Filtering

In [1]:

```
import pandas as pd
import numpy as np
import gc
import matplotlib.pyplot as plt
import os, sys
sys.path.insert(1, os.path.relpath("../functions"))

from filtering import *
from plotting import *

_FOLDER = "../data/"
_FOLDER_2 = "../figures/"
_FOLDER_3 = "../results/"
SAVE_FIGURES = False
```

Original data

In [2]:

```
drug_curves = pd.read_csv(_FOLDER+"normalised_dose_response_data.csv")

if "Unnamed: 0" in drug_curves:
    drug_curves.drop("Unnamed: 0", axis=1, inplace =True)

col_to_drop = ["per_slope_change_"+str(i) for i in range(8)]+W
            ["slope_" + str(i) for i in range(9)]
drug_curves.drop(col_to_drop, axis=1, inplace=True)

conc_columns= ["fd_num_"+str(i) for i in range(10)]
response_norm = ['norm_cells_'+str(i) for i in range(10)]
CCL_names = dict(zip(drug_curves["COSMIC_ID"], drug_curves["CELL_LINE_NAME"]))
df= pd.read_csv(_FOLDER+'Drug_Features.csv')
drug_names = dict(zip(df["Drug ID"].values, df["Drug Name"].values))
del df
drug_curves["drug_name"] = drug_curves["DRUG_ID"].map(drug_names)
drug_curves["CCL_name"] = drug_curves["COSMIC_ID"].map(CCL_names)
drug_curves.shape
```

Out[2]:

```
(225384, 28)
```

Filtering 1: auc>0.7 and spearman_r<0

```
def AucFitration(df, auc_limit=0.7):
    """
    1. Remove all the curves where the normalised response value is greater than one at zero
dosage.
    2. Compute the Area Under the Curve (AUC) for all the curves.
    3. Leave only those curves with an AUC>0.7.
```

```
    4. Compute the Spearman correlation coefficient between the normalised response and the
scaled dosage
    (so the x-axis and the y-axis).
    5. Further remove the curves for which the Spearman correlation coefficient is zero or
positive.
    """
```

In [3]:

```
%%time
df_filt = auc_filtration(drug_curves, conc_columns, response_norm, auc_limit=0.7, save_file_name=
df_filt.to_csv("filtered_data_auc_spearman.csv", index=False)
```

```
 85%|████████▌ | 191643/225384 [11:49<02:00, 278.96it/s]/home/marina/anaconda3/li
b/python3.7/site-packages/sklearn/metrics/ranking.py:114: RuntimeWarning: invalid
value encountered in less
  if np.any(dx < 0):
100%|██████████| 225384/225384 [13:26<00:00, 279.46it/s]

CPU times: user 13min 35s, sys: 2.25 s, total: 13min 38s
Wall time: 13min 33s
```

In [3]:

```
df_filt = pd.read_csv(_FOLDER_3+"data_with_auc.csv")
df_filt = df_filt[(df_filt["auc"]>0.7) & (df_filt["spearman_r"]<0)].copy()
df_filt.shape
```

Out[3]:

```
(122642, 30)
```

In [4]:

```
df_filt.head()
```

Out[4]:

	CELL_LINE_NAME	COSMIC_ID	DRUG_ID	DRUGID_COSMICID	FOLD_DILUTION	MAX_COI
2	HDQ-P1	1290922	245	245_1290922	2	40.0
3	HDQ-P1	1290922	155	155_1290922	2	0.5
5	HDQ-P1	1290922	134	134_1290922	2	16.0
6	HDQ-P1	1290922	310	310_1290922	2	5.1
7	HDQ-P1	1290922	306	306_1290922	2	10.2

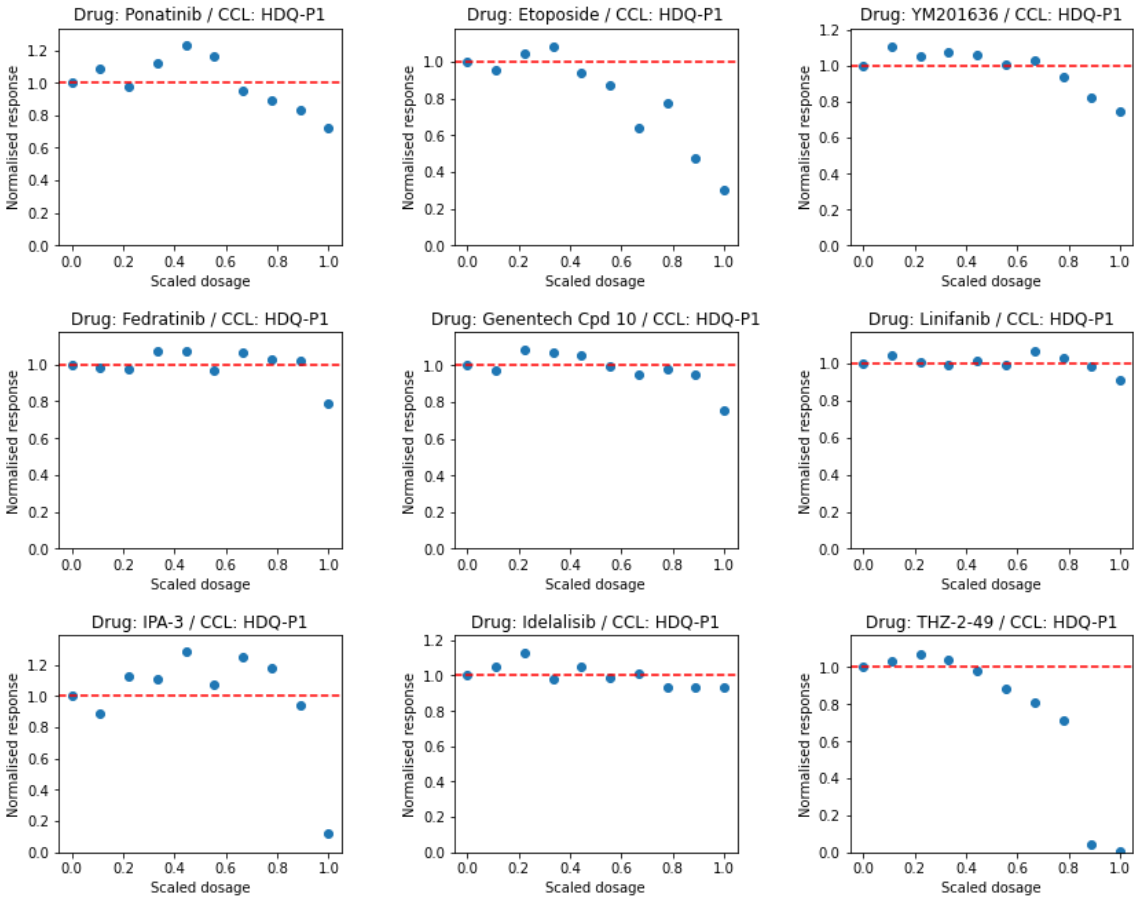
5 rows × 30 columns

In [5]:

```
high_resp_data = find_high_responses(df_filt, response_norm)
print("Responses above 1:", high_resp_data.shape[0])

df = high_resp_data
show_response_curves(df, plots_in_row=3, plots_in_column=3, ₩
                     x_columns=conc_columns, y_columns=response_norm, indexes=df.index[:9],
                    drug_dict = drug_names, CCL_dict = CCL_names, upper_limit=1)
```

Responses above 1: 75014



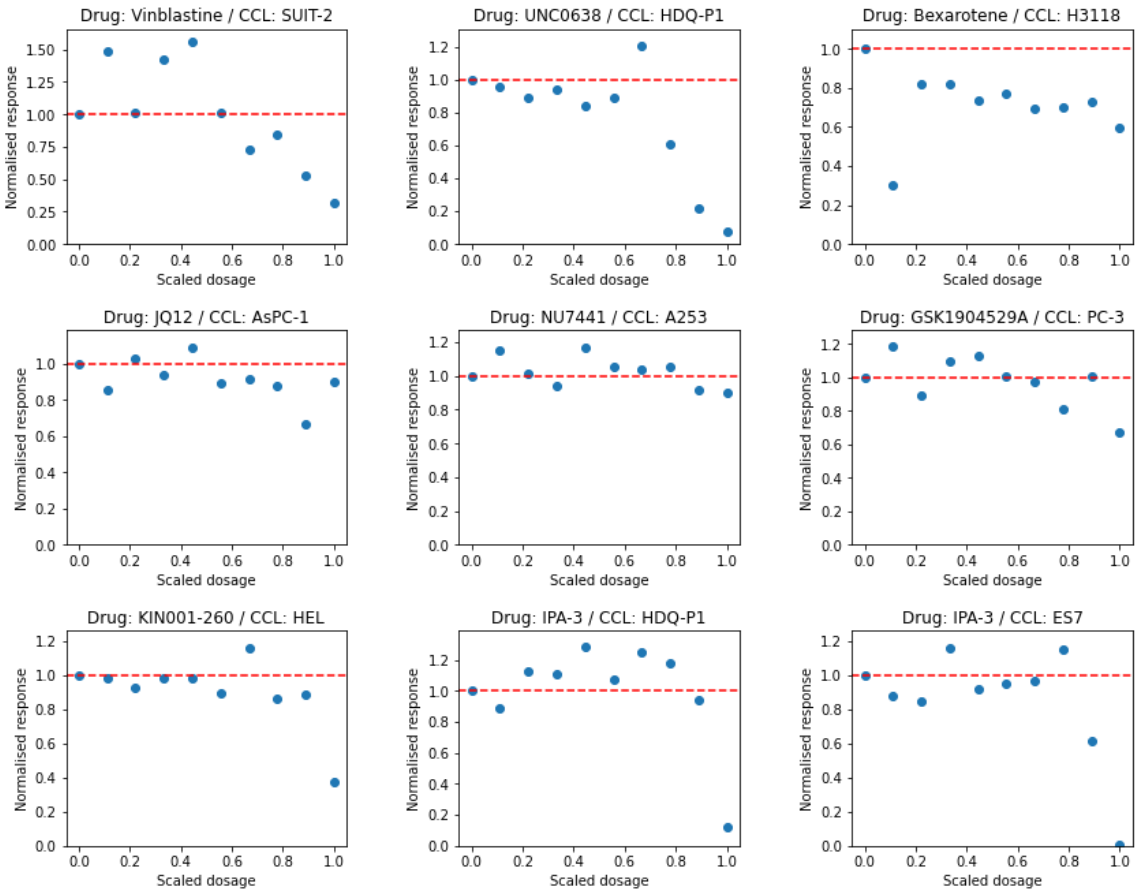
Bad data left after filtering 1

In [6]:

```
ascend_data = find_ascending_data(df_filt, response_norm, middle_points_limit=-0.2)
print("Ascending points:", ascend_data.shape[0])

df = ascend_data
show_response_curves(df, plots_in_row=3, plots_in_column=3, ₩
                     x_columns=conc_columns, y_columns=response_norm, indexes=df.index[:9],
                    drug_dict = drug_names, CCL_dict = CCL_names, upper_limit=1)
```

```
Ascending points: 20081
```



Filtering 2: 3 stage filtering as in MSc project

```
def filtering_sigmoid_curves(df, response_columns, filtering_scenario = [1,2,3],
                     first_columns_to_compare = [1, 2], last_columns_to_compare = [-1, -2],
                     tolerance=0.05,
                     first_points_lower_limit = 0.8,
                     last_points_upper_limit = 0.4,
                      middle_points_limit = 0.1):
    """
    filtering_scenario = [1,2,3,4]
    1. Ensure that all the response are less than 1

    2. Ensure that first and last points form plateus
    the minimal number of points are specified in the function arguments
    by default, two points for both lpateus are considered
    tolerance =0.05 values to ensure the points form a plateu
```

```
    first_columns_to_compare = [1, 2]  - first two columns for plateu
    last_columns_to_compare = [-1, -2] - last two columns for plateu

    3. Specify location of the plateus - first_points_lower_limit and last_points_upper_limit

    4. Cutting off ambiqueos data:
    Among all "middle" datapoints a subsequent point should not be higher than antecedent by
0.2
    if col1 = 1.0 & col2 = 1.2 middle_points_dif = -0.2
    if we want to cut off such data we set middle_points_limit =-0.2
    """
```

In [7]:

```
df_filt_123 = filtering_sigmoid_curves(drug_curves, filtering_scenario=[1,2,3], ₩
                        response_columns = response_norm, ₩
                        first_points_lower_limit = 0.8, last_points_upper_limit = 0.4)
```

```
Original dataset: (225384, 28)
1st filtration (Ensure that all the response are less than 1): Filtered dataset:
(63325, 28)
2d filtration (Ensure that first and last points form plateus): Filtered dataset:
(6321, 30)
3d stage filtration (Specified location of the plateus): Filtered dataset: (2776,
30)
```

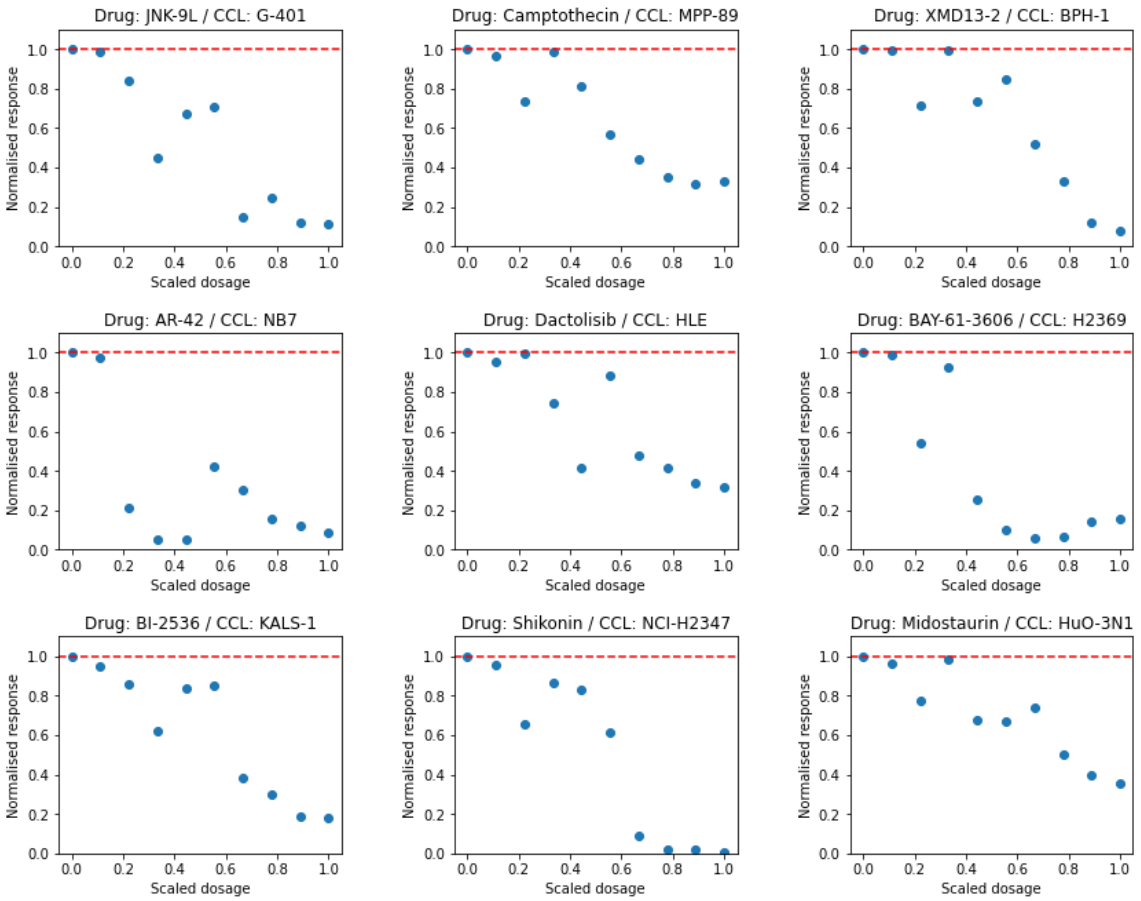
Bad data left after filtering 2

In [8]:

```
ascend_data = find_ascending_data(df_filt_123, response_norm, middle_points_limit=-0.2)
print("Ascending points:", ascend_data.shape[0])

df = ascend_data
show_response_curves(df, plots_in_row=3, plots_in_column=3, ₩
                     x_columns=conc_columns, y_columns=response_norm, indexes=df.index[:9],
                    drug_dict = drug_names, CCL_dict = CCL_names, upper_limit=1)
```

```
Ascending points: 57
```



Filtering 3: 4 stage filtering

In [9]:

```
df_filt_1234 = filtering_sigmoid_curves(drug_curves, filtering_scenario=[1,2,3,4], ₩
                        response_columns = response_norm, ₩
                        first_points_lower_limit = 0.8, last_points_upper_limit = 0.4,
                         middle_points_limit = -0.2)
```

```
Original dataset: (225384, 28)
1st filtration (Ensure that all the response are less than 1): Filtered dataset:
(63325, 28)
2d filtration (Ensure that first and last points form plateus): Filtered dataset:
(6321, 30)
3d stage filtration (Specified location of the plateus): Filtered dataset: (2776,
30)
4th stage filtration (Cut off high ancedent points): Filtered dataset: (2719, 30)
```

In [10]:

```
df_filt_1234 = filtering_sigmoid_curves(drug_curves, filtering_scenario=[1,2,3,4], ₩
                        response_columns = response_norm, ₩
                        first_points_lower_limit = 0.8, last_points_upper_limit = 0.4,
                         middle_points_limit = -0.1)
```

```
Original dataset: (225384, 28)
1st filtration (Ensure that all the response are less than 1): Filtered dataset:
(63325, 28)
2d filtration (Ensure that first and last points form plateus): Filtered dataset:
(6321, 30)
3d stage filtration (Specified location of the plateus): Filtered dataset: (2776,
30)
4th stage filtration (Cut off high ancedent points): Filtered dataset: (2600, 30)
```

Complement Filtering 1 (with auc and spearman) with cutting off outliers

In [11]:

```
df_filt.shape
```

Out[11]:

```
(122642, 30)
```

In [12]:

```
df_filt_outl_02 = cut_off_outliers(df_filt, response_norm, middle_points_limit = -0.2)
df_filt_outl_02.shape
```

Out[12]:

```
(102561, 30)
```

In [13]:

```
df_filt_outl_01= cut_off_outliers(df_filt, response_norm, middle_points_limit = -0.1)
df_filt_outl_01.shape
```

Out[13]:

```
(72581, 30)
```

Conclusion - amount of data left

In [14]:

```
print("MSc filtering (3 stages for perfect sigmoid):", df_filt_123.shape[0])
print("4 stage filtering (with cutting off outliers):", df_filt_1234.shape[0])
```

```
MSc filtering (3 stages for perfect sigmoid): 2776
4 stage filtering (with cutting off outliers): 2600
```

In [15]:

```
print("Filtering with auc>0.7 and spearman<0:", df_filt.shape[0])
print("Filtering with auc>0.7 and spearman<0 & cutting off outliers:", df_filt_outl_02.shape[0])
```

```
Filtering with auc>0.7 and spearman<0: 122642
Filtering with auc>0.7 and spearman<0 & cutting off outliers: 102561
```

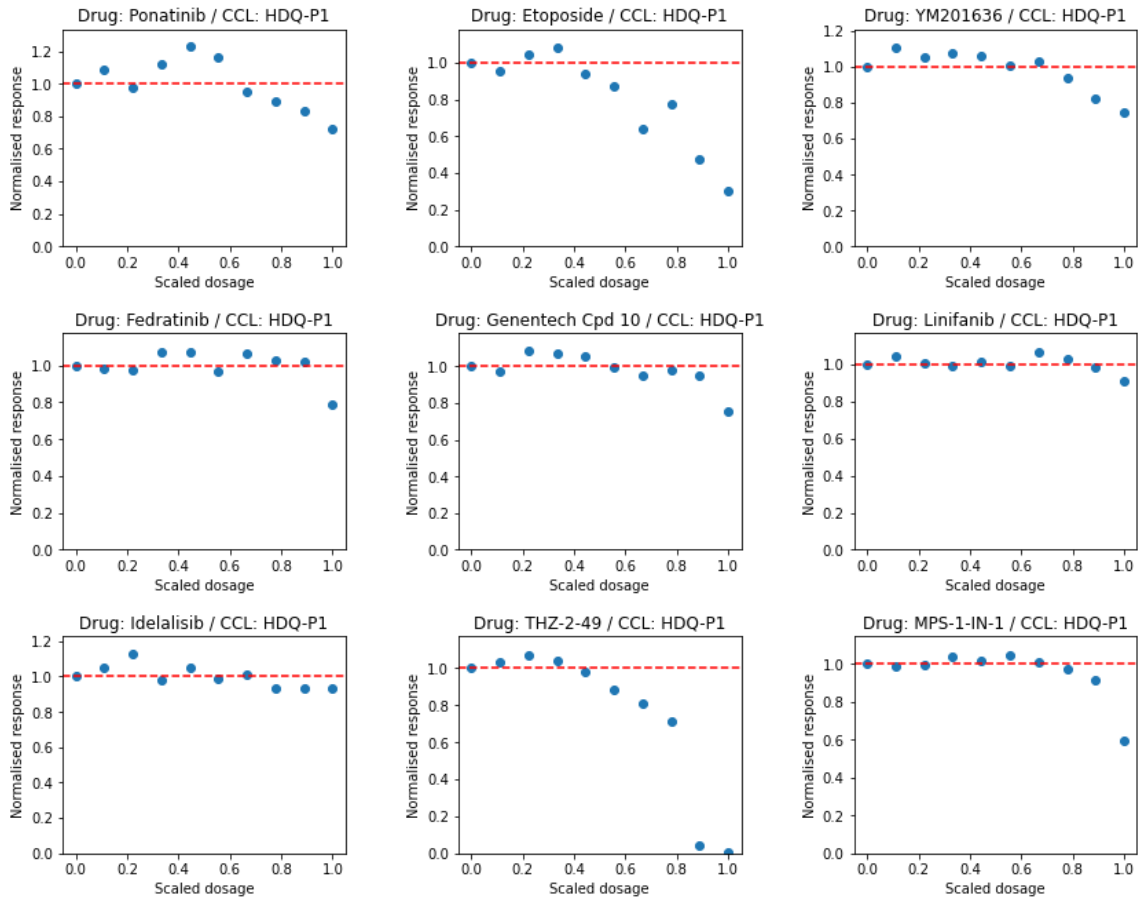
Some data left after Filtering with auc>0.7 and spearman<0 & cutting off outliers

In [16]:

```
print(df_filt_outl_02.shape)
high_resp_data = find_high_responses(df_filt_outl_02, response_norm)
print("Responses above 1:", high_resp_data.shape[0])

df = high_resp_data
show_response_curves(df, plots_in_row=3, plots_in_column=3, ₩
                     x_columns=conc_columns, y_columns=response_norm, indexes=df.index[:9],
                    drug_dict = drug_names, CCL_dict = CCL_names, upper_limit=1)
```

(102561, 30)
Responses above 1: 59284



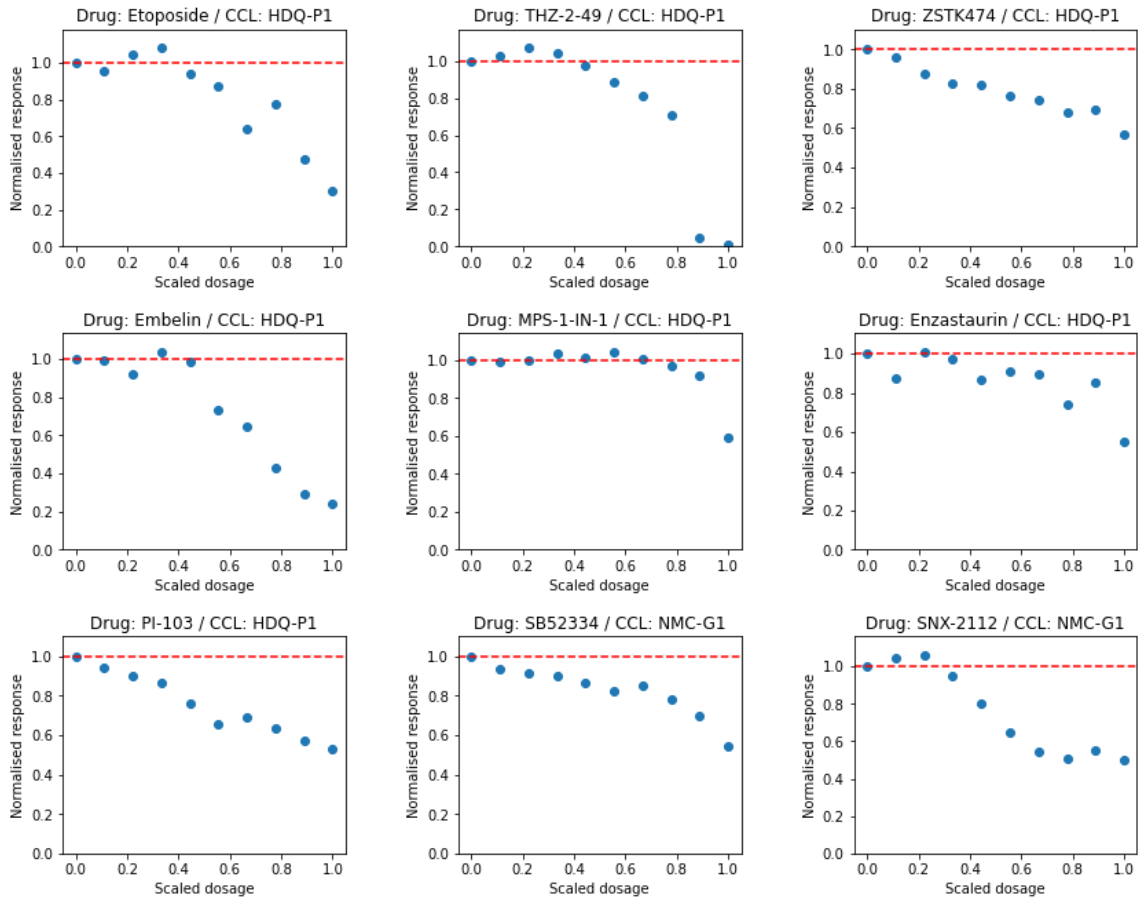
Leave the data where significant final response was observed

In [17]:

```
print("Filtered data:", df_filt_outl_02.shape[0])
df_good_resp_06 = filter_good_response(df_filt_outl_02, response_norm, final_response_limit=0.6)
print("Good responses", df_good_resp_06.shape[0])

df = df_good_resp_06
show_response_curves(df, plots_in_row=3, plots_in_column=3, ₩
                     x_columns=conc_columns, y_columns=response_norm, indexes=df.index[:9],
                    drug_dict = drug_names, CCL_dict = CCL_names, upper_limit=1)
```

```
Filtered data: 102561
Good responses 41613
```

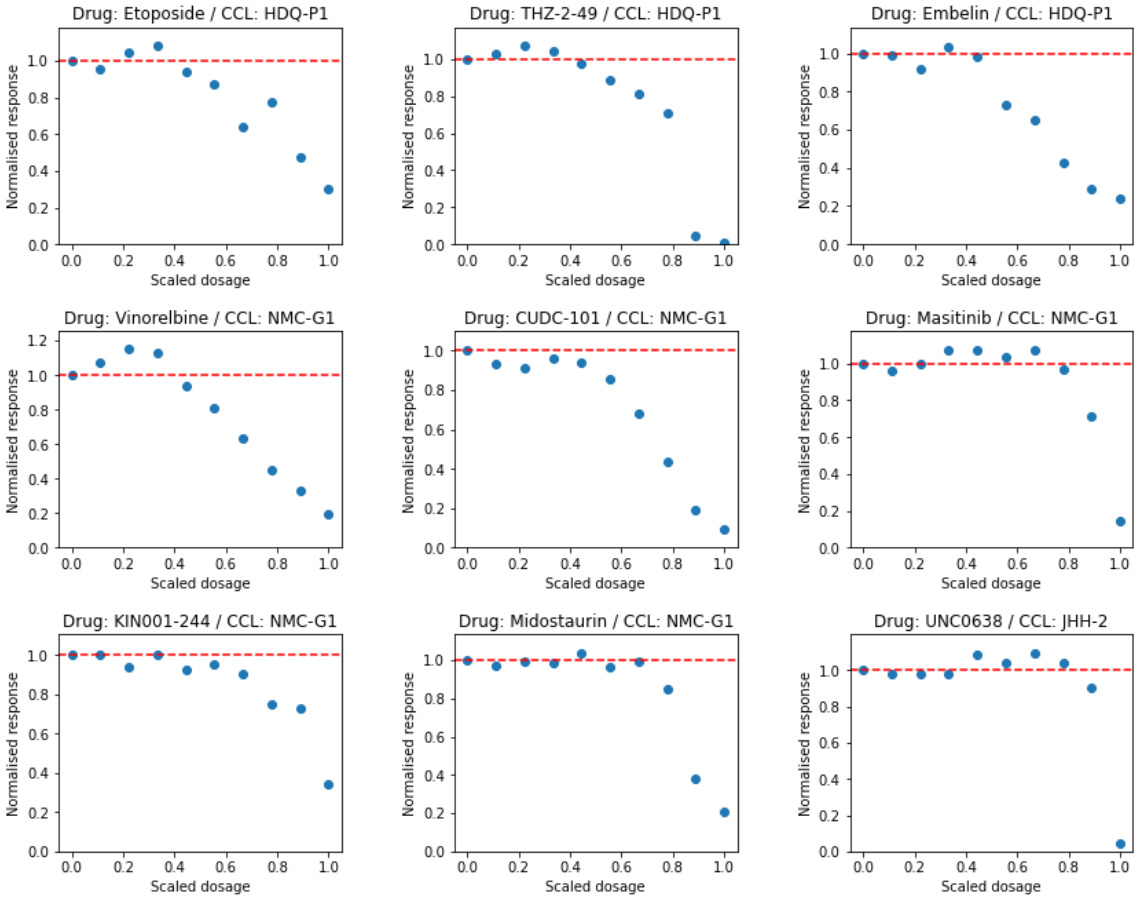


In [18]:

```
print("Filtered data:", df_filt_outl_02.shape[0])
df_good_resp_04 = filter_good_response(df_filt_outl_02, response_norm, final_response_limit=0.4)
print("Good responses", df_good_resp_04.shape[0])

df = df_good_resp_04
show_response_curves(df, plots_in_row=3, plots_in_column=3, ₩
                     x_columns=conc_columns, y_columns=response_norm, indexes=df.index[:9],
                    drug_dict = drug_names, CCL_dict = CCL_names, upper_limit=1)
```

```
Filtered data: 102561
Good responses 24164
```

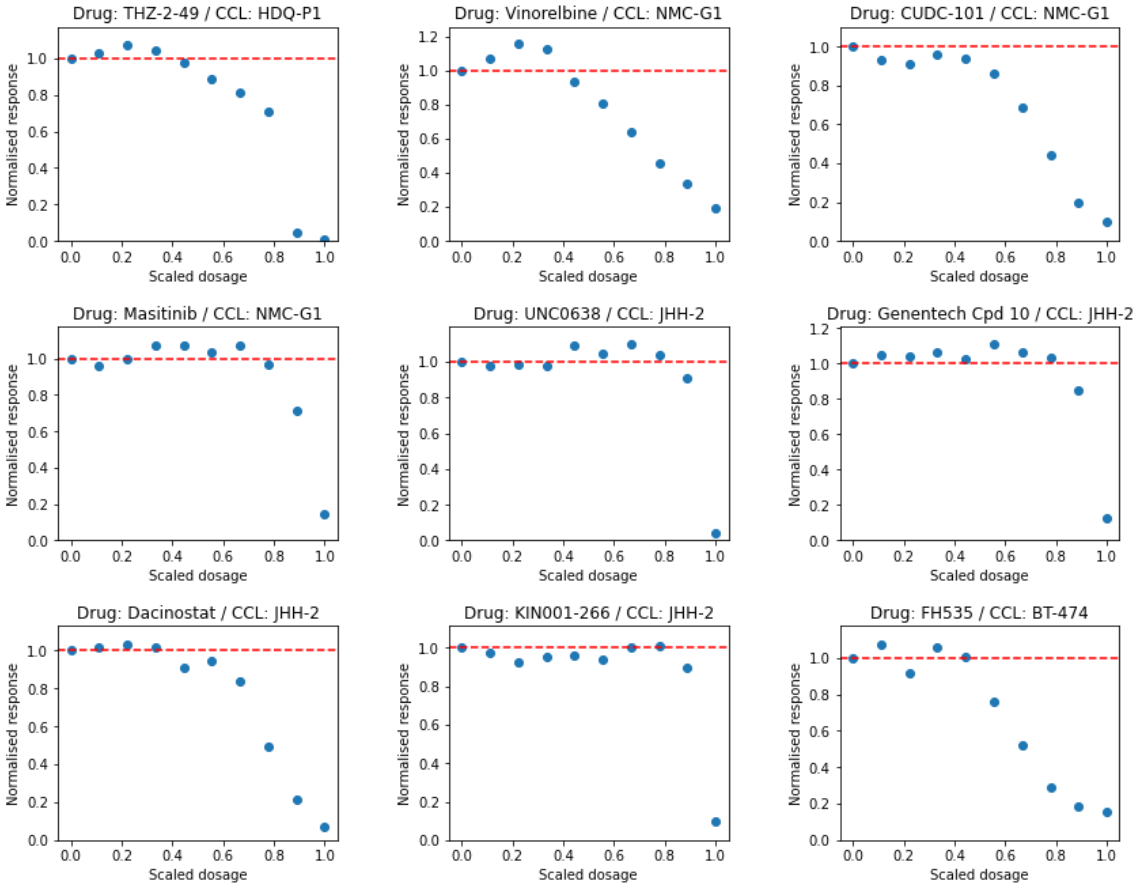


In [19]:

```
print("Filtered data:", df_filt_outl_02.shape[0])
df_good_resp_02 = filter_good_response(df_filt_outl_02, response_norm, final_response_limit=0.2)
print("Good responses", df_good_resp_02.shape[0])

df = df_good_resp_02
show_response_curves(df, plots_in_row=3, plots_in_column=3, ₩
                     x_columns=conc_columns, y_columns=response_norm, indexes=df.index[:9],
                    drug_dict = drug_names, CCL_dict = CCL_names, upper_limit=1)
```

```
Filtered data: 102561
Good responses 12169
```

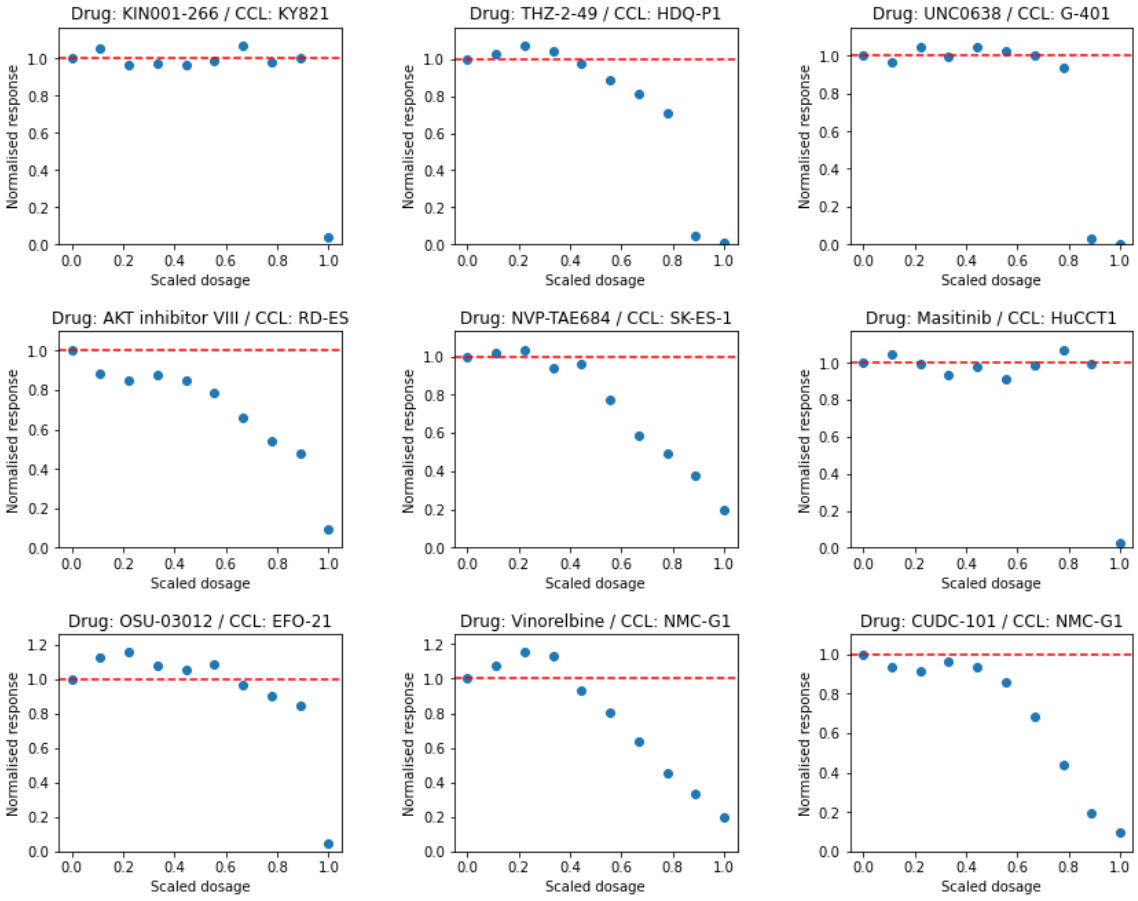


In [20]:

```
print("Filtered data:", df_filt_outl_01.shape[0])
df_good_resp_02_2 = filter_good_response(df_filt_outl_01, response_norm, final_response_limit=0.2
print("Good responses", df_good_resp_02_2.shape[0])

df = df_good_resp_02_2
show_response_curves(df, plots_in_row=3, plots_in_column=3, W
                     x_columns=conc_columns, y_columns=response_norm, indexes=df.index[:9],
                    drug_dict = drug_names, CCL_dict = CCL_names, upper_limit=1)
```

```
Filtered data: 72581
Good responses 9287
```



In [21]:

```
df_filt.to_csv(_FOLDER_3+"/filt_auc.csv", index=False)
```

In [22]:

```
df_good_resp_02.to_csv(_FOLDER_3 + "/filt_auc_02.csv", index=False)
df_good_resp_04.to_csv(_FOLDER_3 + "/filt_auc_04.csv", index=False)
df_good_resp_06.to_csv(_FOLDER_3 + "/filt_auc_06.csv", index=False)
```

In [23]:

```
df_filt_123.to_csv(_FOLDER_3 + "/filt_123.csv",index=False)
df_filt_1234.to_csv(_FOLDER_3 + "/filt_1234.csv",index=False)
```

In [ ]:

```
```