

supplement_notebook_4_agn

December 10, 2019

1 Exploring enrichment analysis of HL60 response to bendamustine.

This is the second notebook for the Pino Et. Al. In this notebook we demonstrate how MAGINE can be used to explore the enrichment analysis.

```
[1]: from IPython.display import display, Image
%matplotlib inline
import networkx as nx
import matplotlib.pyplot as plt
import pandas as pd
pd.set_option('display.precision', 2)
pd.set_option('display.max_colwidth', 50)
%load_ext autoreload
%autoreload 2

[2]: # load magine specific tools
from magine.plotting.wordcloud_tools import create_wordcloud
from magine.plotting.heatmaps import heatmap_by_terms
from magine.plotting.venn_diagram_maker import create_venn3, create_venn2

from magine.enrichment import load_enrichment_csv
from magine.networks import visualization as vis
from magine.networks import utils, exporters

from magine.networks.annotated_set import create_subnetwork
from magine.networks.subgraphs import Subgraph
```

2 Exploring enrichment output

2.1 Loading data and networks

```
[3]: from exp_data import exp_data
```

Load enrichment array. This bendamustine_enrichment.csv.gz was created by run_enrichment.py script. If it doesn't exist, run that file to generate the results. Due to the number of samples, we run this outside a Jupyter notebook as it can take quite a bit of time.

```
[4]: enrichment_array = load_enrichment_csv('Data/bendamustine_enrichment.csv.gz', index_col=0)

enrichment_array['significant'] = False
enrichment_array.loc[(enrichment_array['adj_p_value'] <= 0.05) &
                     (enrichment_array['combined_score'] > 0.0),
                     'significant'] = True
# Remove terms that are not significant in at least one time point/sample/
# →category
enrichment_array.require_n_sig(
    columns='sample_id',
    index='term_name',
    n_sig=1,
    inplace=True
)
```

```
[5]: enrichment_array[['term_name', 'db', 'category']].nunique()
```

```
[5]: term_name      20758
      db            52
      category       15
      dtype: int64
```

```
[6]: display(enrichment_array.head(5))
```

	term_name	rank	combined_score	adj_p_value	\
0	vinblastine-up	1	34.53	2.83e-05	
1	mitotane-up	2	31.90	3.64e-05	
2	dideoxycytidine-dn	3	30.15	1.38e-04	
3	busulfan-dn	4	29.18	1.01e-04	
4	mitotane-up	5	28.76	1.39e-04	

	genes	n_genes	sample_id	\
0	ADSS, APOA1, APOE, BSG, BTF3, CANX, COPA, DNAJC9, EI24...	24	000030_s	
1	ABI1, ACLY, ALDH3A2, BTF3, CD97, COL4A3BP, COPA, GIT2...	23	000030_s	
2	ACADVL, ACLY, ALDH3A2, APOA1, APOA2, APOE, BSG, CANX, ...	23	000030_s	
3	ABI1, BTF3, CANX, COPA, DNAJC9, FBXW5, FLNA, HMHA1, HP...	22	000030_s	
4	ACLY, ADSS, AKAP8L, BTF3, CANX, CD97, COL4A3BP, COPA, ...	21	000030_s	

	category	db	significant
0	proteomics_both	DrugMatrix	True
1	proteomics_both	DrugMatrix	True
2	proteomics_both	DrugMatrix	True
3	proteomics_both	DrugMatrix	True
4	proteomics_both	DrugMatrix	True

```
[7]: # clean up printing by selecting fewer columns
cols = ['term_name', 'rank', 'combined_score', 'adj_p_value',
```

```
'n_genes', 'sample_id', 'category']

[8]: display(enrichment_array[cols].head(5))
```

	term_name	rank	combined_score	adj_p_value	n_genes	sample_id	\
0	vinblastine-up	1	34.53	2.83e-05	24	000030_s	
1	mitotane-up	2	31.90	3.64e-05	23	000030_s	
2	dideoxycytidine-dn	3	30.15	1.38e-04	23	000030_s	
3	busulfan-dn	4	29.18	1.01e-04	22	000030_s	
4	mitotane-up	5	28.76	1.39e-04	21	000030_s	

	category
0	proteomics_both
1	proteomics_both
2	proteomics_both
3	proteomics_both
4	proteomics_both

Load network Load in the network and initialize Subgraph. We will use this later to construct networks from queries.

```
[9]: network = nx.read_gpickle('Networks/bendamustine_network_w_attributes.p')
net_sub = Subgraph(network)
```

3 Single database exploration

Here we will focus on the Reactome enrichment.

```
[10]: reactome_only = enrichment_array.filter_multi(
    db='Reactome_2016', # Only reactome db
)
# This just cleans up the term name
display(reactome_only['term_name'].head(5))
reactome_only['term_name'] = reactome_only['term_name'].str.split('_').str.
    get(0)
display(reactome_only['term_name'].head(5))
```

```
80551    processing of capped intron-containing pre-mrn...
80552                  gene expression_hsa_r-hsa-74160
80553                  mrna splicing_hsa_r-hsa-72172
80554      mrna splicing - major pathway_hsa_r-hsa-72163
80555      transport of mature transcript to cytoplasm_hs...
Name: term_name, dtype: object
```

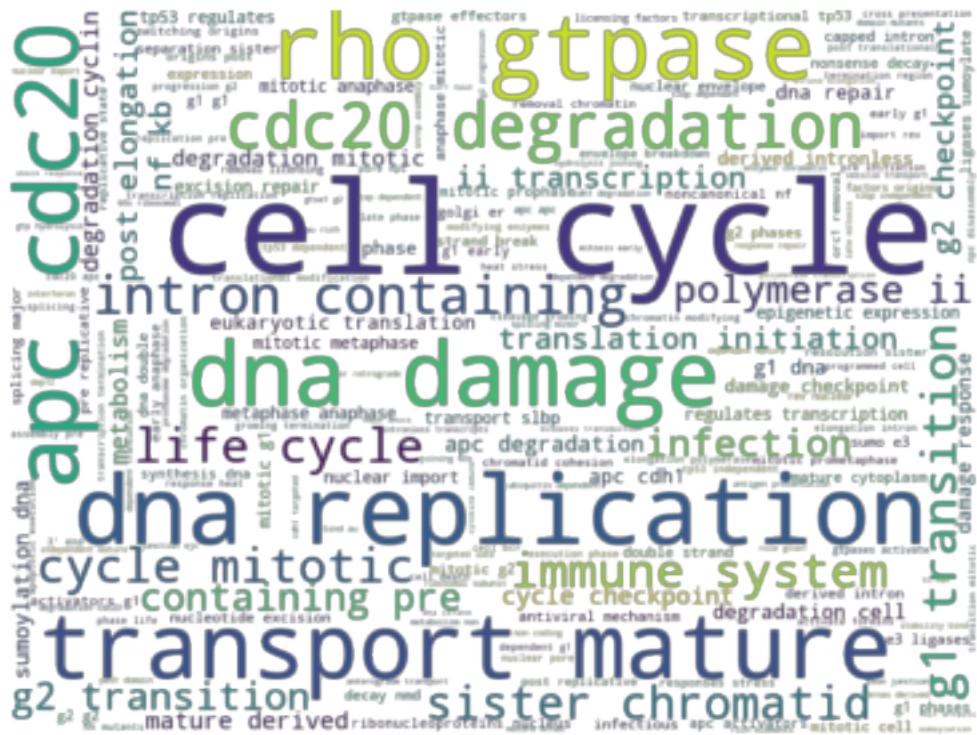
```
80551    processing of capped intron-containing pre-mrna
80552      gene expression
```

```

80553          mrna splicing
80554          mrna splicing - major pathway
80555          transport of mature transcript to cytoplasm
Name: term_name, dtype: object

```

```
[11]: # we can use a word cloud to view what terms are enriched
word_cloud = create_wordcloud(reactome_only.sig)
word_cloud.plot();
```



```
[12]: word_cloud.data.head(20)
```

	words	counts
181	cell cycle	325
208	DNA replication	160
249	rho gtpase	142
243	DNA damage	139
179	transport mature	138
586	apc cdc20	132
587	cdc20 degradation	122
175	intron containing	121
353	immune system	98
199	life cycle	97
228	cycle mitotic	95

```

381      g1 transition      94
355      sister chromatid   86
176      containing pre    80
236      polymerase ii     79
400      g2 transition     79
5          infection       77
429  translation initiation 77
237      ii transcription   75
261      post elongation   73

```

[]:

3.1 Phospho-SILAC enrichment

3.1.1 Filtering enrichment output

```

[13]: # subset the data to only look at ph-silac data.
# Later on we will look at label-free, then both.

ph_silac = reactome_only.filter_multi(
    category=['ph_silac_up', 'ph_silac_down'],
)

# arbitrary value that can be easily changed.
ph_silac.require_n_sig(n_sig=3, inplace=True)

not_useful = [
    'gene expression', 'translation',
    'immune system',
    'disease', 'diseases of signal transduction',
    'infectious disease',
    'influenza infection', 'influenza life cycle',
    'influenza viral rna transcription and replication',
]
ph_silac = ph_silac.loc[~ph_silac['term_name'].isin(not_useful)]
ph_silac_copy = ph_silac.copy()

ph_silac.remove_redundant(
    threshold=.5,
    level='dataframe',
    sort_by='combined_score',
    inplace=True
)

```

Number of rows went from 159 to 29

```
[14]: ph_silac.heatmap(  
    convert_to_log=False,  
    cluster_by_set=False,  
    cluster_row=False,  
    values='combined_score',  
    columns=['category', 'sample_id'],  
    annotate_sig=True,  
    div_colors=True,  
    linewidths=.005,  
    figsize=(5,12)  
);  
plt.savefig("ph_silac_enriched.png", dpi=300, bbox_inches='tight')
```



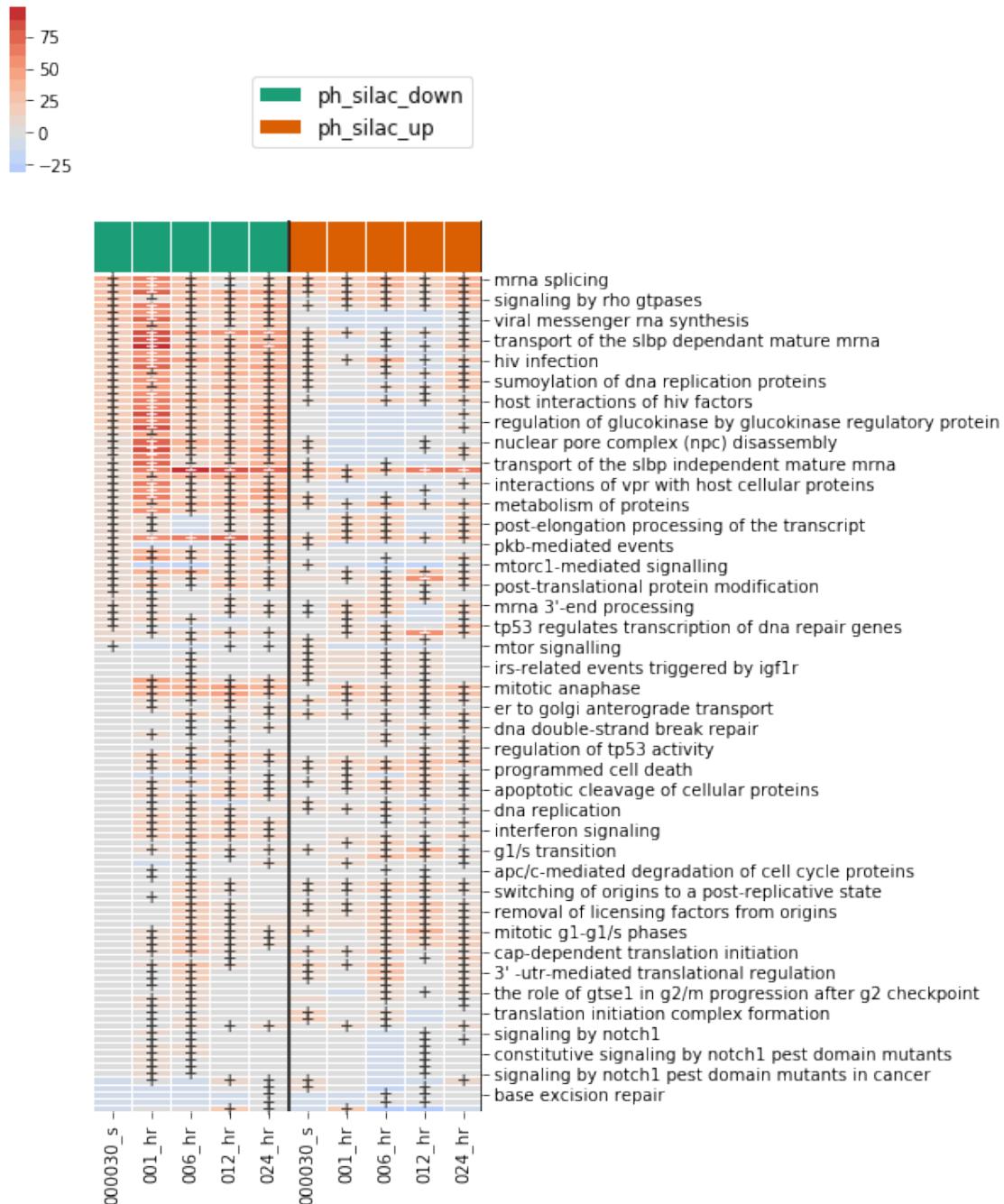
```
[15]: # This figure shows the process of discovering "too general/not useful term".
# Translation involves various genes that make up individual process
ph_silac_copy.show_terms_below("translation").heatmap(
    convert_to_log=False,
    cluster_by_set=False,
    cluster_row=False,
    values='combined_score',
    columns=['category', 'sample_id'],
    annotate_sig=True,
```

```

    div_colors=True,
    linewidths=.005,
    figsize=(5, 12)
);

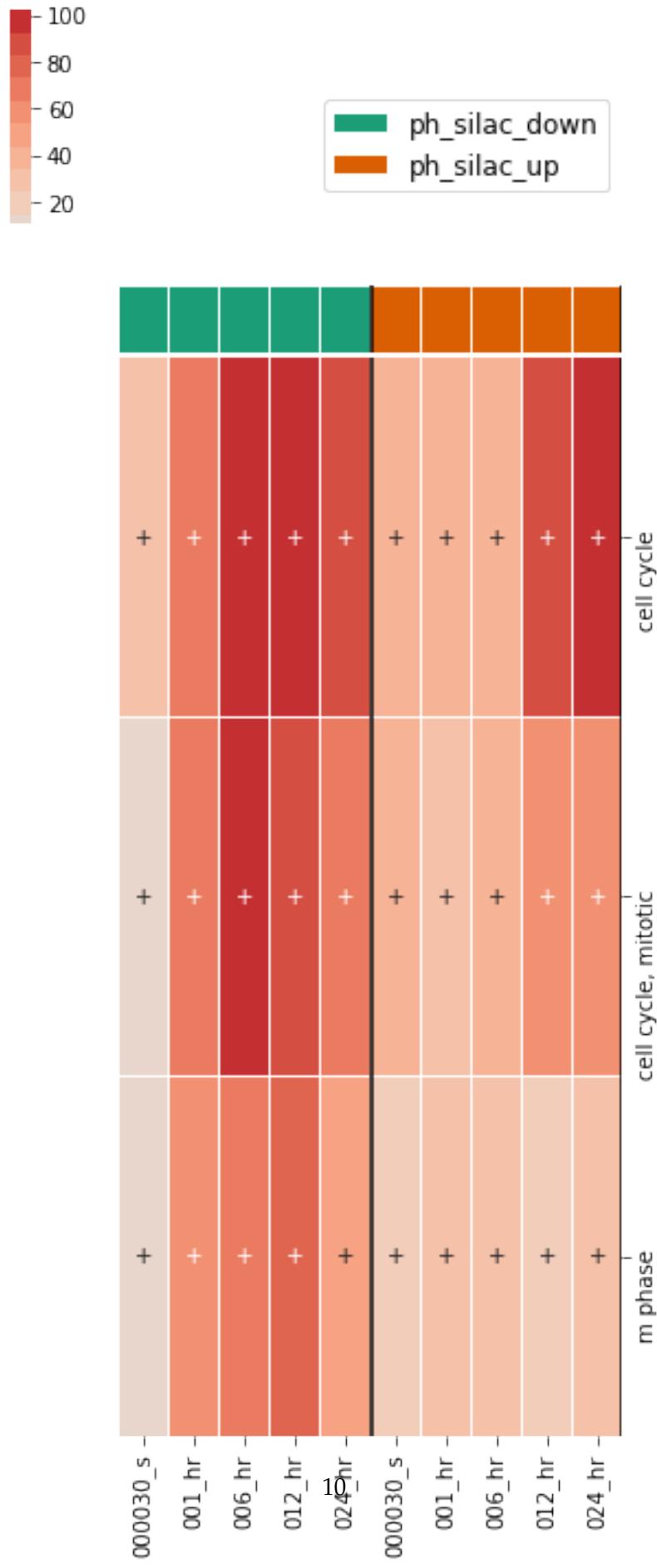
```

Number of rows went from 159 to 36



```
[16]: # We can use the same thought process to find more specific terms
ph_silac_copy.show_terms_below("cell cycle", remove_subset=False, threshold=.5).
    →require_n_sig(n_sig=1).heatmap(
        convert_to_log=False,
        cluster_by_set=False,
        cluster_row=False,
        values='combined_score',
        columns=['category', 'sample_id'],
        annotate_sig=True,
        div_colors=True,
        linewidths=.005,
        figsize=(5, 12)
);
plt.savefig("below_cell_cycle.png", bbox_inches='tight', dpi=300)
```

Number of rows went from 159 to 29



3.1.2 Creating an annotated set network.

```
[17]: term_net, mol_net = create_subnetwork(  
    ph_silac,  
    network=network,  
    save_name='ph_silac_ags',  
    use_cytoscape=False,  
    use_fdr=True, use_threshold=True, min_edges=20  
)
```

Creating ontology network

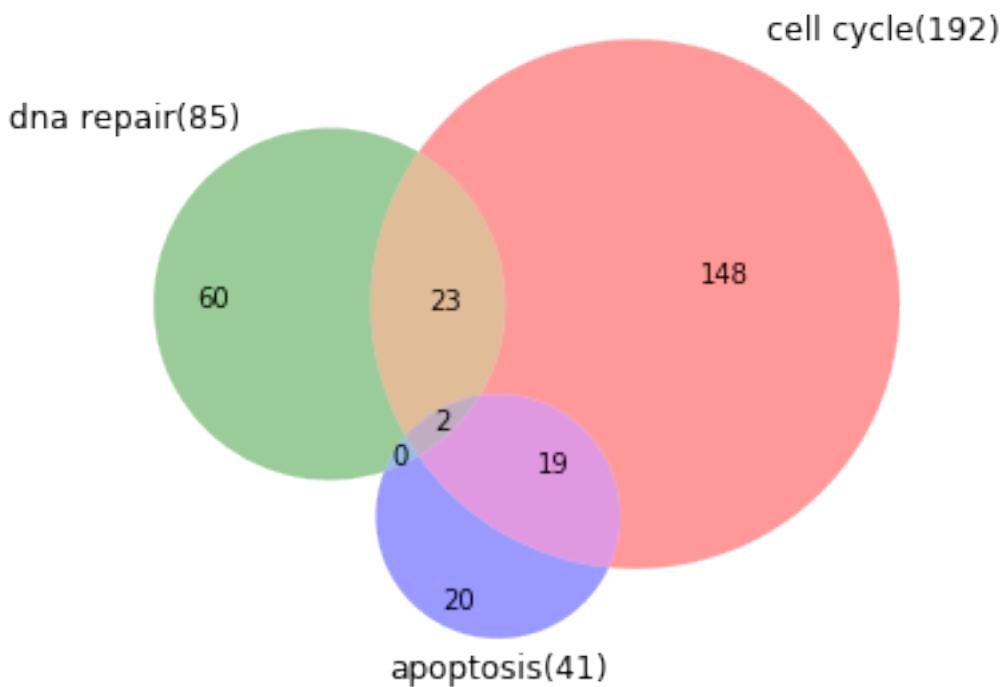
```
[18]: vis.draw_cyjs(term_net, default_color='white', layout='concentric',  
    spacingFactor=2.8)
```

<IPython.core.display.HTML object>

3.1.3 Known mechanisms of bendamustine

Next, we will demonstrate how to explore known mechanisms. Bendamustine is known to cause dna damage, cell cycle arrest, and apoptosis, so we will start there.

```
[19]: selected_terms = ['dna repair', 'cell cycle', 'apoptosis']  
  
# extract out lists of genes for each term  
g_sets = [ph_silac.sig.term_to_genes(i) for i in selected_terms]  
  
# visualize the overlap  
create_venn3(*g_sets+selected_terms, save_name='venn_canonical');
```



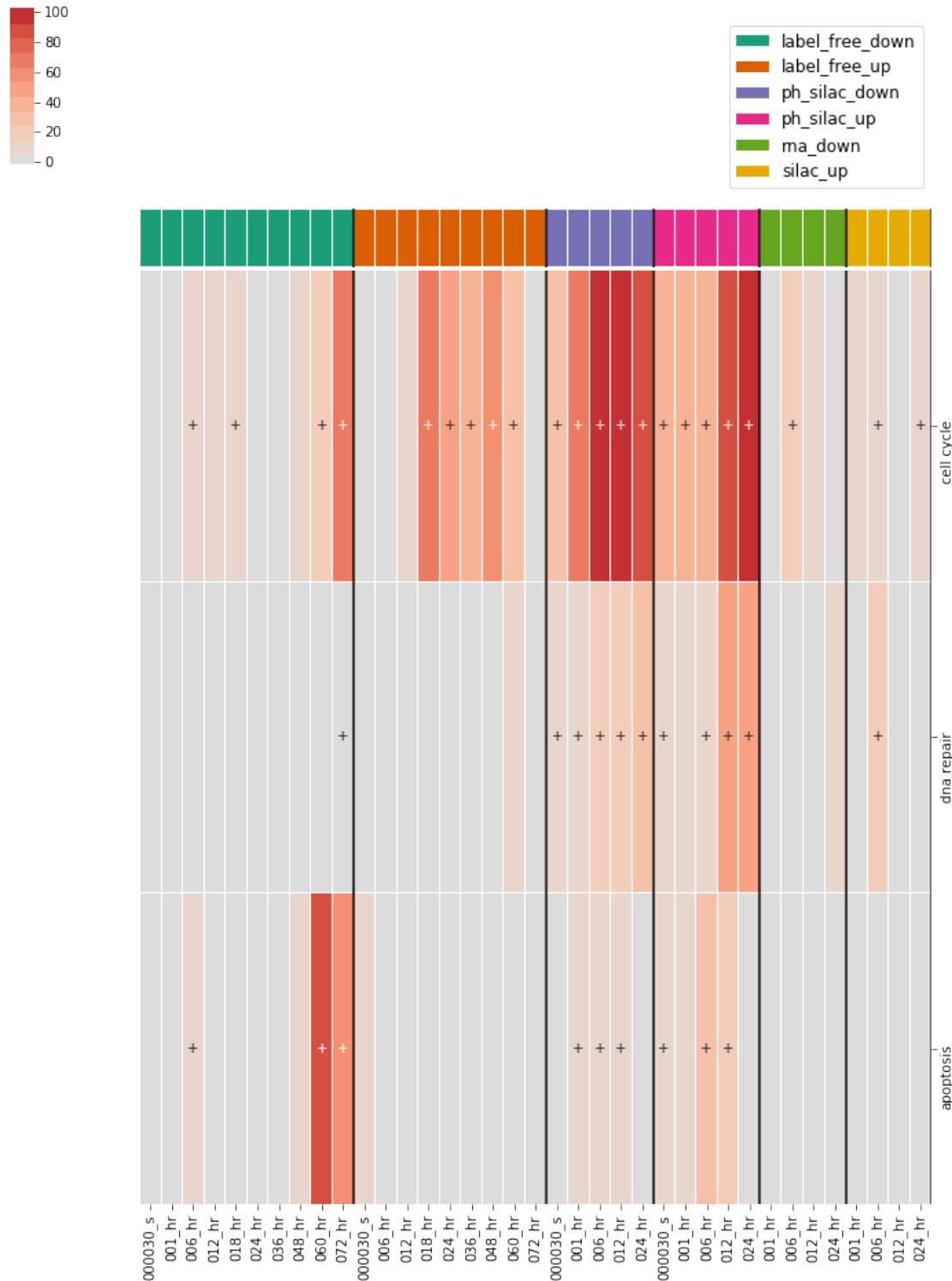
```
[20]: def print_numbers(term_name):
    genes = reactome_only.sig.term_to_genes(term_name)
    n_sig = len(genes)
    if n_sig:
        n_sig_ptms = len(exp_data.subset(genes).sig.label_list)
        print("{} : {} : {}".format(term_name, n_sig, n_sig_ptms))
print_numbers('cell cycle')
print_numbers('dna repair')
print_numbers('apoptosis')
```

cell cycle : 247 : 600
 dna repair : 110 : 232
 apoptosis : 72 : 138

```
[21]: # though we focused on ph-silac originally, we can see if the terms of interest
      →are in other datasets
subset = reactome_only.loc[reactome_only.term_name.isin(selected_terms)].copy()

subset.filter_multi(
    category=['ph_silac_down', 'ph_silac_up',
              'label_free_down', 'label_free_up',
              'silac_up', 'silac_down',
              'rna_up', 'rna_down'],
    inplace=True,
```

```
)  
  
# remove time points and samples that don't contain a significant term  
subset.require_n_sig(  
    n_sig=1,  
    index=['category', 'sample_id'],  
    columns='term_name',  
    inplace=True  
)  
subset.require_n_sig(  
    n_sig=1,  
    index=['sample_id', 'category'],  
    columns='term_name',  
    inplace=True  
)  
  
fig = subset.heatmap(  
    convert_to_log=False,  
    cluster_by_set=False,  
    annotate_sig=True,  
    columns=['category', 'sample_id'],  
    div_colors=True,  
    linewidths=.005,  
    figsize=(12, 16)  
);
```



```
[22]: # subset the data to only include these terms
subset = reactome_only.sig.loc[reactome_only.sig.term_name.
    →isin(selected_terms)].copy()
```

```

term_net, mol_net = create_subnetwork(
    subset,
    network=network,
    save_name='apop_dna_cell_cycle',
    use_cytoscape=False,
    use_fdr=True,
    use_threshold=True,
    min_edges=10
)
print(len(mol_net.nodes))
print(len(mol_net.edges))

```

Creating ontology network
311
2406

[23]:

```

nx.set_node_attributes(term_net, 'white', 'color')
vis.draw_cyjs(term_net, layout='concentric', spacingFactor=2.8)

```

<IPython.core.display.HTML object>

DNA damage response

[24]:

```

# extract dna repair genes
dna_repair_genes = reactome_only.sig.term_to_genes('dna repair')

exp_data.ph_silac.heatmap(
    dna_repair_genes,
    convert_to_log=True,
    subset_index='identifier',
    index='label',
    cluster_row=True,
    rank_index=True,
    min_sig=2,
    num_colors=13,
    linewidths=0.01
);

exp_data.label_free.heatmap(
    dna_repair_genes,
    convert_to_log=True,
    subset_index='identifier',
    index='label',

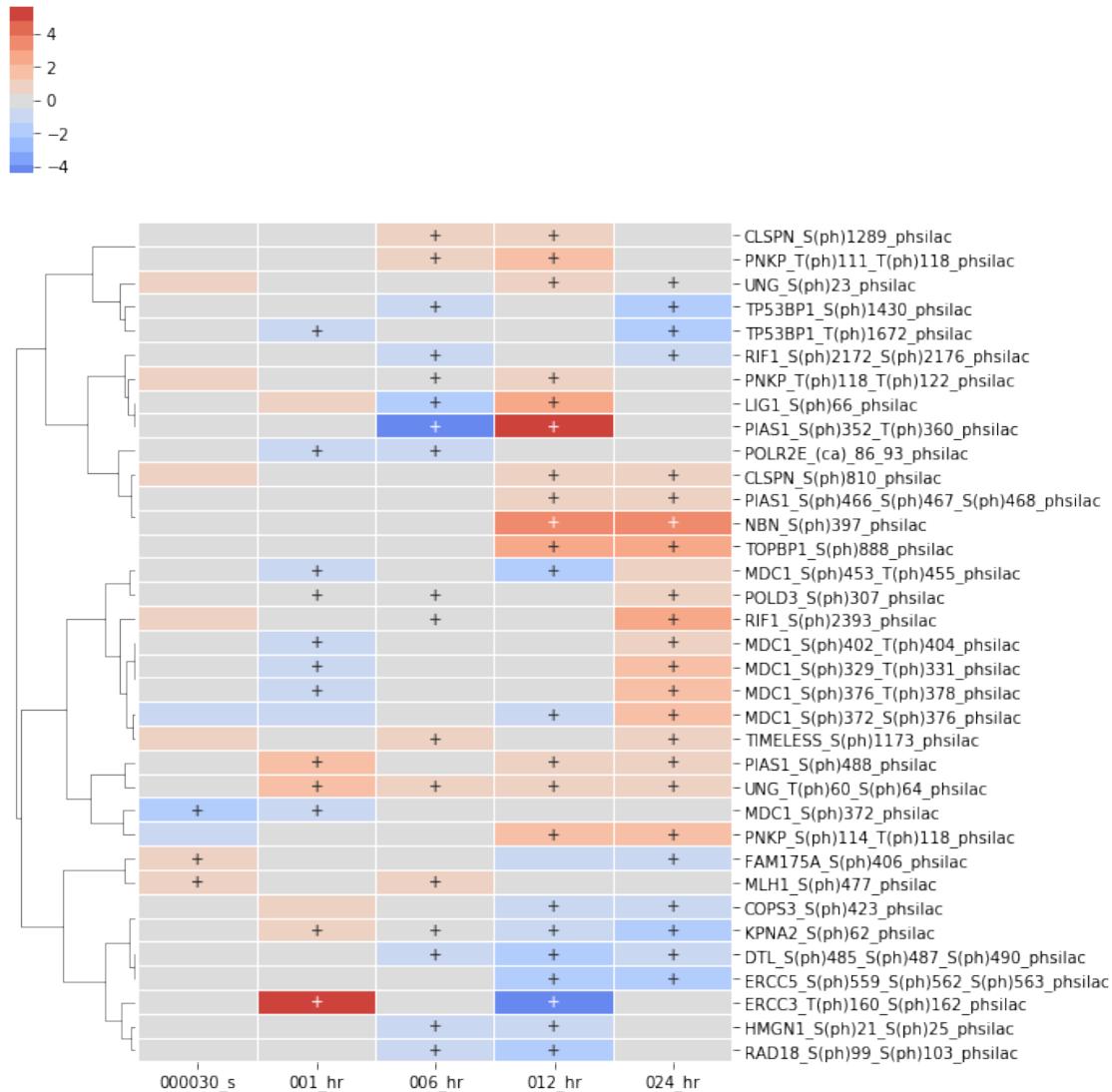
```

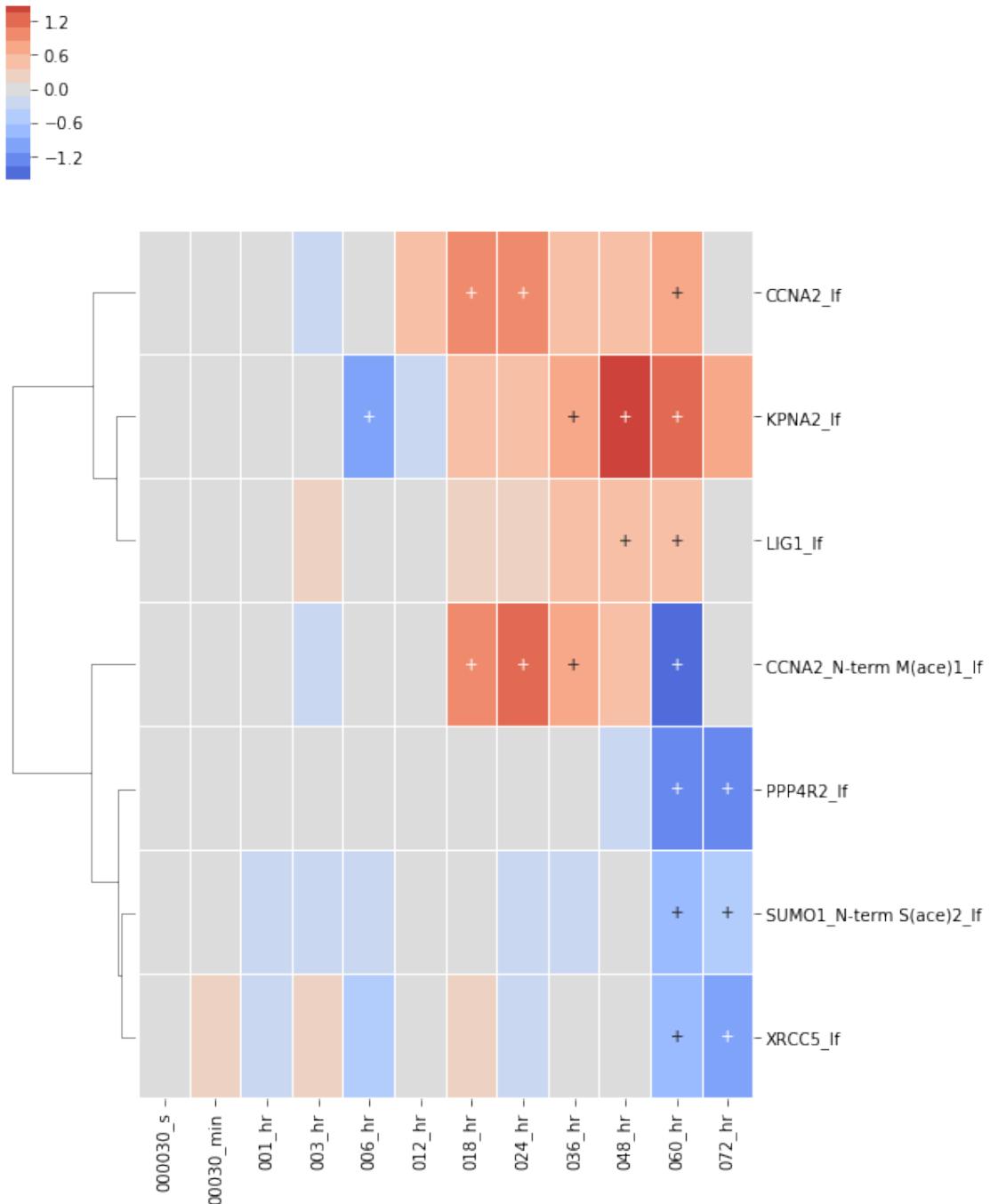
```
    cluster_row=True,
    rank_index=True,
    min_sig=2,
    num_colors=13,
    linewidths=0.01
);

exp_data.rna_seq.heatmap(
    dna_repair_genes,
    convert_to_log=True,
    subset_index='identifier',
    index='label',
    cluster_row=True,
    rank_index=True,
    min_sig=2,
    num_colors=13,
    linewidths=0.01
);
```

E:\PycharmProjects\PycharmProjects\Magine\magine\plotting\heatmaps.py:64:
UserWarning:

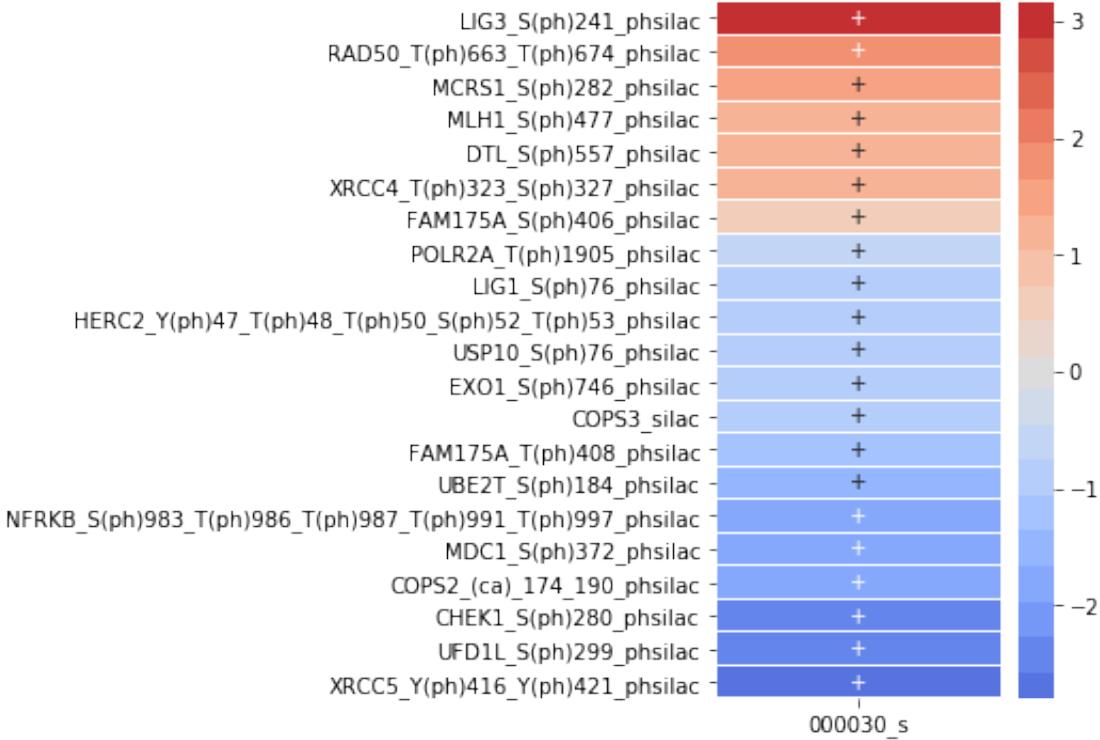
Empty array after filtering.





```
[25]: # look at only first time point
first_tp = exp_data.species.subset(dna_repair_genes, sample_ids=['000030_s']).
→sig

first_tp.heatmap(figsize=(3,6), index='label', linewidths=0.01);
```



```
[26]: # Create a network based on the first time point, include dna damage nodes
dna_repair_subnet = net_sub.expand_neighbors(
    nodes=first_tp.id_list,
    upstream=False,
    downstream=True,
    add_interconnecting_edges=True,
    max_dist=1,
    include_only=dna_repair_genes
)
# removes disconnected nodes in network
dna_repair_subnet = utils.delete_disconnected_network(dna_repair_subnet)

colors = dict()
for i in dna_repair_subnet.nodes:
    if i in first_tp.id_list:
        colors[i] = 'lightgreen'
    else:
        colors[i] = 'lightblue'
nx.set_node_attributes(dna_repair_subnet, colors, 'color')
```

Node not in graph

```
[27]: vis.draw_cyjs(dna_repair_subnet, layout='cose-bilkent', spacingFactor=2.)
```

```
<IPython.core.display.HTML object>
```

```
[28]: vis.draw_cyjs(dna_repair_subnet, layout='concentric', spacingFactor=1.)
```

```
<IPython.core.display.HTML object>
```

Cell cycle

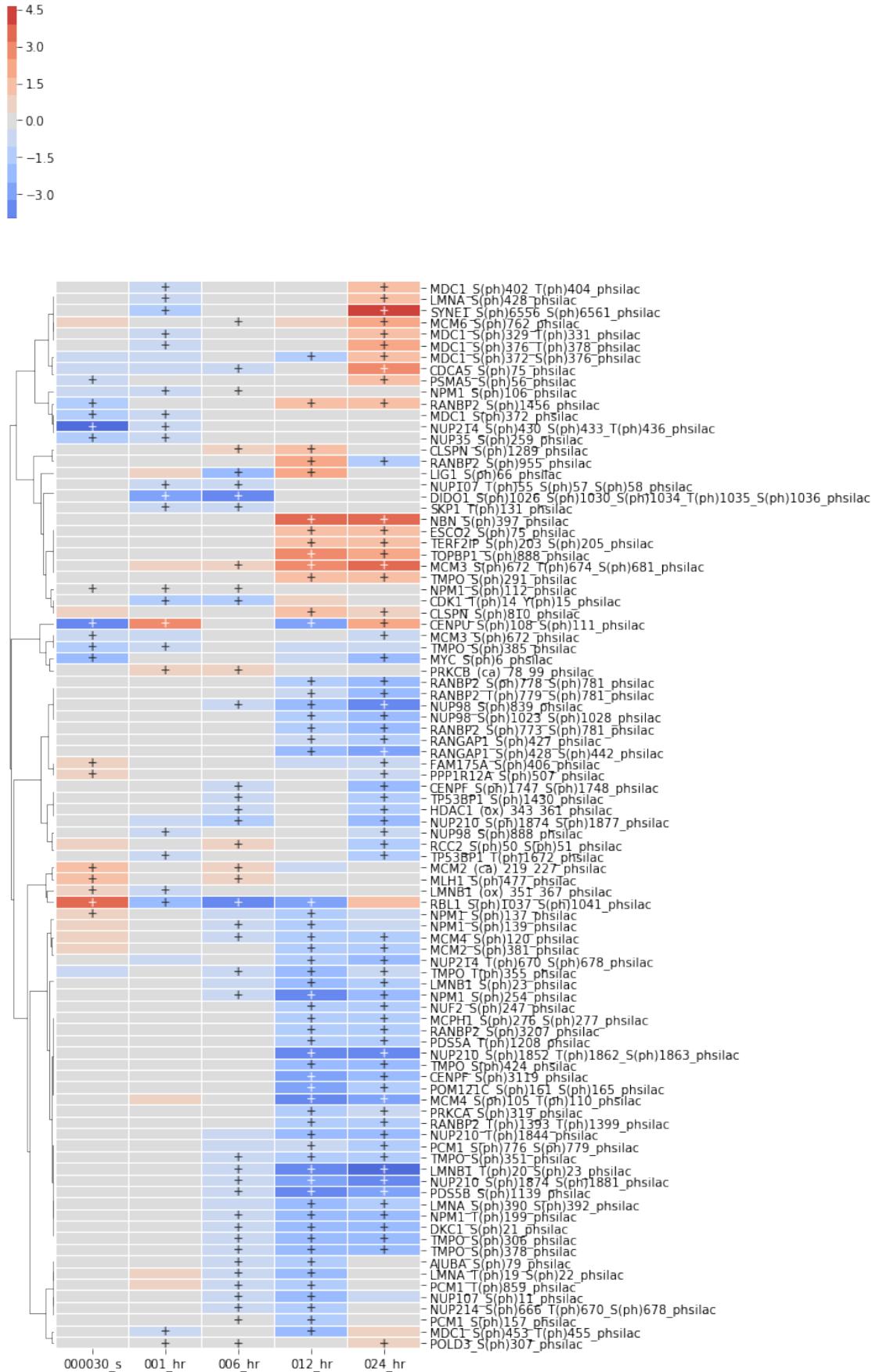
```
[29]: cell_cycle_genes = reactome_only.sig.term_to_genes('cell cycle')
```

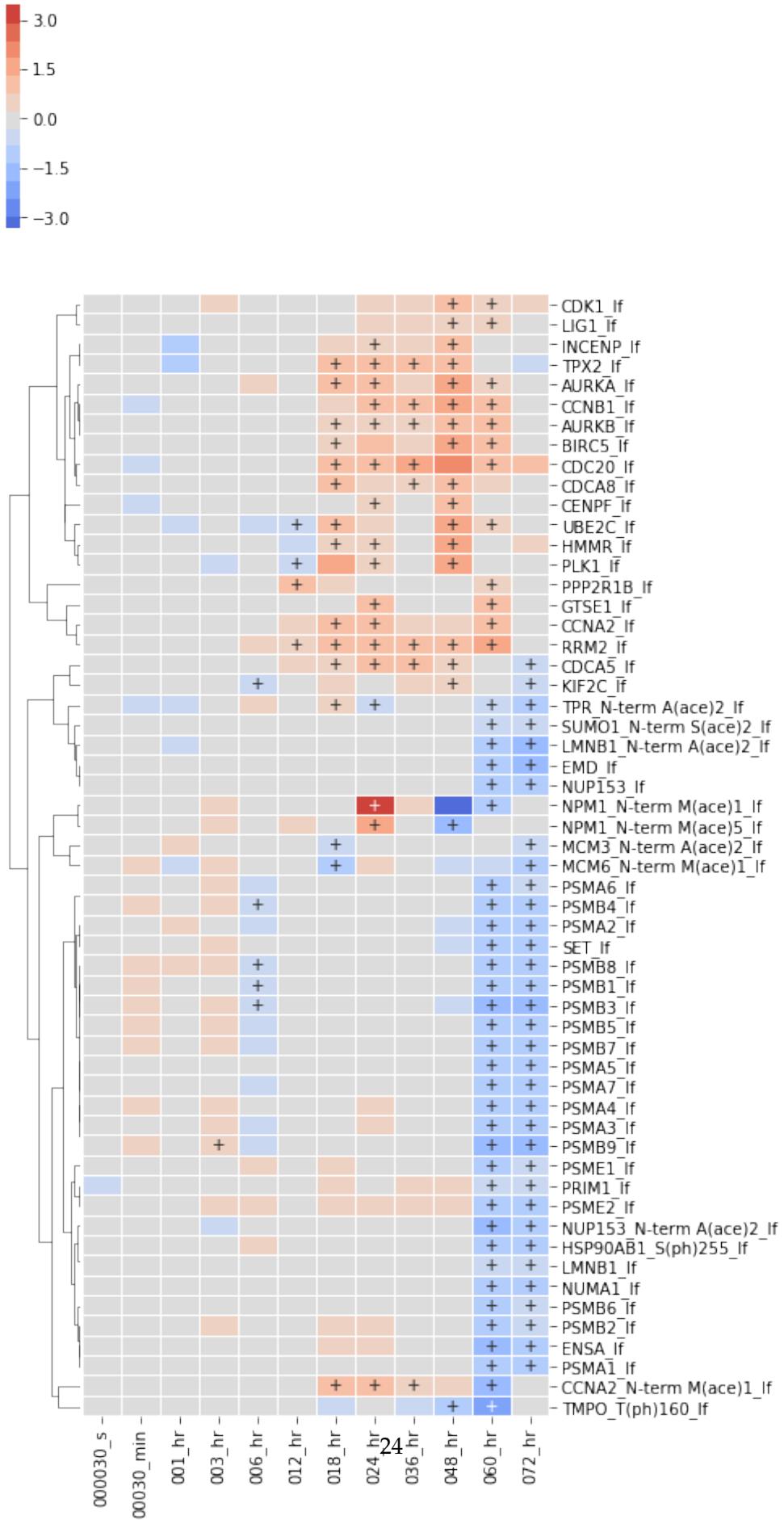
```
exp_data.ph_silac.heatmap(
    cell_cycle_genes,
    subset_index='identifier',
    index='label',
    cluster_row=True,
    rank_index=True,
    min_sig=2,
    num_colors=13,
    linewidths=0.01,
    figsize=(6, 20),
    y_tick_labels=True
);

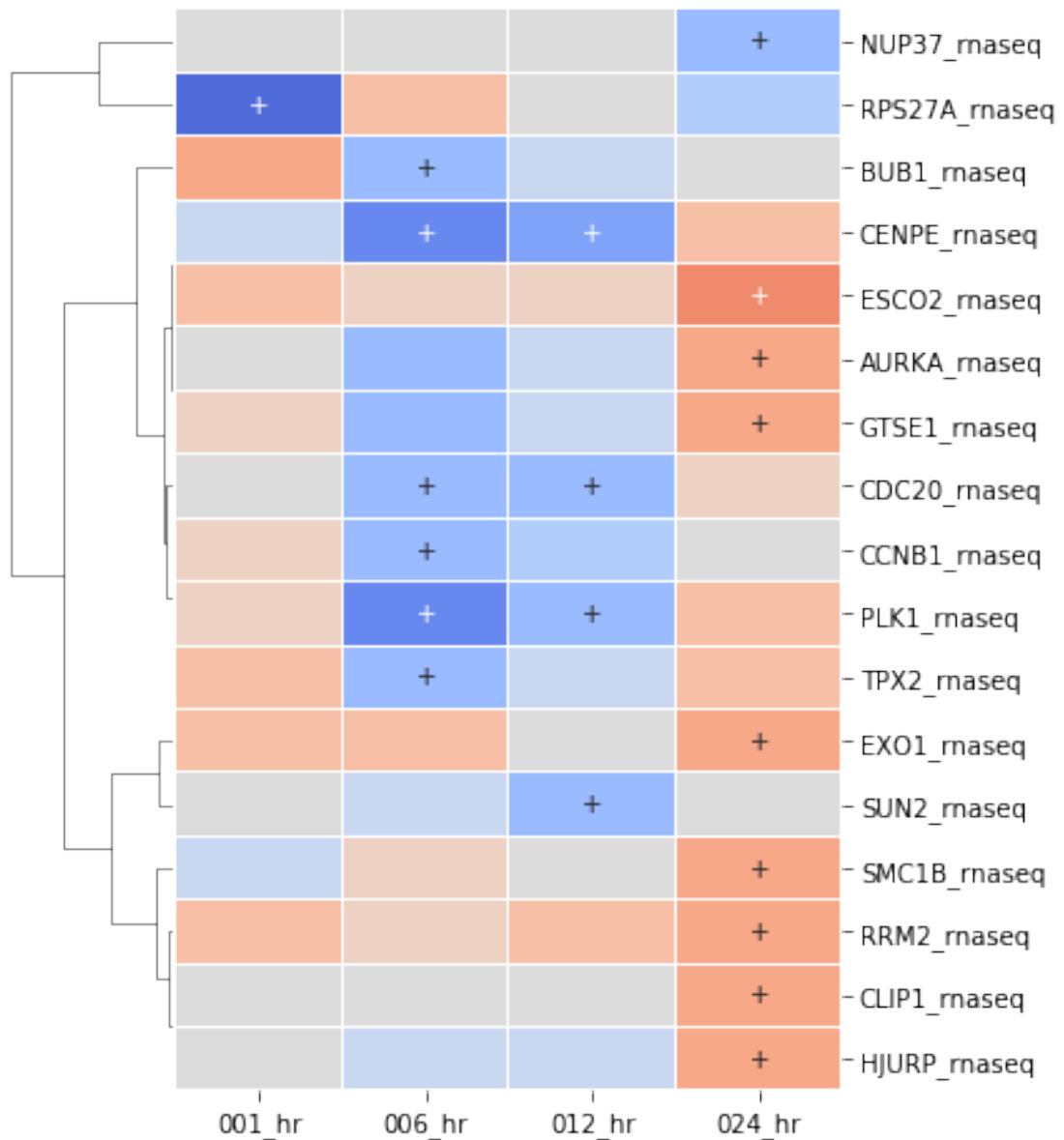
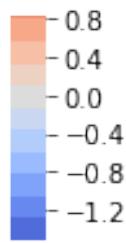
exp_data.label_free.heatmap(
    cell_cycle_genes,
    subset_index='identifier',
    index='label',
    cluster_row=True,
    rank_index=True,
    min_sig=2,
    num_colors=13,
    linewidths=0.01,
    figsize=(6,16)
);

exp_data.rna_seq.heatmap(
    cell_cycle_genes,
    subset_index='identifier',
    index='label',
    cluster_row=True,
    rank_index=True,
    min_sig=1,
    num_colors=13,
    linewidths=0.01,
    figsize=(6,10)
```

) ;



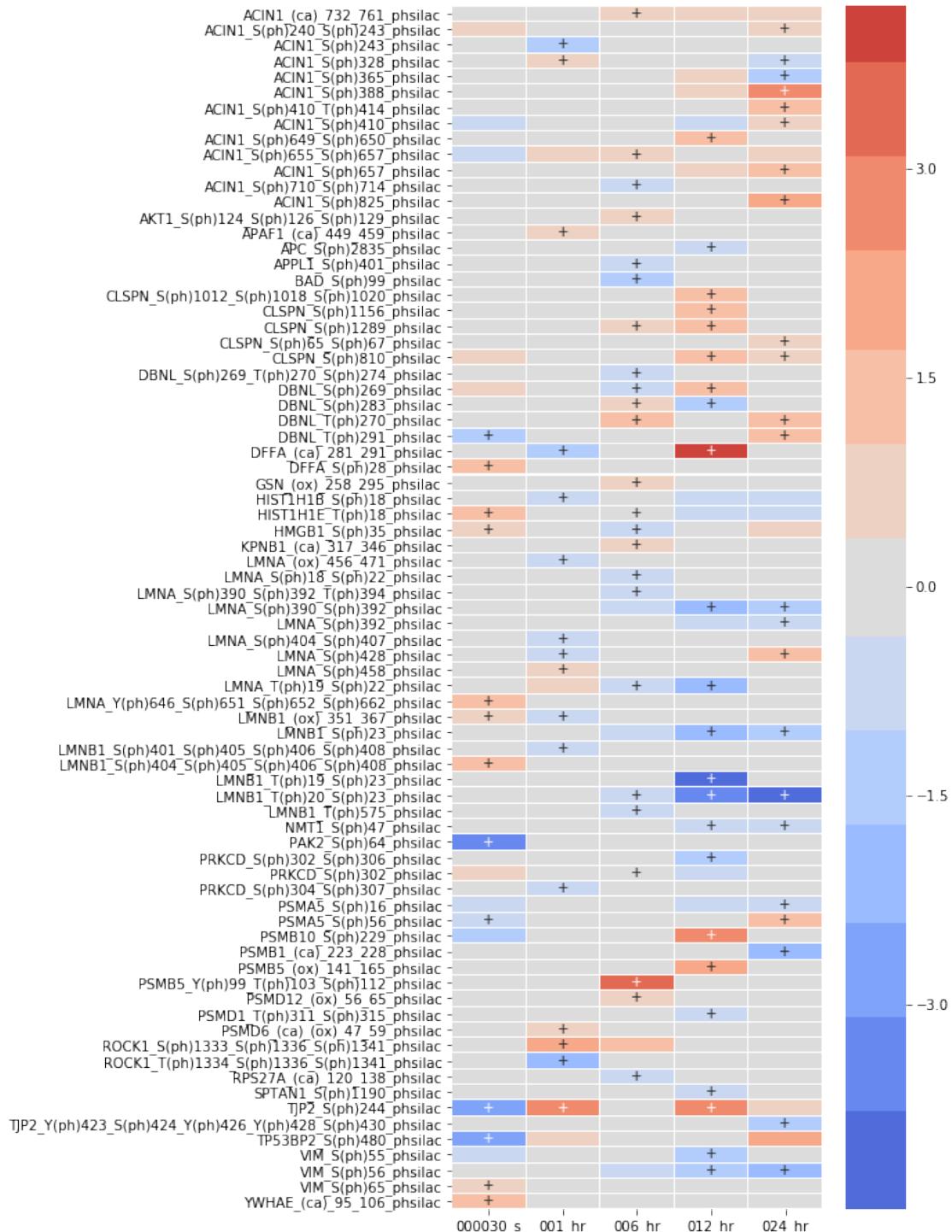


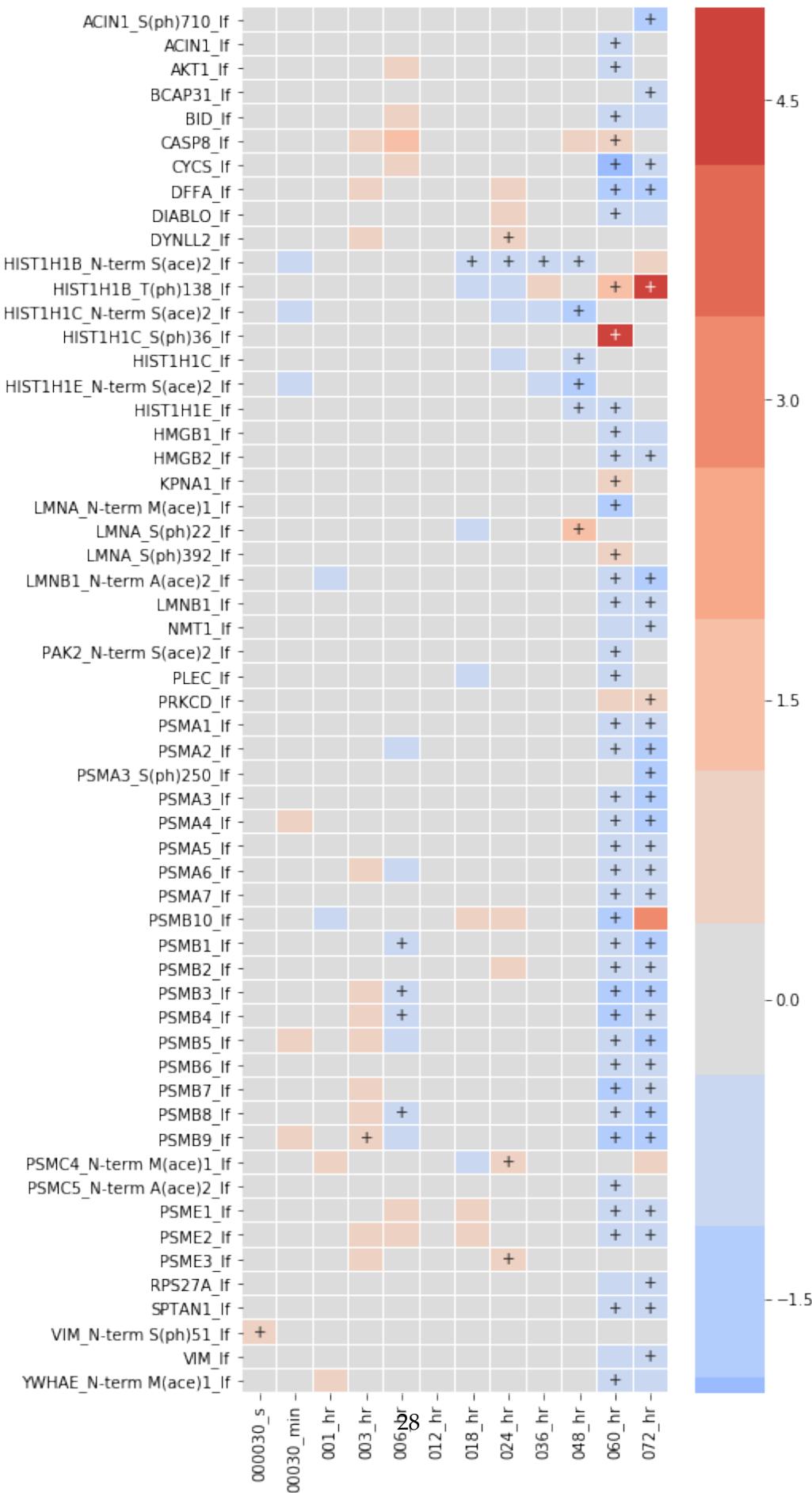


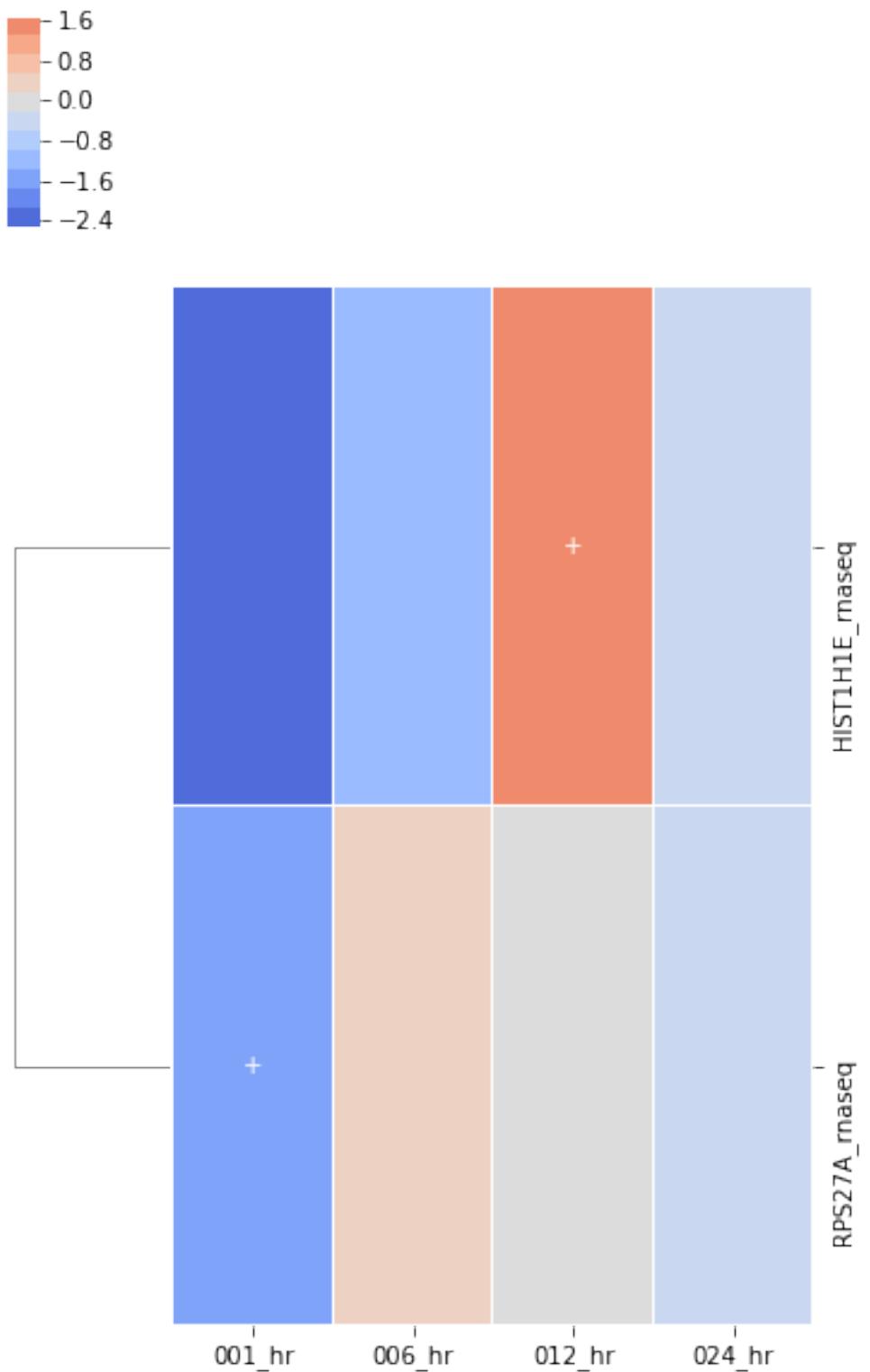
```
[30]: apoptosis_genes = reactome_only.sig.term_to_genes('apoptosis')

exp_data.ph_silac.heatmap(
    apoptosis_genes,
    subset_index='identifier',
    index='label',
    cluster_row=False,
    rank_index=True,
    min_sig=1,
    num_colors=13,
    linewidths=0.01,
    figsize=(6,16)
);

exp_data.label_free.heatmap(
    apoptosis_genes,
    subset_index='identifier',
    index='label',
    cluster_row=False,
    rank_index=True,
    min_sig=1,
    num_colors=13,
    linewidths=0.01,
    figsize=(6,16)
);
exp_data.rna_seq.heatmap(
    apoptosis_genes,
    subset_index='identifier',
    index='label',
    cluster_row=True,
    rank_index=True,
    min_sig=1,
    num_colors=13,
    linewidths=0.01,
    figsize=(6,10)
);
```







```
[31]: print("DNA repair")
for i in exp_data.exp_methods:
    print("\t", i, len(exp_data[i].subset(dna_repair_genes).sig.id_list),
          len(exp_data[i].subset(dna_repair_genes).sig.label_list))
print("Cell cycle")
for i in exp_data.exp_methods:
    print("\t", i, len(exp_data[i].subset(cell_cycle_genes).sig.id_list),
          len(exp_data[i].subset(cell_cycle_genes).sig.label_list))
print("Apoptosis")
for i in exp_data.exp_methods:
    print("\t", i, len(exp_data[i].subset(apoptosis_genes).sig.id_list),
          len(exp_data[i].subset(apoptosis_genes).sig.label_list))
```

DNA repair

silac	7	7
ph_silac	80	173
HILIC	0	0
C18	0	0
label_free	43	47
rna_seq	5	5

Cell cycle

silac	10	10
ph_silac	153	444
HILIC	0	0
C18	0	0
label_free	111	129
rna_seq	17	17

Apoptosis

silac	2	2
ph_silac	33	77
HILIC	0	0
C18	0	0
label_free	47	57
rna_seq	2	2

```
[32]: # create plots grouping together data for each platform

genes_in_labels = utils.create_dict_from_node_attributes(mol_net, 'termName')

# phosph-silac
heatmap_by_terms(
    exp_data.ph_silac,
    convert_to_log=True,
    index='label',
    term_labels=list(genes_in_labels.keys()),
    term_sets=list(genes_in_labels.values()),
    div_colors=True,
```

```

linewidths=0.01,
min_sig=3,
annotate_sig=True,
cluster_col=False,
cluster_row=False,
y_tick_labels=True,
figsize=(6, 12)
);

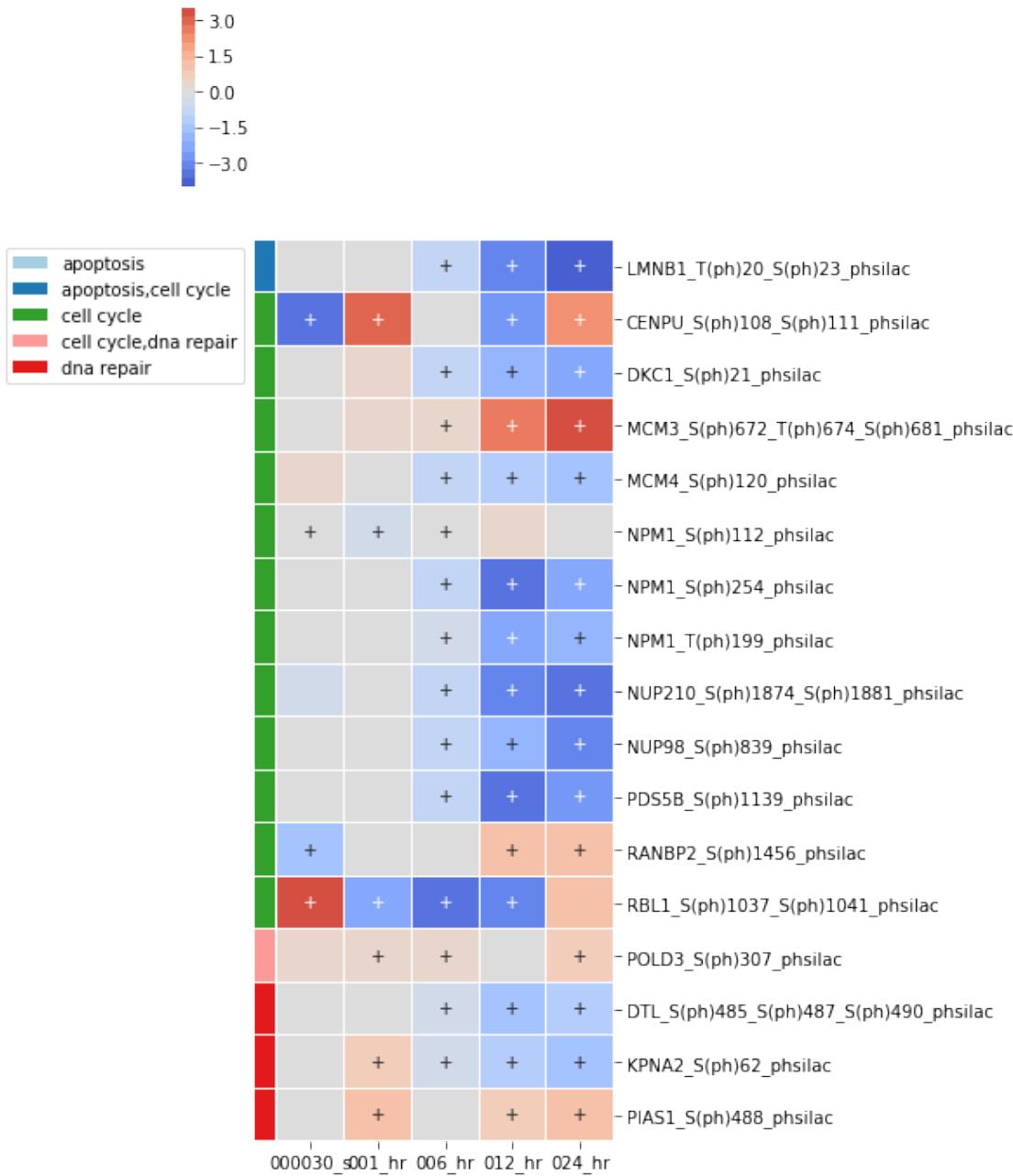
# label free
heatmap_by_terms(
    exp_data.label_free,
    convert_to_log=True,
    index='label',
    term_labels=list(genes_in_labels.keys()),
    term_sets=list(genes_in_labels.values()),
    div_colors=True,
    linewidths=0.01,
    min_sig=3,
    annotate_sig=True,
    cluster_col=False,
    cluster_row=False,
    y_tick_labels=True,
    figsize=(8, 12)
);

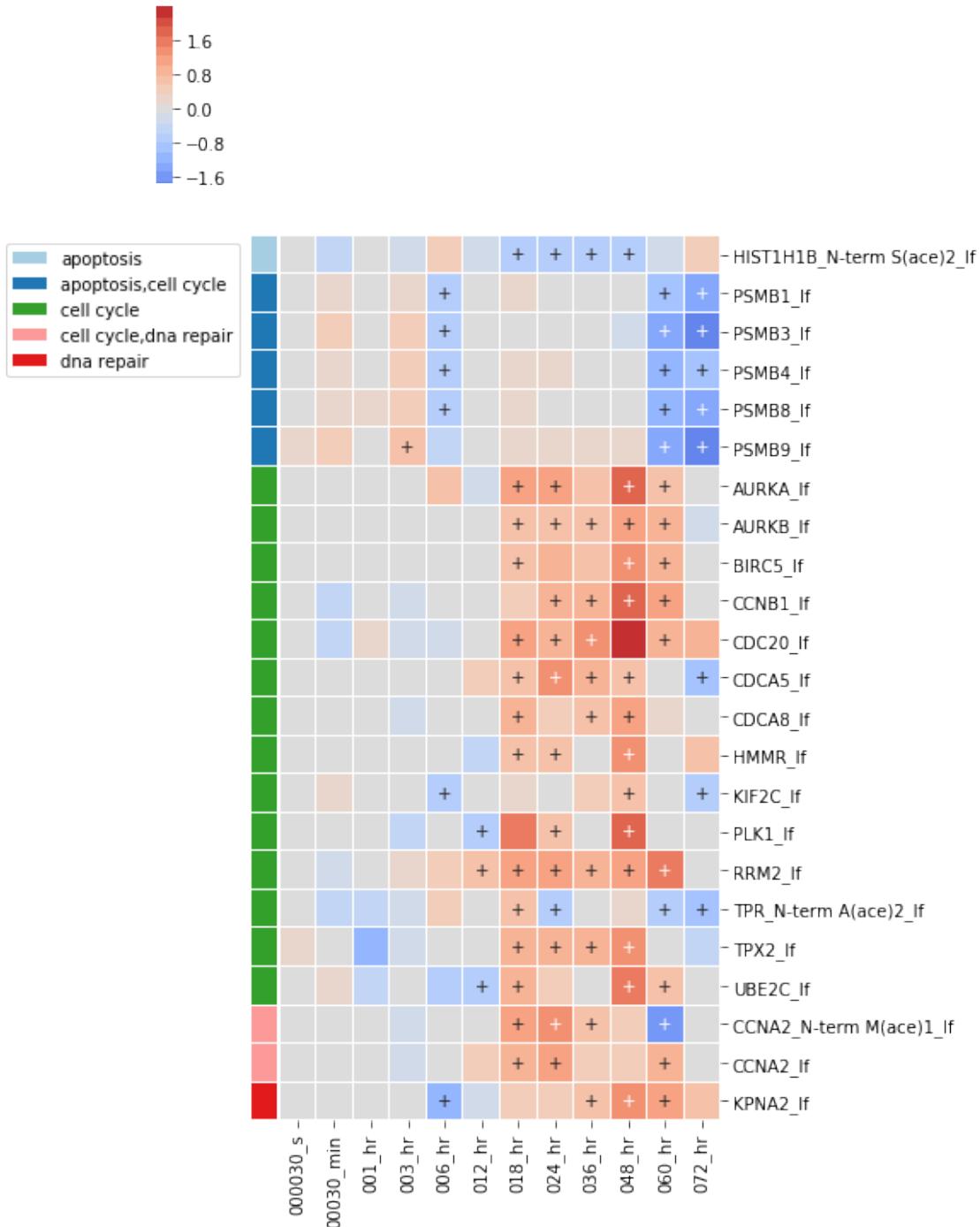
# silac
heatmap_by_terms(
    exp_data.silac,
    convert_to_log=True,
    index='label',
    term_labels=list(genes_in_labels.keys()),
    term_sets=list(genes_in_labels.values()),
    div_colors=True,
    linewidths=0.01,
    min_sig=2,
    annotate_sig=True,
    cluster_col=False,
    cluster_row=False,
    y_tick_labels=True,
    figsize=(4, 4)
);

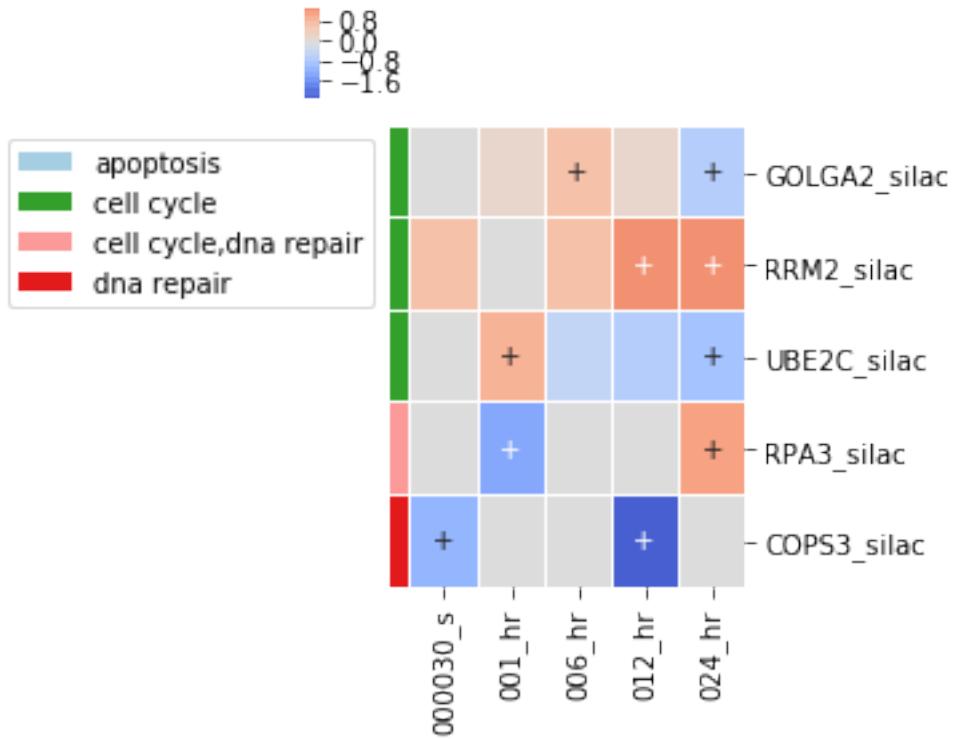
# rna
heatmap_by_terms(

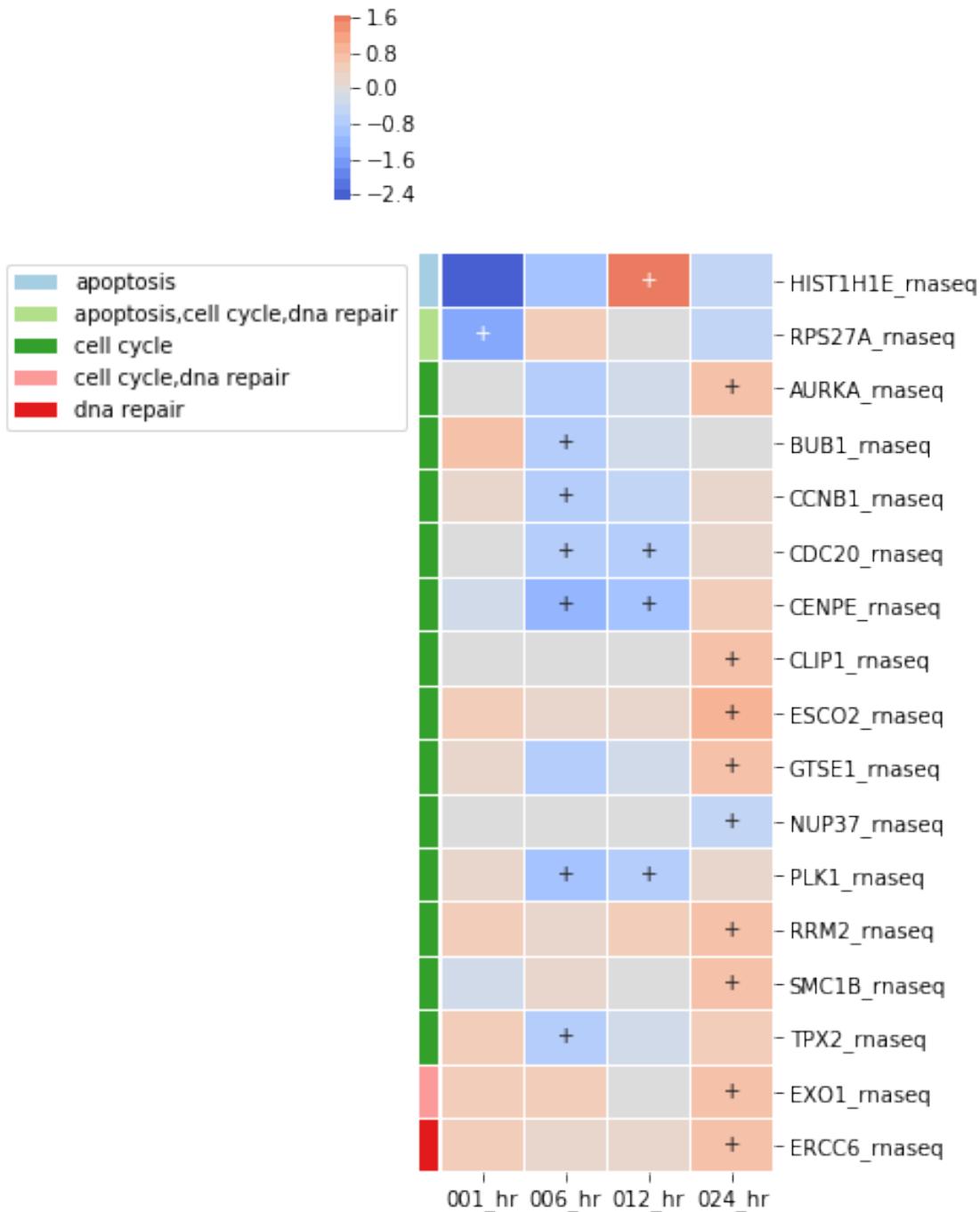
```

```
exp_data.rna_seq,  
convert_to_log=True,  
index='label',  
term_labels=list(genes_in_labels.keys()),  
term_sets=list(genes_in_labels.values()),  
div_colors=True,  
linewidths=0.01,  
min_sig=1,  
annotate_sig=True,  
cluster_col=False,  
cluster_row=False,  
y_tick_labels=True,  
figsize=(5, 10)  
) ;
```









3.1.4 Lets look at only up-regulated ph-silac

```
[33]: ph_silac_up = reactome_only.filter_multi(category=['ph_silac_up'])
```

```
not_useful = [
```

```

'gene expression', 'translation',
'immune system',
'disease', 'diseases of signal transduction',
'infectious disease',
'influenza infection', 'influenza life cycle',
'influenza viral rna transcription and replication',
]

ph_silac_up = ph_silac_up.loc[~ph_silac_up['term_name'].isin(not_useful)]

print("Number of terms before filtering base on minimum time points : {}\"\
      ".format(len(ph_silac_up.term_name.unique())))

ph_silac_up.require_n_sig(
    index='term_name',
    columns='sample_id',
    n_sig=3,
    inplace=True
)
ph_silac_up_copy = ph_silac_up.copy()
print("Number of terms after filtering base on minimum time points : {}\"\
      ".format(len(ph_silac_up.term_name.unique())))

fig = ph_silac_up_copy.heatmap(figsize=(4, 24));
fig.savefig('ph_silac.png', dpi=300, bbox_inches='tight')

ph_silac_up.remove_redundant(
    threshold=.7,
    level='sample',
    inplace=True,
    sort_by='combined_score'
)

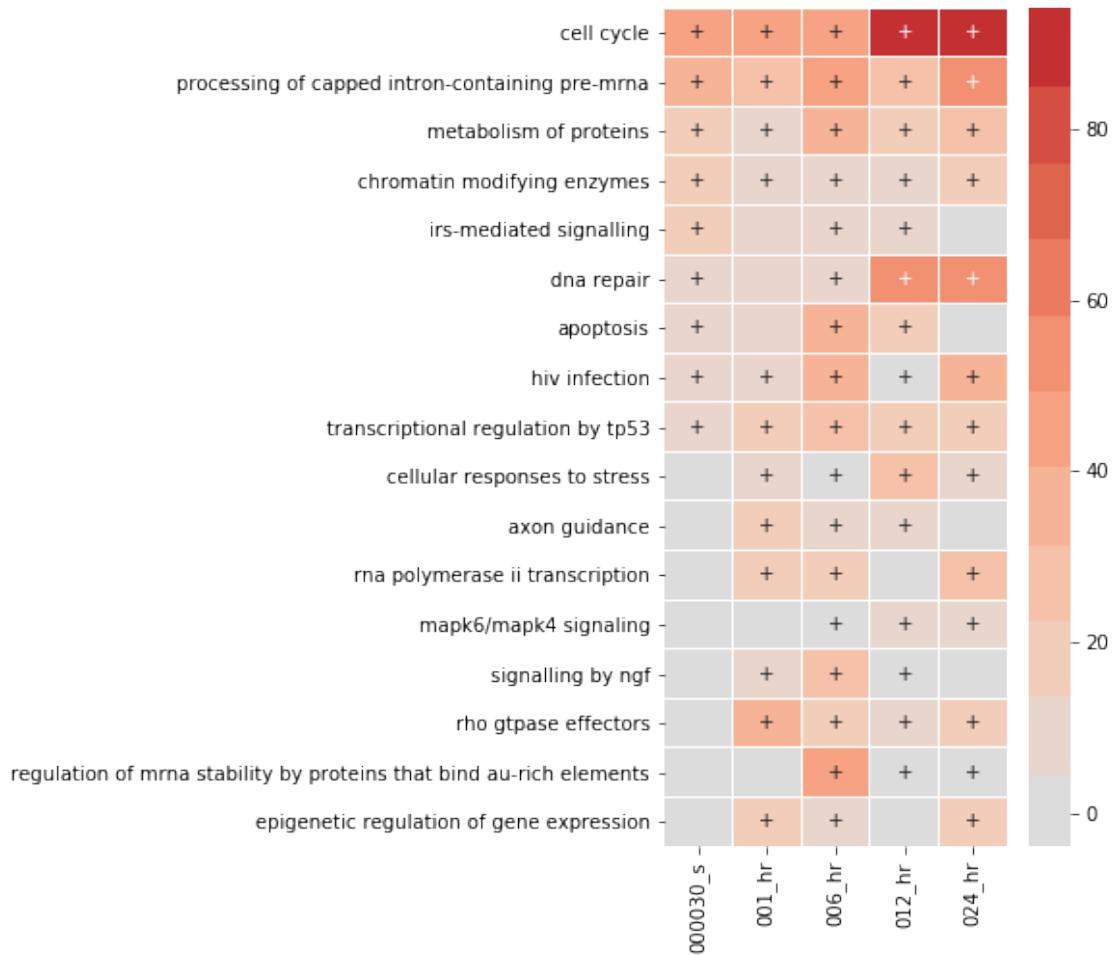
ph_silac_up.remove_redundant(
    threshold=.7,
    level='dataframe',
    inplace=True,
    sort_by='combined_score'
)
print("Number of genes before : {}".format(len(ph_silac_up_copy.
    →all_genes_from_df())))
print("Number of genes after : {}".format(len(ph_silac_up.all_genes_from_df())))
fig = ph_silac_up.heatmap(
    convert_to_log=False,
    cluster_by_set=False,
    cluster_row=False,
    values='combined_score',

```

```
annotate_sig=True,  
div_colors=True,  
linewidths=.005,  
figsize=(4,8)  
);  
fig.savefig('ph_silac_slimmed.png', dpi=300, bbox_inches='tight')
```

Number of terms before filtering base on minimum time points : 560
Number of terms after filtering base on minimum time points : 83
Number of rows went from 83 to 30
Number of rows went from 30 to 17
Number of genes before : 570
Number of genes after : 550





```
[34]: # note that we can still get back terms that were compressed
ph_silac_up_copy.show_terms_below(
    'cell cycle',
    threshold=.7,
    remove_subset=True
).heatmap(
    figsize=(3, 12),
    cluster_by_set=False,
    linewidths=0.01,
);
```

Number of rows went from 83 to 17



```
[35]: term_net, mol_net = create_subnetwork(
    ph_silac_up,
    network=network,
    save_name='ph_silac ASN',
    use_cytoscape=False,
    use_fdr=True, use_threshold=True, min_edges=25
)
for i in ph_silac_up.term_name.unique():
    if i not in term_net.nodes:
        print(i)
```

Creating ontology network

```
[36]: nx.set_node_attributes(term_net, 'white', 'color')
vis.draw_cyjs(term_net, layout='cose-bilkent', spacingFactor=1.4)
```

<IPython.core.display.HTML object>

3.1.5 Exploring terms outside canonical

Since enrichment analysis is performed over all terms in a gene set, some of the terms might not make sense in certain context. This could be due to genes being classified in multiple sets, bad assignment, or possible new biology. The time it takes to explore each term in understanding why it is enriched can be time consuming, which is perhaps why people disregard terms frequently. We decided to use MAGINE to explore one of these terms. HIV infection

```
[37]: reactome_only.sig.show_terms_below(
    'hiv infection',
    level='dataframe',
    threshold=.25,
    remove_subset=False
).heatmap(
    figsize=(6,16),
    linewidths=0.05,# cluster_by_set=True,
);
```

Number of rows went from 581 to 73



```
[38]: reactome_only.sig.find_similar_terms('hiv infection', level='dataframe',
                                         remove_subset=False )
```

	term_name	similarity_score
37	host interactions of hiv factors	0.70
21	infectious disease	0.68
17	hiv life cycle	0.57
46	late phase of hiv life cycle	0.49
27	disease	0.46
282	pcp/ce pathway	0.30
112	cytokine signaling in immune system	0.29
61	m phase	0.28

167	scf-beta-trcp mediated degradation of emi1	0.28
234	gli3 is processed to gli3r by the proteasome	0.27
233	degradation of gli2 by the proteasome	0.27
214	activation of nf-kappab in b cells	0.27
121	nik-->noncanonical nf-kb signaling	0.27
122	dectin-1 mediated noncanonical nf-kb signaling	0.27
134	tnfr2 non-canonical nf-kb pathway	0.27
223	cyclin e associated events during g1/s transition	0.27
254	cdk-mediated phosphorylation and removal of cdc6	0.26
240	scf(skp2)-mediated degradation of p27/p21	0.26
207	cyclin a:cdk2-associated events at s phase entry	0.26
274	auf1 (hnrrnp d0) binds and destabilizes mrna	0.26
215	vpu mediated degradation of cd4	0.26
194	assembly of the pre-replicative complex	0.26
270	ubiquitin mediated degradation of phosphorylat...	0.26
268	p53-independent g1/s dna damage checkpoint	0.26
265	p53-independent dna damage response	0.26
224	cdc20:phospho-apc/c mediated degradation of cy...	0.26
227	apc:cdc20 mediated degradation of cell cycle p...	0.26
129	regulation of mrna stability by proteins that ...	0.26
228	apc/c:cdc20 mediated degradation of mitotic pr...	0.26
192	switching of origins to a post-replicative state	0.26
..
389	mitochondrial translation initiation	0.00
418	clearance of nuclear envelope membranes from c...	0.00
390	mitochondrial translation termination	0.00
153	copi-mediated anterograde transport	0.00
155	apoptotic cleavage of cellular proteins	0.00
394	trna aminoacylation	0.00
395	cytosolic trna aminoacylation	0.00
396	rna polymerase iii transcription initiation fr...	0.00
397	rna polymerase iii transcription initiation fr...	0.00
398	deadenylation-dependent mrna decay	0.00
399	map2k and mapk activation	0.00
400	rna polymerase iii transcription initiation	0.00
401	transcriptional activation of mitochondrial bi...	0.00
402	rna polymerase iii abortive and retractive ini...	0.00
403	rna polymerase iii transcription	0.00
404	hdr through mmej (alt-nhej)	0.00
405	semet incorporation into proteins	0.00
406	tp53 regulates transcription of genes involved...	0.00
407	hdl-mediated lipid transport	0.00
408	growth hormone receptor signaling	0.00
409	pyrimidine biosynthesis	0.00
410	pyrimidine metabolism	0.00
411	initiation of nuclear envelope reformation	0.00
412	nuclear envelope reassembly	0.00

```

413 purine ribonucleoside monophosphate biosynthesis          0.00
414 fatty acid, triacylglycerol, and ketone body m...          0.00
415                               signaling by robo receptor      0.00
416                               rho gtpases activate rocks    0.00
417                               purine metabolism            0.00
579 synthesis of pyrophosphates in the cytosol           0.00

```

[580 rows x 2 columns]

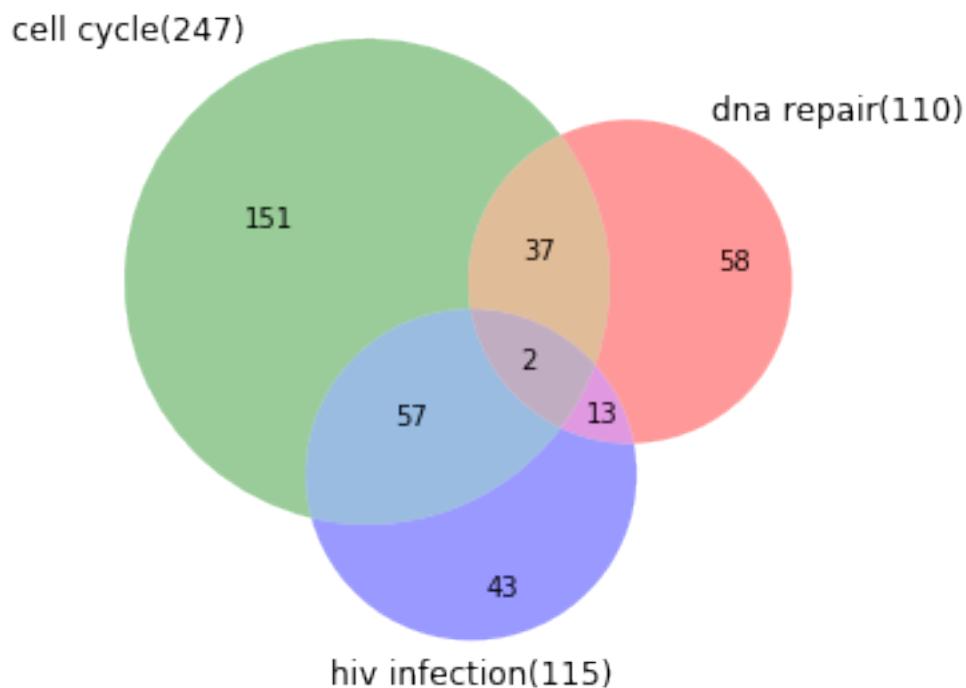
Since HIV infection hijacks dna repair and regulates cell cycle, lets check to see if that explains why HIV infection is enriched.

```

[39]: hits = [
    'cell cycle',
    'dna repair',
    'hiv infection'
]
subset = reactome_only.sig.loc[reactome_only.sig.term_name.isin(hits)].copy()

g_sets = [reactome_only.sig.term_to_genes(i) for i in hits]
create_venn3(*g_sets+hits);
plt.savefig('venn_hiv.png', dpi=450)

```



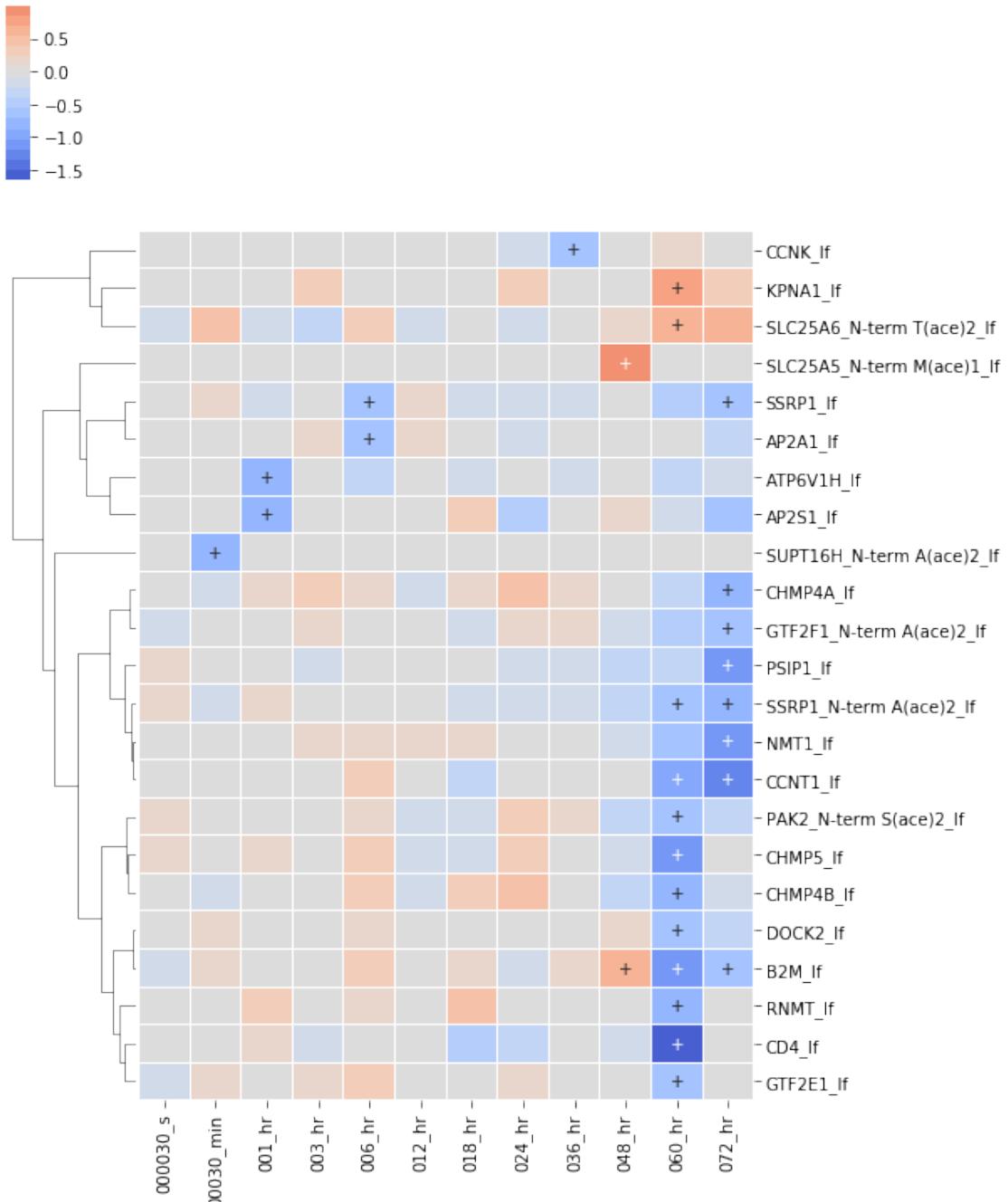
```

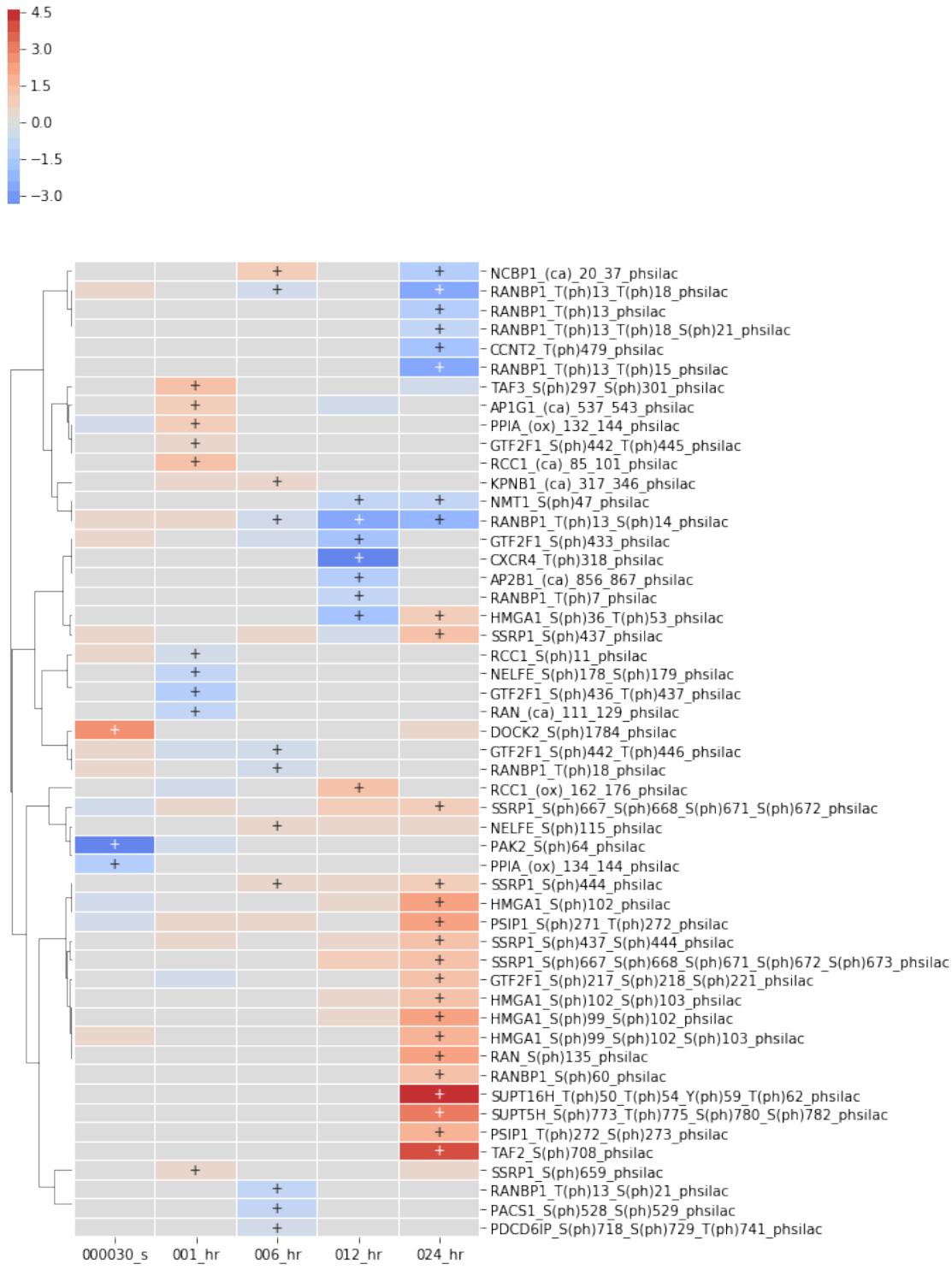
[40]: hiv_only= g_sets[-1]
hiv_only.difference_update(g_sets[0]) # remove dna repair genes

```

```
hiv_only.difference_update(g_sets[1]) # removes cell cycle genes

exp_data.label_free.heatmap(
    hiv_only,
    cluster_row=True,
    subset_index='identifier',
    index='label',
    min_sig=1,
    linewidths=0.01
);
exp_data.ph_silac.heatmap(
    hiv_only,
    subset_index='identifier',
    index='label',
    cluster_row=True,
    rank_index=True,
    min_sig=1,
    figsize=(6, 16),
    linewidths=0.01
);
```





```
[41]: term_net, mol_net = create_subnetwork(
    subset,
    network=network,
```

```
        save_name='hiv',
        use_cytoscape=False,
        use_fdr=True, use_threshold=True, min_edges=10
    )
print(len(mol_net.edges))
print(len(mol_net.nodes))
```

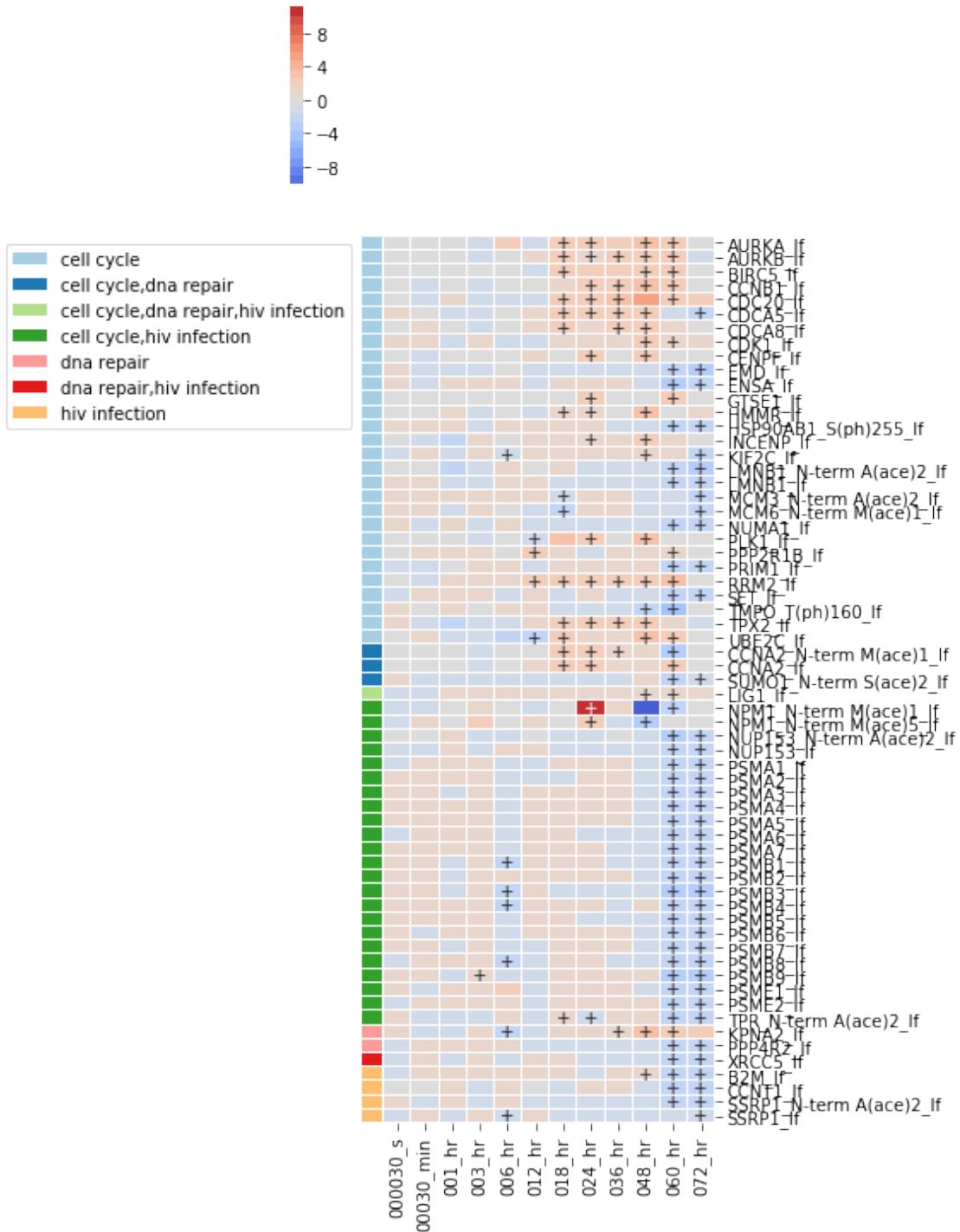
Creating ontology network
2475
316

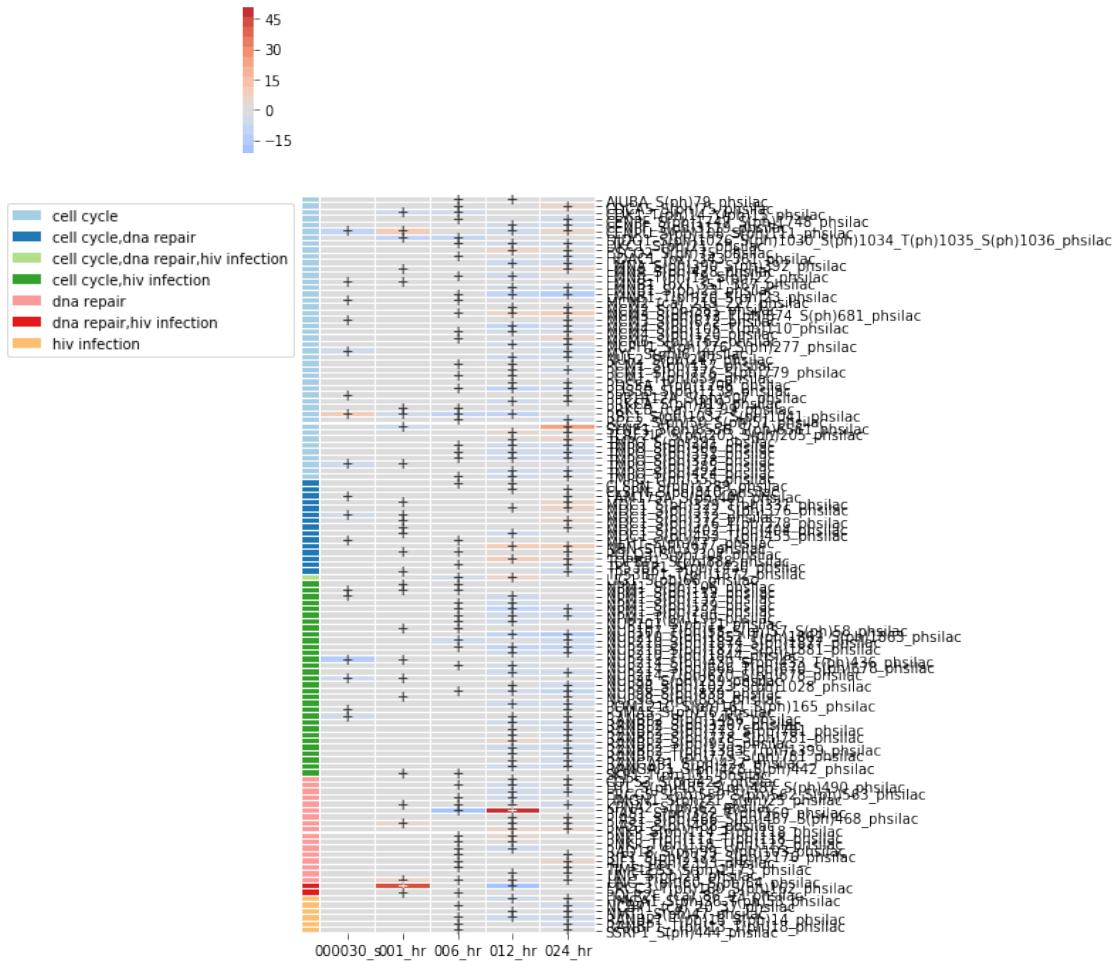
[42]: vis.draw_cyjs(term_net)

<IPython.core.display.HTML object>

[43]: term_to_gene = subset.term_to_genes_dict()

```
heatmap_by_terms(
    exp_data.label_free,
    convert_to_log=False,
    index='label',
    term_labels=list(term_to_gene.keys()),
    term_sets=list(term_to_gene.values()),
    div_colors=True,
    linewidths=0.01,
    min_sig=2,
    annotate_sig=True,
    cluster_col=False,
    figsize=(6,12),
    y_tick_labels=True
);
heatmap_by_terms(
    exp_data.ph_silac,
    convert_to_log=False,
    index='label',
    term_labels=list(term_to_gene.keys()),
    term_sets=list(term_to_gene.values()),
    div_colors=True,
    linewidths=0.01,
    min_sig=2,
    annotate_sig=True,
    cluster_col=False,
    figsize=(6,12),
    y_tick_labels=True
);
```





3.1.6 Exploring a subset of terms based on a keyword

```
[44]: # Filter by terms
damage_terms = enrichment_array.sig.filter_based_on_words(['damage'])
print(len(damage_terms.all_genes_from_df()))
damage_terms.head(10)
damage_terms_all = damage_terms.filter_multi(category='ph_silac_both')
first_damage = damage_terms_all.filter_multi(sample_id='000030_s')
display(first_damage[cols].head(20))

fig = first_damage.heatmap(
    convert_to_log=False,
    cluster_by_set=False,
    annotate_sig=True,
    div_colors=True,
    linewidths=.01,
    num_colors=21,
```

```

    figsize=(3,5)
);
fig.savefig('dna_damage_terms_30s.png', dpi=300, bbox_inches='tight')

damage_terms.require_n_sig(inplace=True, columns='sample_id', n_sig=2)
damage_terms.remove_redundant(inplace=True, threshold=.5, level='sample')
fig = damage_terms.heatmap(
    convert_to_log=False,
    cluster_by_set=False,
    annotate_sig=True,
    div_colors=True,
    linewidths=.01,
    num_colors=21,
    figsize=(4,8)
);
fig.savefig('dna_damage_terms_all.png', bbox_inches='tight')

dna_gene_df = exp_data.subset(first_damage.all_genes_from_df())
dna_gene_df = dna_gene_df.loc[dna_gene_df.sample_id.isin(['000030_s'])]

dna_gene_df.sig.heatmap(
    index='label',
    rank_index=False,
    convert_to_log=True,
    annotate_sig=True,
    div_colors=True,
    linewidths=.01,
    num_colors=21,
    figsize=(2,22)
);

```

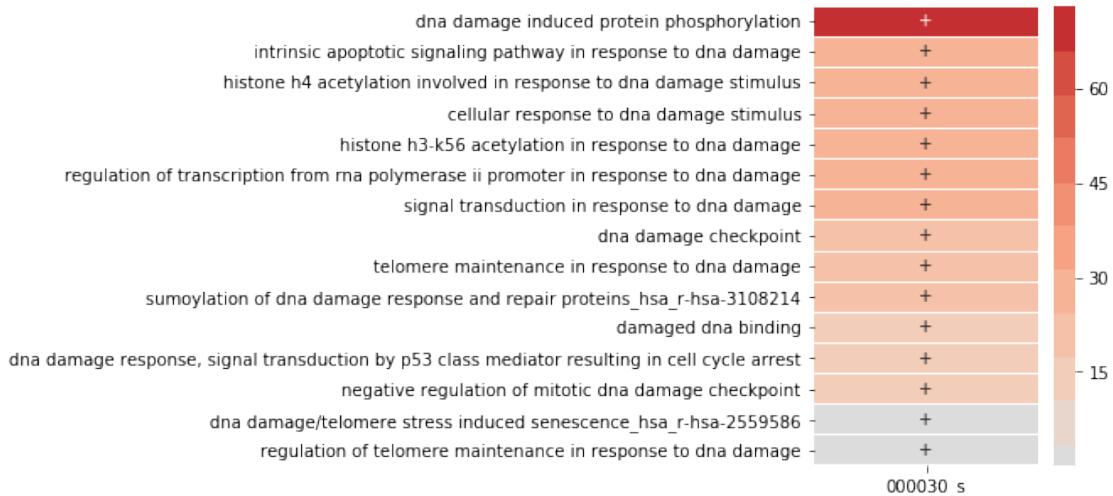
478

	term_name	rank	\
3504789	dna damage induced protein phosphorylation	12	
3504932	intrinsic apoptotic signaling pathway in respo...	155	
3504975	histone h4 acetylation involved in response to...	198	
3504976	cellular response to dna damage stimulus	199	
3504977	histone h3-k56 acetylation in response to dna ...	200	
3504979	regulation of transcription from rna polymeras...	202	
3504989	signal transduction in response to dna damage	212	
3505010	dna damage checkpoint	233	
3505029	telomere maintenance in response to dna damage	252	
3505202	dna damage response, signal transduction by p5...	425	
3505281	negative regulation of mitotic dna damage chec...	504	
3505631	regulation of telomere maintenance in response...	854	
3511095	damaged dna binding	74	
3516776	sumoylation of dna damage response and repair ...	36	

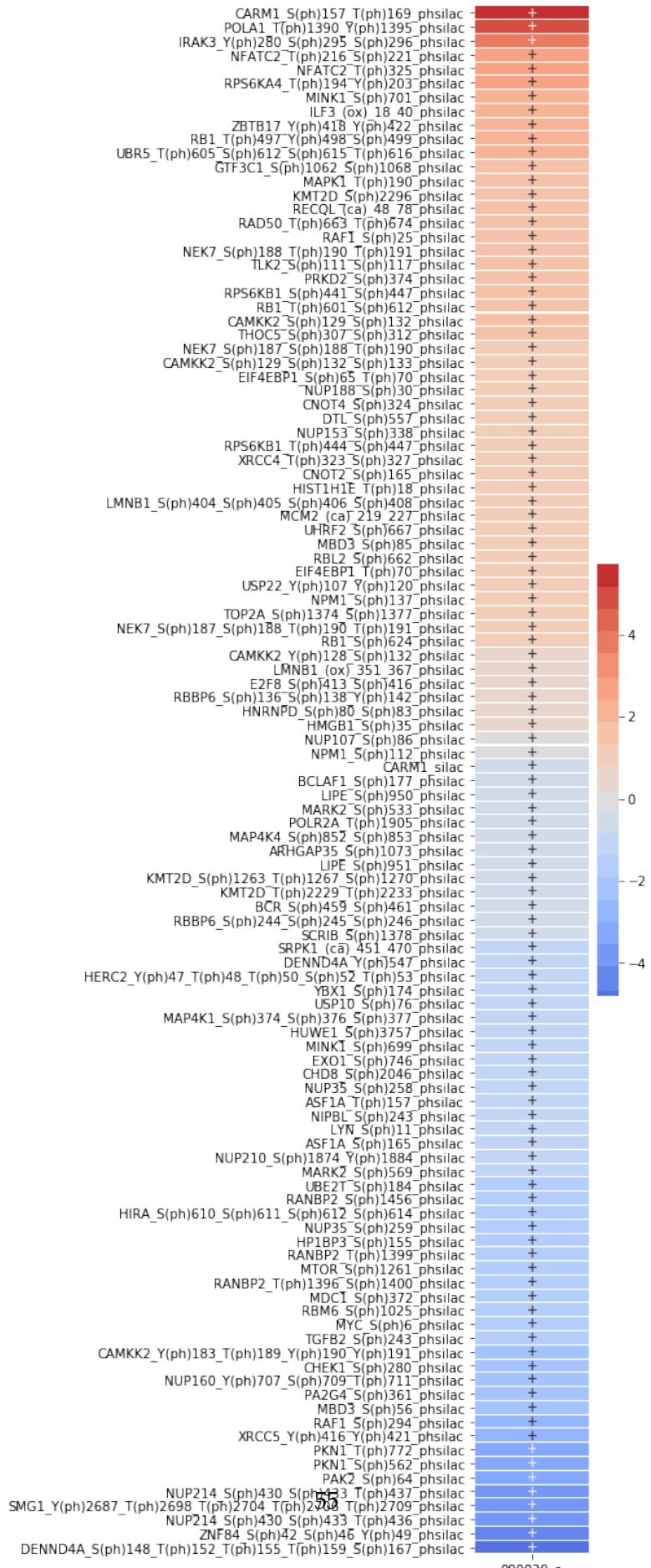
3516900 dna damage/telomere stress induced senescence_... 160

	combined_score	adj_p_value	n_genes	sample_id	category
3504789	73.18	5.42e-05	36	000030_s	ph_silac_both
3504932	29.22	2.92e-03	18	000030_s	ph_silac_both
3504975	25.87	5.85e-03	17	000030_s	ph_silac_both
3504976	25.87	4.08e-03	16	000030_s	ph_silac_both
3504977	25.85	4.08e-03	16	000030_s	ph_silac_both
3504979	25.76	4.26e-03	16	000030_s	ph_silac_both
3504989	24.76	5.45e-03	16	000030_s	ph_silac_both
3505010	23.40	8.45e-03	16	000030_s	ph_silac_both
3505029	21.49	1.27e-02	16	000030_s	ph_silac_both
3505202	12.66	3.12e-02	9	000030_s	ph_silac_both
3505281	11.45	2.22e-02	6	000030_s	ph_silac_both
3505631	0.10	4.32e-02	4	000030_s	ph_silac_both
3511095	17.49	9.51e-03	29	000030_s	ph_silac_both
3516776	21.40	6.71e-05	12	000030_s	ph_silac_both
3516900	0.86	4.80e-02	6	000030_s	ph_silac_both

Number of rows went from 34 to 23







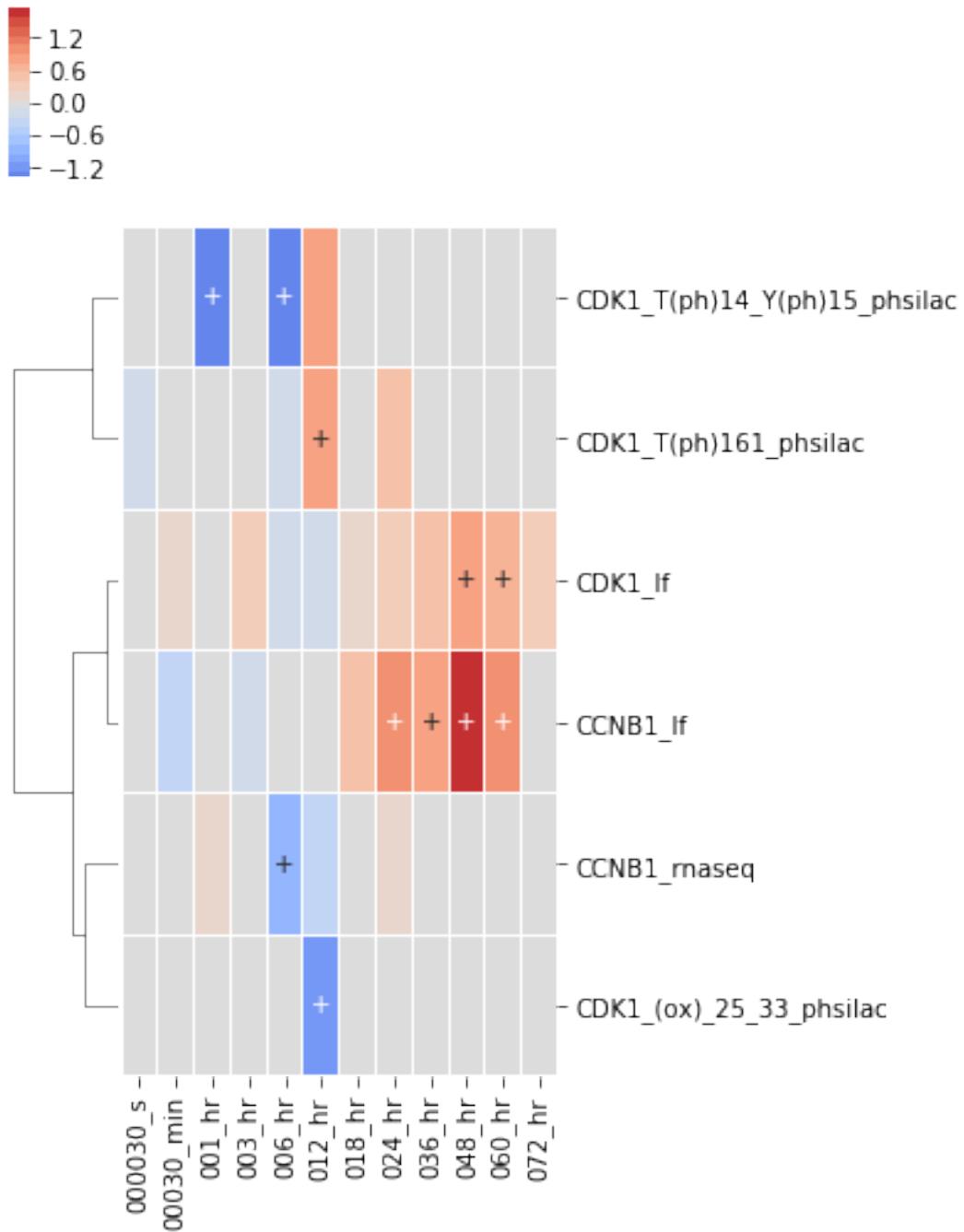
000030_s

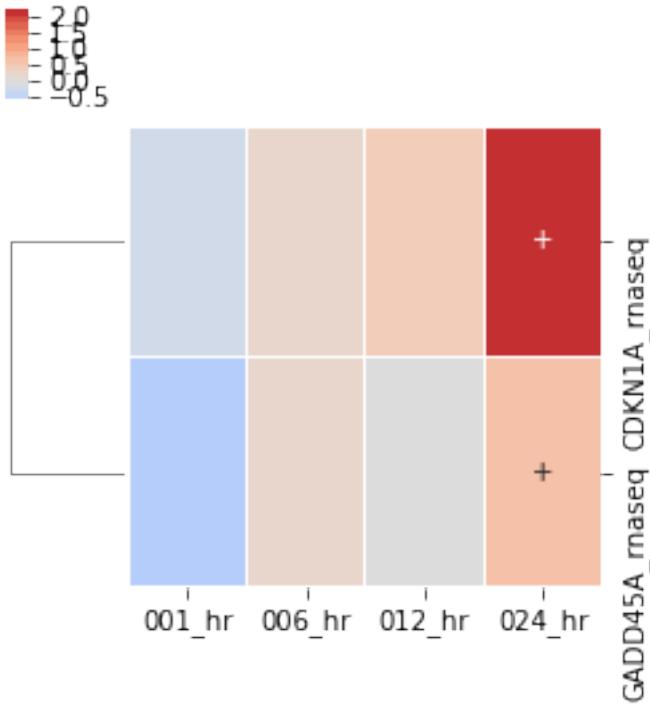
3.1.7 Exploring species of interest

```
[45]: g2_m = ['CDK1', 'CCNB1']
cdk1_inhibitors = ['GADD45A', 'GADD45B', 'GADD45G', 'CDKN1A']

exp_data.species.heatmap(
    g2_m,
    index='label',
    subset_index='identifier',
    figsize=(4,8),
    linewidths=0.01,
    cluster_row=True,
    min_sig=1
);

exp_data.species.heatmap(
    cdk1_inhibitors,
    index='label',
    subset_index='identifier',
    figsize=(4, 4),
    linewidths=0.01,
    cluster_row=True,
    min_sig=1
);
```



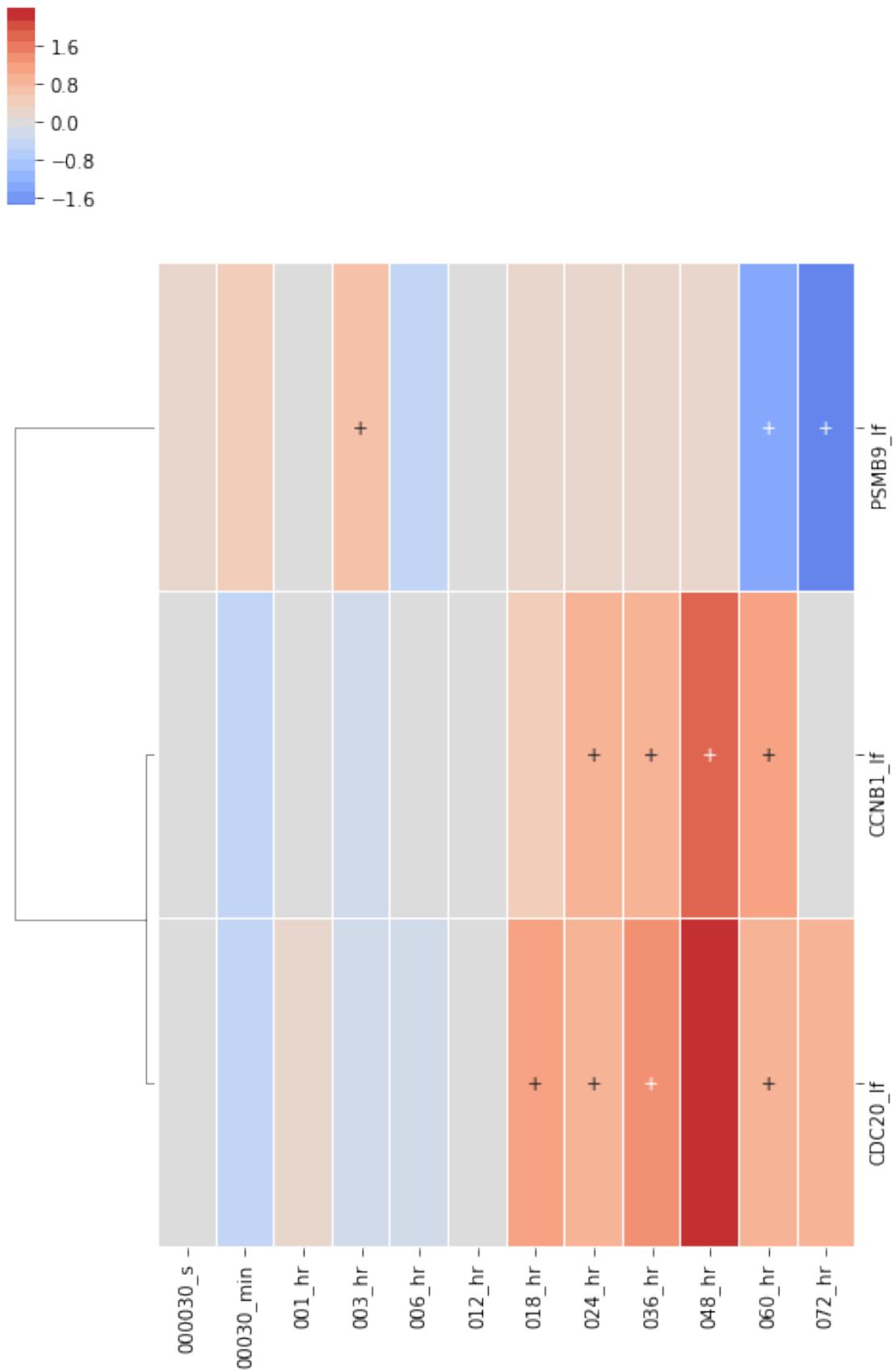


```
[46]: expand_neigh = net_sub.expand_neighbors(
    network=None,
    nodes=g2_m,
    downstream=True,
    upstream=True,
    max_dist=1,
    include_only=reactome_only.filter_multi(category=['label_free_up']),
    term_to_genes('cell_cycle'),
    add_interconnecting_edges=False,
)

expand_neigh = net_sub.paths_between_two_lists(
    reactome_only.filter_multi(category=['ph_silac_up']).term_to_genes('dna_repair'),
    g2_m,
    max_length=3,
    include_only=reactome_only.filter_multi(category=['label_free_up']),
    term_to_genes('cell_cycle'),
    add_interconnecting_edges=True
)
print(len(expand_neigh.nodes))
print(len(expand_neigh.edges))
expand_neigh = utils.delete_disconnected_network(expand_neigh)
exp_data.label_free.heatmap()
```

```
expand_neigh.nodes,
subset_index='identifier',
index='label',
rank_index=False,
cluster_row=True,
min_sig=3,
annotate_sig=True,
linewidths=0.01
);
```

```
Warning : 2 do not exist in graph
Removing from list
25
47
```



```
[47]: #expand_neigh = utils.add_attribute_to_network(expand_neigh, g2_m, 'color', ▾
    ↪ 'red', 'white')
vis.draw_cyjs(
    expand_neigh, layout='dagre',
    spacingFactor=1.,
    nodeRepulsion=100, gravity=.1,
    rankDir='TB', nodeSep=5, rankSep=25, ranker='longest-path'
)
```

<IPython.core.display.HTML object>

```
[48]: expand_neigh = net_sub.paths_between_two_lists(
    reactome_only.filter_multi(category=['ph_silac_up']).term_to_genes('dna' ▾
    ↪ 'repair'),
    g2_m,
    reverse=True,
    max_length=3,
    include_only=exp_data.species.sig,
    add_interconnecting_edges=False
)

expand_neigh = utils.delete_disconnected_network(expand_neigh)

print(len(expand_neigh.nodes))
print(len(expand_neigh.edges))
```

Warning : 2 do not exist in graph
Removing from list
29
38

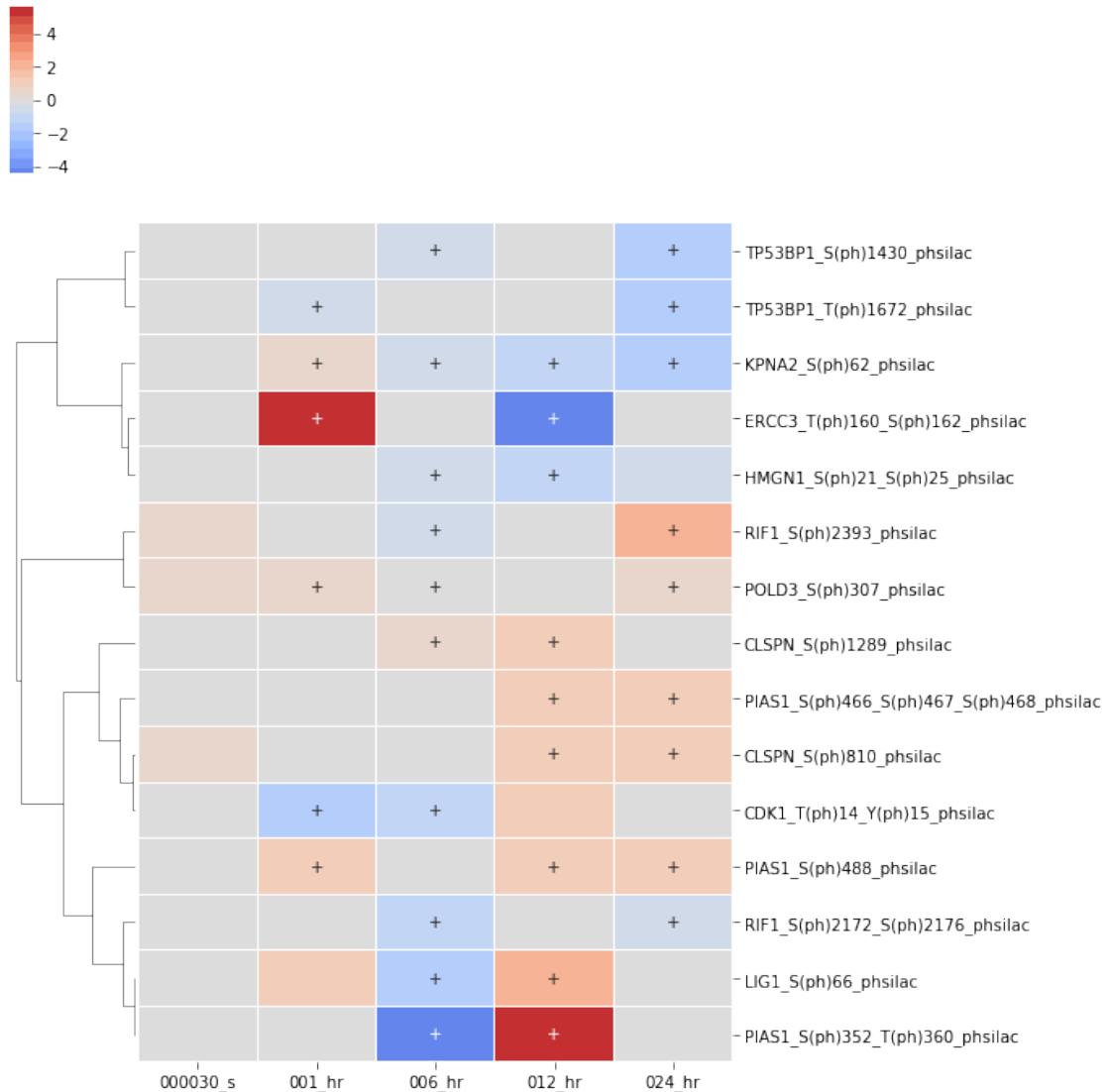
```
[49]: vis.draw_cyjs(
    expand_neigh, layout='dagre',
    spacingFactor=1.,
    nodeRepulsion=100, gravity=.1,
    rankDir='TB', nodeSep=5, rankSep=25, ranker='longest-path'
)
```

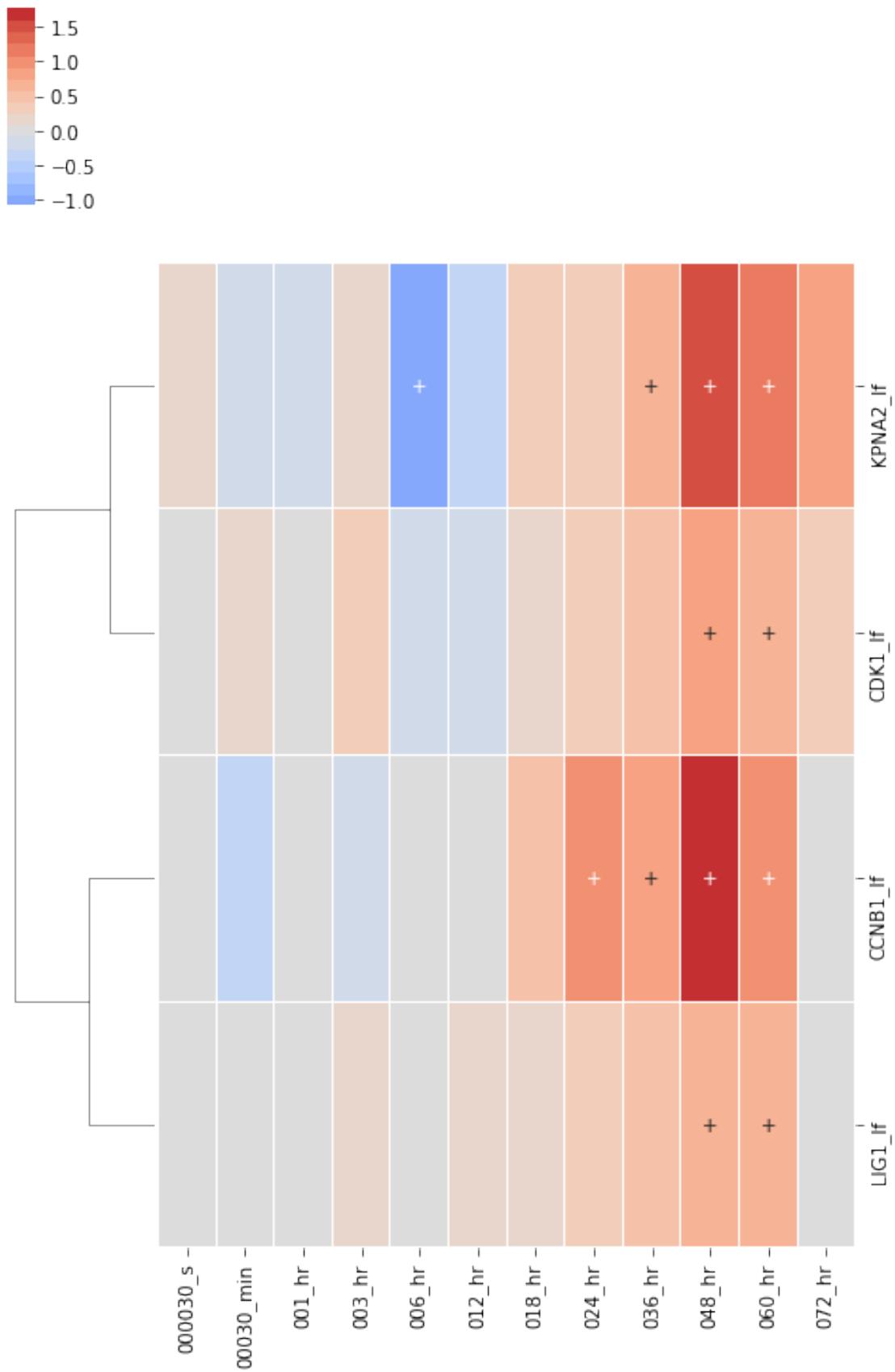
<IPython.core.display.HTML object>

```
[50]: exp_data.ph_silac.heatmap(
    expand_neigh.nodes,
    subset_index='identifier',
```

```
    index='label',
    rank_index=False,
    cluster_row=True,
    min_sig=2,
    linewidths=0.01
);

exp_data.label_free.heatmap(
    expand_neigh.nodes,
    subset_index='identifier',
    index='label',
    rank_index=False,
    cluster_row=True,
    min_sig=2,
    linewidths=0.01
);
```

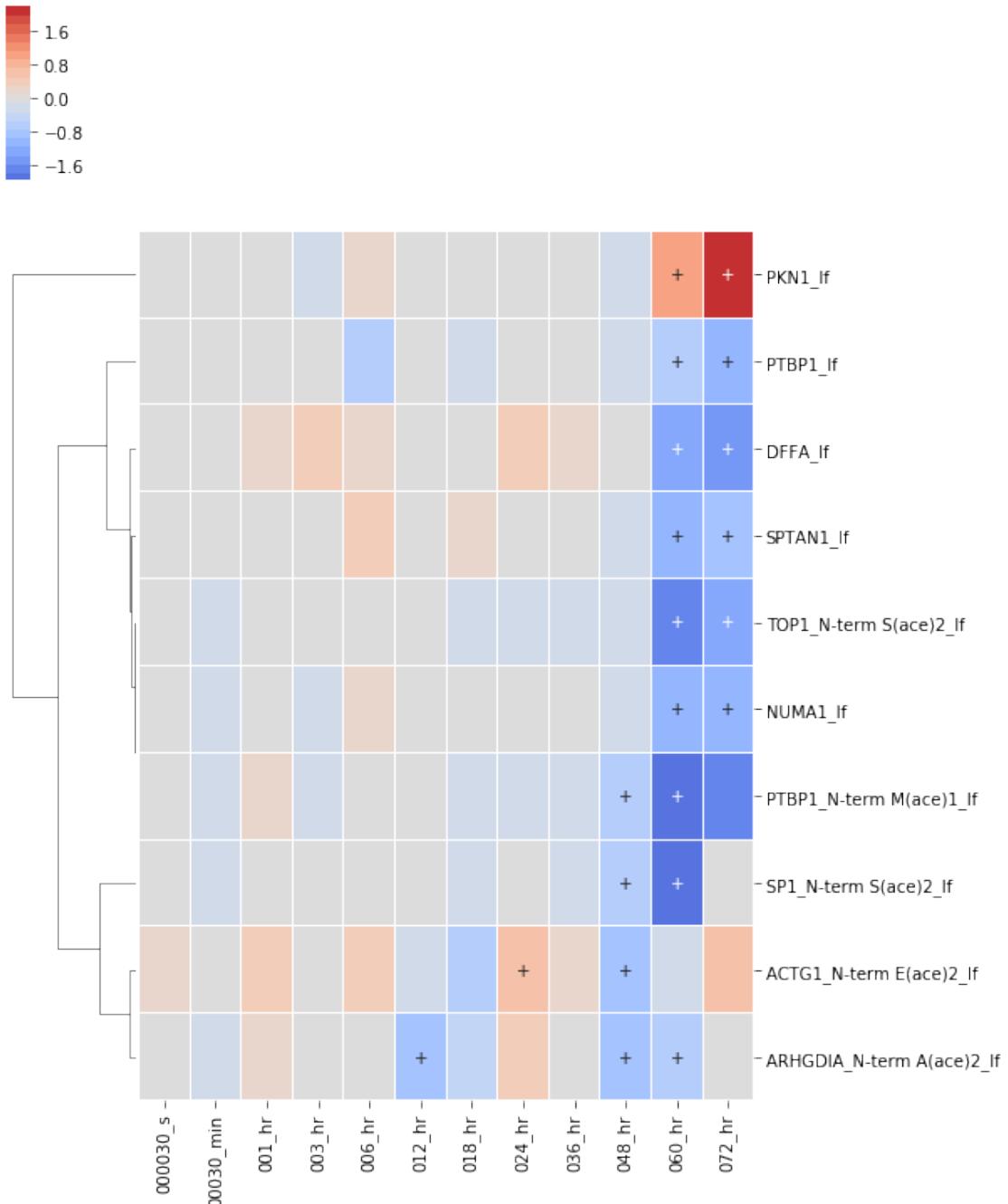


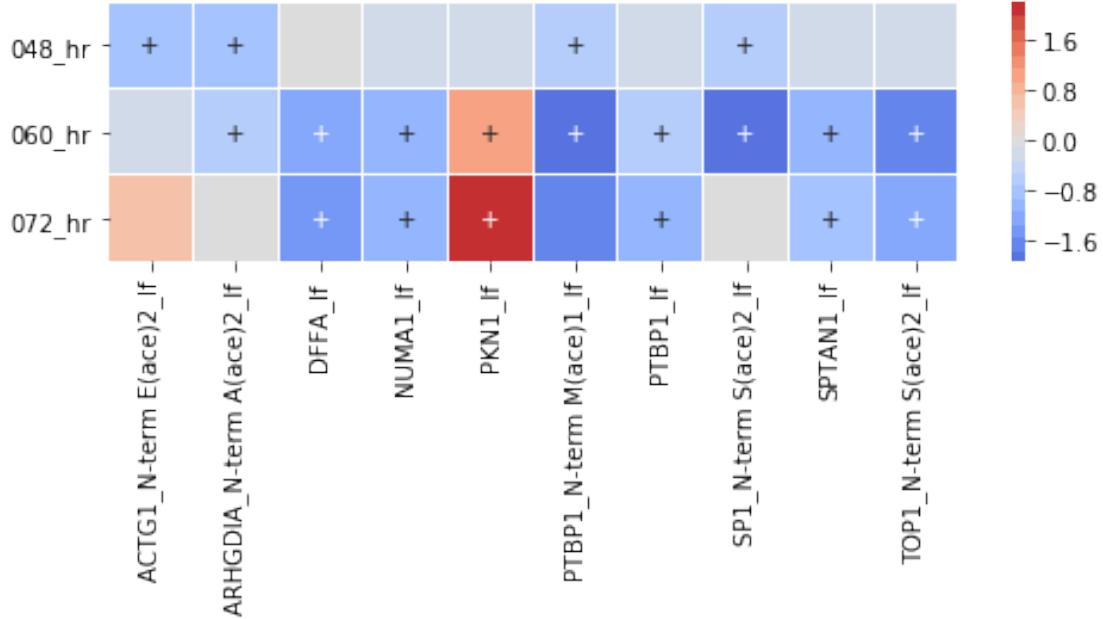


```
[51]: down_casp3 = net_sub.expand_neighbors(
    network=None,
    nodes=['CASP3'],
    upstream=False, downstream=True,
    max_dist=1,
    include_only=exp_data.label_free.sig.require_n_sig(index='label', n_sig=2).
    →id_list
)
down_casp3 = utils.delete_disconnected_network(down_casp3)
print(len(down_casp3.nodes))
print(len(down_casp3.edges))
exp_data.label_free.heatmap(
    down_casp3.nodes,
    subset_index='identifier', index='label',
    min_sig=2, linewidths=0.01,
    cluster_row=True
);
exp_data.label_free.require_n_sig(index='label', n_sig=2).heatmap(
    down_casp3.nodes, subset_index='identifier',
    #index='label',
    rank_index=True,
    figsize=(8, 2),
    index='sample_id', columns='label',
    min_sig=2, linewidths=0.01, cluster_row=False, cluster_col=False
);
plt.yticks(rotation=0)
plt.savefig('down_from_casp3.png', dpi=300, bbox_inches='tight')
```

10

9





```
[52]: vis.draw_cyjs(down_casp3)
```

<IPython.core.display.HTML object>

```
[53]: def show_neighbors(node, df, upstream=True, downstream=False, max_dist=1,
                      include_only=None, figsize=None):

    df_copy = df.copy()
    df_copy.require_n_sig(n_sig=1, inplace=True)

    neighbors = net_sub.expand_neighbors(
        network=None,
        nodes=[node],
        upstream=upstream,
        downstream=downstream,
        max_dist=max_dist,
        include_only=include_only
    )

    neighbors = utils.delete_disconnected_network(neighbors)
    s_name = 'node_{}.png'.format(node)
    exporters.export_to_dot(neighbors, s_name, image_format='png', ↪
    engine='circo')
    display(Image(s_name, width=400))

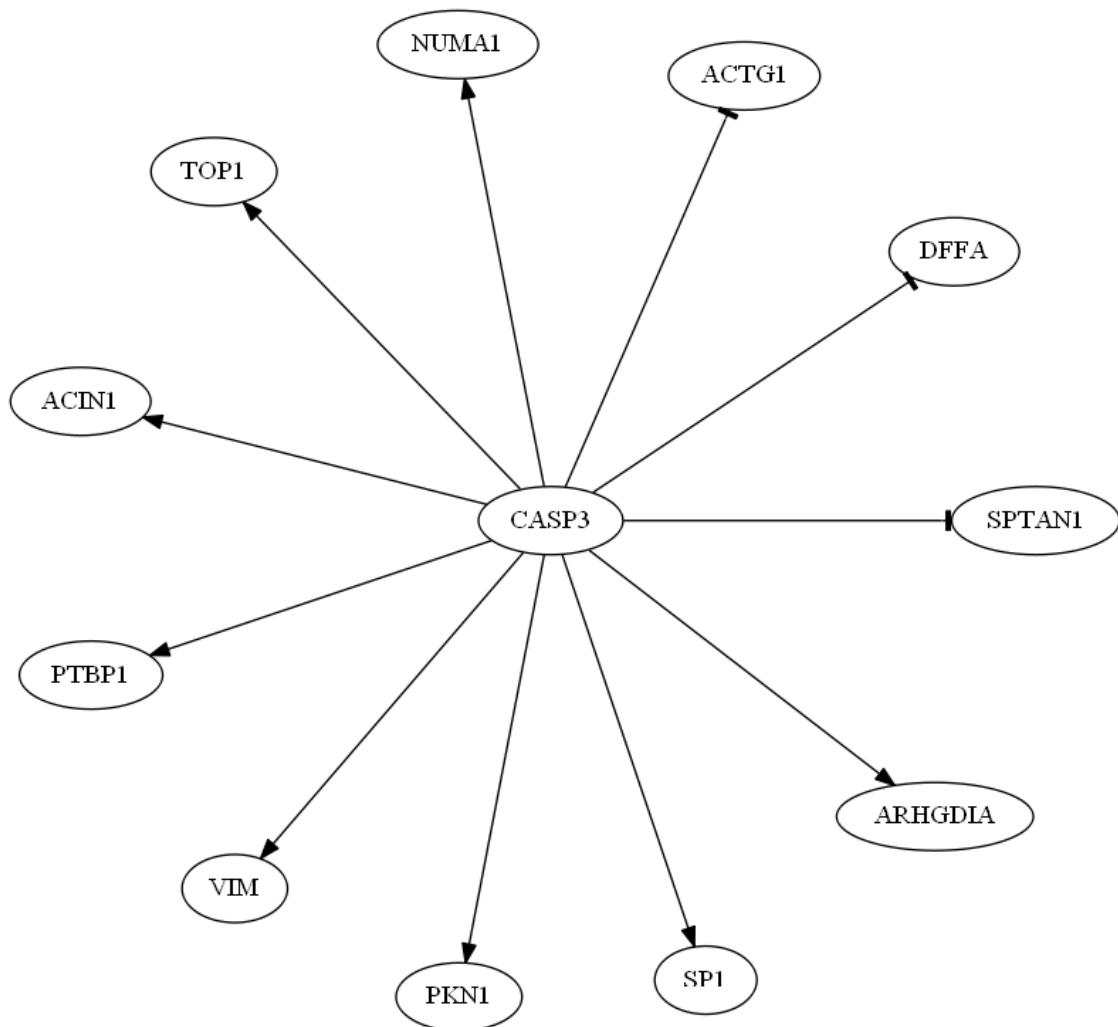
    g = df_copy.heatmap(
```

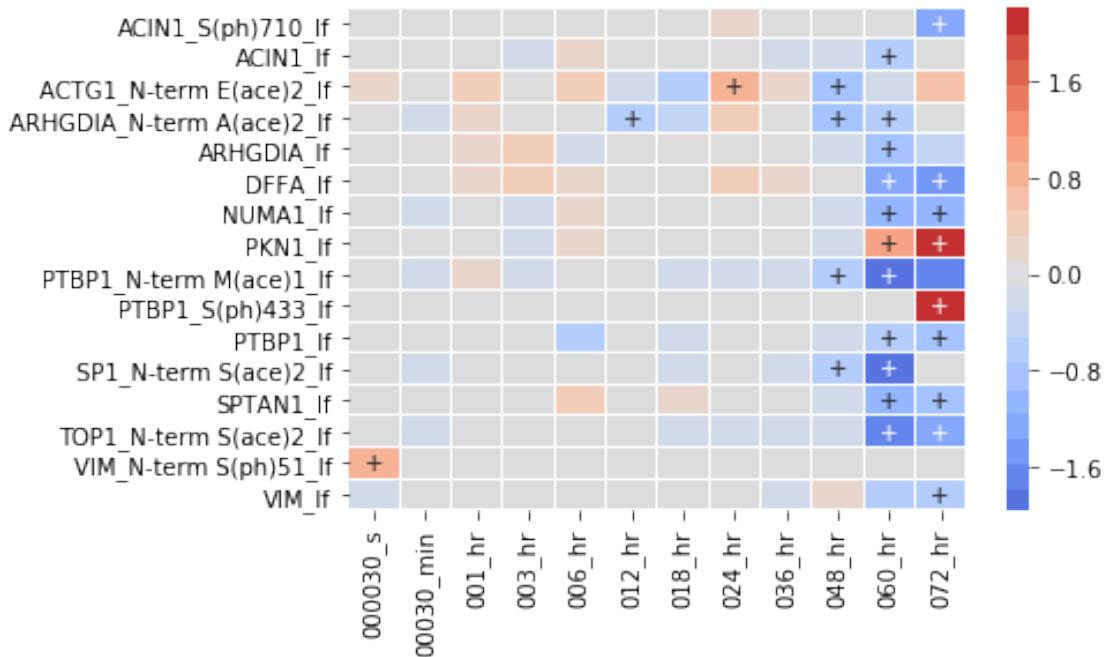
```

        sorted(neighbors.nodes),
        subset_index='identifier',
        index='label',
        min_sig=1,
        rank_index=True,
        linewidths=0.01,
        figsize=figsize
    );

    g.savefig('{}_heatmap.png'.format(node), bbox_inches='tight', dpi=300)
show_neighbors('CASP3', exp_data.label_free, False, True, max_dist=1,
               include_only=exp_data.label_free.sig.require_n_sig(n_sig=2).
               →id_list)

```





```
[54]: show_neighbors('BAX', exp_data.species, False, True, max_dist=1,
                   include_only=exp_data.species.sig.require_n_sig(n_sig=1).id_list)
```

