm_rel_fc_protein_",
                                         doPlot = FALSE,
                                         resultPath = resultPath,
                                         nCores = 1)

head(analysisResults)

## representative qupm qusm clustername sumionarea_protein_5
## 1 IPI00000001.2 15 25 STAU1 1193994914
## 2 IPI00000001.2 15 25 STAU1 1272771185
## 3 IPI00000001.2 13 22 STAU1 1482437522
## 4 IPI00000001.2 13 22 STAU1 1157290962
## 5 IPI00000001.2 15 24 STAU1 396823892
## 6 IPI00000001.2 15 24 STAU1 345169960
## sumionarea_protein_1 sumionarea_protein_0.143 sumionarea_protein_0.02
## 1 1337957734 1375948494 1956350223
## 2 1473572092 1273285951 1669312103
## 3 1513181000 1284434575 1487032006
## 4 1050288621 1110810226 1128507681
## 5 458022616 453860821 412257039
## 6 350182409 352193788 344410388
## sumionarea_protein_0 temperature experiment unique_ID
## 1 1801848318 42.0 X020466 X020466_42_IPI00000001.2
## 2 1404292404 44.1 X020466 X020466_44.1_IPI00000001.2
## 3 1422365645 46.2 X020467 X020467_46.2_IPI00000001.2
## 4 999666282 48.1 X020467 X020467_48.1_IPI00000001.2
## 5 439399665 50.4 X020468 X020468_50.4_IPI00000001.2
## 6 309019704 51.9 X020468 X020468_51.9_IPI00000001.2
## rel_fc_5 rel_fc_1 rel_fc_0.143 rel_fc_0.02 rel_fc_0 norm_rel_fc_5
## 1 0.6626501 0.7425474 0.7636317 1.0857463 1 1.107187
## 2 0.9063434 1.0493342 0.9067100 1.1887212 1 1.114453
## 3 1.0422338 1.0638481 0.9030270 1.0454640 1 1.187727
```

```
## 4 1.1576773 1.0506392 1.1111810 1.1288844 1 1.249516
## 5 0.9031047 1.0423827 1.0329112 0.9382279 1 1.123552
## 6 1.1169837 1.1332041 1.1397130 1.1145257 1 1.171933
## norm_rel_fc_1 norm_rel_fc_0.143 norm_rel_fc_0.02 norm_rel_fc_0 F_statistic
## 1 1.059331 1.105019 1.244416 1 6.006917
## 2 1.164559 1.022695 1.195813 1 6.006917
## 3 1.229422 1.078735 1.211522 1 6.006917
## 4 1.147406 1.108487 1.329434 1 6.006917
## 5 1.268366 1.267164 1.256324 1 6.006917
## 6 1.176446 1.158041 1.163566 1 6.006917
## F_moderated F_scaled residual_df_H1 prior_df_H1 df1 df2 df2_moderated
## 1 188.3243 9.416217 35 2.581745 20 35 37.58174
## 2 188.3243 9.416217 35 2.581745 20 35 37.58174
## 3 188.3243 9.416217 35 2.581745 20 35 37.58174
## 4 188.3243 9.416217 35 2.581745 20 35 37.58174
## 5 188.3243 9.416217 35 2.581745 20 35 37.58174
## 6 188.3243 9.416217 35 2.581745 20 35 37.58174
## posterior_var_H1 p_NPARC p_adj_NPARC
## 1 0.0003455887 3.140386e-09 3.50153e-07
## 2 0.0003455887 3.140386e-09 3.50153e-07
## 3 0.0003455887 3.140386e-09 3.50153e-07
## 4 0.0003455887 3.140386e-09 3.50153e-07
## 5 0.0003455887 3.140386e-09 3.50153e-07
## 6 0.0003455887 3.140386e-09 3.50153e-07
```

Moreover, these fits can be used then, in order to access confidence on whether the curves fitting the relative treatment data points represent the data better than a model which does not distinguish between the different treatment concentrations. The confidence assessment is thereby based on a moderated F statistic adapted from a method by Storey and others [6] which they developed for microarray time course data. The method calculates a moderated F statistic following:

$$F = \frac{SS_0 - SS_1}{\hat{s}^2(\sigma^2, df_2)} \quad 2$$

with  $SS_0$  representing the sum of squares of the null model (fitting the data without distinguishing between different treatment concentrations) and  $SS_1$  those of the full model (which fits the data by in this case 5 different splines for every treatment concentration respectively). With  $\hat{s}^2$  representing the empirical Bayes estimator for  $SS_1$ , with  $df_2 = n - \nu_1$ , where  $\nu_1$  denoted the parameters of the full model and  $n$  denotes the number of data points.

```
analysisResults %>% filter(representative == IPI_id_HDAC2) %>%
  select(temperature, p_NPARC, p_adj_NPARC)

## temperature p_NPARC p_adj_NPARC
## 1 42.0 0 0
## 2 44.1 0 0
## 3 46.2 0 0
## 4 48.1 0 0
## 5 50.4 0 0
## 6 51.9 0 0
## 7 54.0 0 0
## 8 56.1 0 0
## 9 58.2 0 0
```

## TPP

## 10	60.1	0	0
## 11	62.4	0	0
## 12	63.9	0	0

By defining the `methods` argument to include `"splineFit"`, one prompts the main function `analyze2DTPP` to directly perform spline fits and a moderated F-test for each protein in the data set.

## References

- [1] Daniel Martinez Molina, Rozbeh Jafari, Marina Ignatushchenko, Takahiro Seki, E Andreas Larsson, Chen Dan, Lekshmy Sreekumar, Yihai Cao, and Paer Nordlund. Monitoring drug target engagement in cells and tissues using the cellular thermal shift assay. *Science*, 341(6141):84–7, 2013.
- [2] Mikhail M Savitski, Friedrich BM Reinhard, Holger Franken, Thilo Werner, Maria Fälth Savitski, Dirk Eberhard, Daniel Martinez Molina, Rozbeh Jafari, Rebecca Bakszt Dovega, Susan Klaeger, et al. Tracking cancer drugs in living cells by thermal profiling of the proteome. *Science*, 346(6205):1255784, 2014.
- [3] Isabelle Becher, Thilo Werner, Carola Doce, Esther A Zaal, Cecilia R Berkers, Ina Tögel, Elsa Salzer, Marcus Bantscheff, and Mikhail M Savitski. Comprehensive thermal and chemoproteomics profiling identifies phenylalanine hydroxylase as a potent off-target of the histone deacetylase inhibitor panobinostat. *in submission*, 2016.
- [4] Holger Franken, Toby Mathieson, Dorothee Childs, Gavain Sweetman, Thilo Werner, Wolfgang Huber, and Mikhail M Savitski. Thermal proteome profiling for unbiased identification of drug targets and detection of downstream effectors. *Nature protocols*, 10(10):1567 – 1593, 2015.
- [5] Alexander Walker. *openxlsx: Read, Write and Edit XLSX Files*, 2015. R package version 2.4.0. URL: <http://CRAN.R-project.org/package=openxlsx>.
- [6] John D Storey, Wenzhong Xiao, Jeffrey T Leek, Ronald G Tompkins, and Ronald W Davis. Significance analysis of time course microarray experiments. *Proceedings of the National Academy of Sciences of the United States of America*, 102(36):12837–42, 2005. URL: <http://www.pnas.org/content/102/36/12837.abstract>, doi:10.1073/pnas.0504609102.