1. Overview of node perturbation analysis

In this section, we have described the input file and script file used for node perturbation analysis. Then for each simulation (simulations 1-4), we have shown how all the five perturbation conditions were specified and the results were analyzed.

2.1 Description of files used for node perturbation (NP) analysis

Input file: Based on the inward interaction of each node (gene, miRNA and TF) and the type of simulation (given below), a transfer function represented by Boolean operators was given for each node. Transfer function was also given for each stage of development.

For example, when Gene A and miRNA-c is regulated by TF1:

Simulation 1 (TF activates gene and miRNA expression), Transfer function for Gene A = TF1; miRNA-c = TF1

Simulation 2 (TF represses gene and miRNA expression), Transfer function for Gene A = not(TF1); miRNA-c =not(TF1)

Simulation 3 (TF activates gene and represses miRNA expression), Transfer function for Gene A = TF1; miRNA-c = not(TF1)

Simulation 4 (TF activates miRNA and represses gene expression), Transfer function for Gene A = not(TF1); miRNA-c = TF1

Script file for running perturbation analysis: For each simulation, all the five perturbation conditions (PC) along with the input text file were specified in a single Python script file. In the script, code was also written to save the perturbation results in bin file and export the results as excel sheet.

Five perturbation conditions (PC) given for each stage of development at each simulation:

PC1: OE as well as KO of each miRNA and each TF regulating GRM5 interactome

PC2: OE of genes positively regulating each stage (regulated by GRM5 interactome) and OE of miRNAs and TFs regulating GRM5 interactome

PC3: KO of genes positively regulating each stage (regulated by AP interactome) and KO of miRNAs and TFs regulating GRM5 interactome

PC4: OE of genes positively regulating each stage (regulated by GRM5 interactome) and OE of each factor (miRNA/ TF) regulating GRM5 interactome

PC5: KO of genes positively regulating each stage (regulated by GRM5 interactome) and KO of each factor (miRNA/ TF) regulating GRM5 interactome

2.2 Node perturbation analysis (NP analysis)

We have showed the steps followed for node perturbation analysis of GRM5 interactome. In each simulation (Simulations 1-4), we have specified five perturbation conditions (PCs) for each neurodevelopmental stage and analyzed the perturbation effect on different stages of neurodevelopment. For example, in simulation 1, we gave the input text file (Step 1) and specified five PCs for the stage proliferation in the script file (Step 2) to analyze the perturbation effect of regulatory factors along with genes regulating proliferation on each stage of development (Step 3). Similarly, we specified five PC for other neurodevelopmental stages regulated by GRM5 interactome and performed NP analysis. We followed the same steps 1, 2 and 3 for performing NP analysis at simulations 2, 3 and 4.

Simulation 1

Step1: Input file for Simulation 1 (TF activates gene and miRNA expression)

```
1: GRM5* = not(miR335 or miR4719) and (RUNX2 or TAL1 or REST or YAP1)
and (CAM and SHANK3 and SIAH1 and GRASP and NECAB2 and LRRC7 and DNM2)
1: ERK1* = GRM5
1: ERK2* = not(miR335) and (RUNX2) and (GRM5)
1: JNK2* = GRM5
1: MAPK14* = not(GRM5 or miR4719) and (TAL1)
1: CCND1* = GRM5
1: RUNX2* = not(miR335)
```

```
1: miR335* = YAP1 or TAL1
1: YAP1* = not(miR335)
1: TAL1* = not(miR4719)
1: miR4719* = miR4719
1: CAM* = CAM
1: SHANK3* = SHANK3
1: SIAH1* = SIAH1
1: GRASP* = not(miR335) and (REST)
1: PKD* = GRM5
1: NECAB2* = NECAB2
1: LRRC7* = LRRC7
1: DISC1* = LRRC7
1: DNM2* =  DNM2
1: miR137* = GRM5
1: GRIA1* = not(miR137 or miR335) and (REST or TAL1 or YAP1)
1: PKC* = GRM5
1: UBE2I* = PKC
1: REST* = not(miR335)
1: Proliferation* = not(MAPK14) and (GRM5 and ERK1 and ERK2 and JNK2
and CCND1)
1: Differentiation* = GRM5 and ERK1 and ERK2
1: Synaptogenesis* =  not(GRIA1) and (CAM and GRM5 and SHANK3 and SIAH1
and GRASP and PKD and NECAB2 and LRRC7 and DNM2 and miR137 and PKC and
UBE2I and DISC1)
1: Neurodevelopment* = not(GRIA1 or MAPK14) and (ERK1 and ERK2 and JNK2
and CCND1 and CAM and GRM5 and SHANK3 and SIAH1 and GRASP and PKD and
NECAB2 and LRRC7 and DNM2 and miR137 and PKC and UBE2I and DISC1)
```

Step 2: Specify the five perturbation conditions (PC1-PC5) for each stage regulated by

GRM5 interactome

PC1: OE as well as KO of each miRNA and each TF regulated by GRM5 interactome (remains same for each stage)

PC for the stage proliferation regulated by GRM5 interactome at Simulation 1

PC2: OE of genes positively regulating proliferation (regulated by GRM5 interactome) and OE of miRNAs and TFs regulating GRM5 interactome

PC3: KO of genes positively regulating proliferation (regulated by GRM5 interactome) and KO of miRNAs and TFs regulating GRM5 interactome

PC4: OE of genes positively regulating proliferation (regulated by GRM5 interactome) and OE of each factor (miRNA/ TF) regulating GRM5 interactome

PC5: KO of genes positively regulating proliferation (regulated by GRM5 interactome) and KO of each factor (miRNA/ TF) regulating GRM5 interactome

PC for the stage differentiation regulated by GRM5 interactome at Simulation 1

PC2: OE of genes positively regulating differentiation (regulated by GRM5 interactome) and OE of miRNAs and TFs regulating GRM5 interactome

PC3: KO of genes positively regulating differentiation (regulated by GRM5 interactome) and KO of miRNAs and TFs regulating GRM5 interactome

PC4: OE of genes positively regulating differentiation (regulated by GRM5 interactome) and OE of each factor (miRNA/ TF) regulating GRM5 interactome

PC5: KO of genes positively regulating differentiation (regulated by GRM5 interactome) and KO of each factor (miRNA/ TF) regulating GRM5 interactome

PC for the stage synaptogenesis regulated by GRM5 interactome at Simulation 1

PC2: OE of genes positively regulating synaptogenesis (regulated by GRM5 interactome) and OE of miRNAs and TFs regulating GRM5 interactome

PC3: KO of genes positively regulating synaptogenesis (regulated by GRM5 interactome) and KO of miRNAs and TFs regulating GRM5 interactome

PC4: OE of genes positively regulating synaptogenesis (regulated by GRM5 interactome) and OE of each factor (miRNA/ TF) regulating GRM5 interactome

PC5: KO of genes positively regulating synaptogenesis (regulated by GRM5 interactome) and KO of each factor (miRNA/ TF) regulating GRM5 interactome

PC for overall neurodevelopment process (all stages) regulated by GRM5 interactome at Simulation 1

PC2: OE of genes positively regulating overall neurodevelopment process (regulated by GRM5 interactome) and OE of miRNAs and TFs regulating GRM5 interactome

PC3: KO of genes positively regulating overall neurodevelopment process (regulated by GRM5 interactome) and KO of miRNAs and TFs regulating GRM5 interactome

PC4: OE of genes positively regulating overall neurodevelopment process (regulated by GRM5 interactome) and OE of each factor (miRNA/ TF) regulating GRM5 interactome

PC5: KO of genes positively regulating overall neurodevelopment process (regulated by GRM5 interactome) and KO of each factor (miRNA/ TF) regulating GRM5 interactome

Script 1: Script for perturbation of genes regulating proliferation on stages of neurodevelopment (Simulation 1, PC1-PC5)

```
"""GRM5_ap simulator (Proliferation)

It is also a demonstration on how the collector works

"""
import boolean2
from boolean2 import Model, util
from random import choice
import xlsxwriter
import pandas as pd
from pylab import *
from boolean2 import util
import matplotlib.pyplot as plt


# occasionally randomized nodes
TARGETS = set( "Proliferation".split() )
TARGETS = set( "Differentiation".split() )
TARGETS = set( "Synaptogenesis".split() )
TARGETS = set( "Neurodevelopment".split() )

def new_getvalue( state, name, p):
    """
    Called every time a node value is used in an expression.
    It will override the value for the current step only.
    Returns random values for the node states
    """
    global TARGETS
    value = util.default_get_value( state, name, p )

    if name in TARGETS:
        # pick at random from True, False and original value
        return choice( [True, False, value] )
    else:
        return value

def run( text, nodes, repeat, steps ):
    """
    Runs the simulation and collects the nodes into a collector,
    a convenience class that can average the values that it collects.
    """
    coll = util.Collector()

    for i in xrange( repeat ):
        engine  = Model( mode='async', text=text )
        engine.RULE_GETVALUE = new_getvalue
        # minimalist initial conditions, missing nodes set to false
        engine.initialize( missing=util.false )
        engine.iterate( steps=steps)
        coll.collect( states=engine.states, nodes=nodes )

    print '- completed'
    avgs = coll.get_averages( normalize=True )
    return avgs

if __name__ == '__main__':
    text = file( 'Simulation1.txt').read() # Input file for Simulation
1
```

```
    # this collects the state of all nodes
    NODES  = boolean2.all_nodes( text )

    # REPEAT = 1000, STEPS=150
    REPEAT = 1000
    STEPS  = 150

    data = []

    print '- starting simulation with REPEAT=%s, STEPS=%s' % (REPEAT,
STEPS)
    mtext = boolean2.modify_states( text=text, turnon=['GRM5', 'ERK1',
'ERK2', 'JNK2','CCND1','miR335', 'miR4719', 'RUNX2', 'TAL1', 'REST',
'YAP1' ] ) # Highlighted genes positively regulating proliferation
avgs = run( text=mtext, repeat=REPEAT, nodes=NODES, steps=STEPS)
    data.append( avgs ) # Perturbation condition 2
    mtext = boolean2.modify_states( text=text, turnoff=['GRM5', 'ERK1',
'ERK2', 'JNK2','CCND1','miR335', 'miR4719', 'RUNX2', 'TAL1', 'REST',
'YAP1' ] )
    avgs = run( text=mtext, repeat=REPEAT, nodes=NODES, steps=STEPS)
    data.append( avgs ) # Perturbation condition 3
# Single node perturbation
factorslist = ['miR335', 'miR4719', 'RUNX2', 'TAL1', 'REST', 'YAP1'] #
miRNAs and TFs regulating GRM5 interactome
for i in factorslist:
        mtext = boolean2.modify_states( text=text, turnon=[i] )
        avgs = run( text=mtext, repeat=REPEAT, nodes=NODES,
steps=STEPS)
        data.append( avgs ) # Perturbation condition 1
        mtext = boolean2.modify_states( text=text, turnoff=[i] )
        avgs = run( text=mtext, repeat=REPEAT, nodes=NODES,
steps=STEPS)
        data.append( avgs ) # Perturbation condition 1
        mtext = boolean2.modify_states( text=text, turnon=['GRM5',
'ERK1', 'ERK2', 'JNK2','CCND1', i] )
        avgs = run( text=mtext, repeat=REPEAT, nodes=NODES,
steps=STEPS)
        data.append( avgs ) # Perturbation condition 4
        mtext = boolean2.modify_states( text=text, turnoff=['GRM5',
'ERK1', 'ERK2', 'JNK2','CCND1', i] )
        avgs = run( text=mtext, repeat=REPEAT, nodes=NODES,
steps=STEPS)
        data.append( avgs ) # Perturbation condition 5
        fname = 'GRM5_proliferation_factors.bin'
        util.bsave( data, fname=fname )
        print '- data saved into %s' % fname
       data = util.bload( 'GRM5_proliferation_factors.bin') #
Perturbation results
        avgs = data
        df = pd.DataFrame.from_dict(data)
        df.to_excel("proliferation_factors_sim1_clubbed.xlsx")
        #print (df)

        avgs = data
        for x in data:
                f = lambda x:'Value {}'.format(x+1)
          d = {x: pd.DataFrame(df[x].values.tolist(),
index=df.index).rename(columns=f)
```

```
            for x in df.columns}
            df_1 = pd.concat(d, axis=1)

# Effect of perturbations (PC1-PC5) on each stage of development at t =
0 and t = 150
df_1.columns.get_loc("Proliferation")
df_1.columns.get_loc("Differentiation")
df_1.columns.get_loc("Synaptogenesis")
df_1.columns.get_loc("Neurodevelopment")

df_1= df_1.iloc[:, [2567, 2717, 604,754, 3322, 3472, 2114, 2264]]
df_1.to_excel('proliferation_factors_sim1.xlsx')
```

Script 2: Script for perturbation of genes regulating differentiation on each stage of development (Simulation 1, PC1-PC5)

```
"""GRM5_ap simulator (Differentiation)

It is also a demonstration on how the collector works

"""
import boolean2
from boolean2 import Model, util
from random import choice
import xlsxwriter
import pandas as pd
from pylab import *
from boolean2 import util
import matplotlib.pyplot as plt


# occasionally randomized nodes
TARGETS = set( "Proliferation".split() )
TARGETS = set( "Differentiation".split() )
TARGETS = set( "Synaptogenesis".split() )
TARGETS = set( "Neurodevelopment".split() )

def new_getvalue( state, name, p):
    """
    Called every time a node value is used in an expression.
    It will override the value for the current step only.
    Returns random values for the node states
    """
    global TARGETS
    value = util.default_get_value( state, name, p )

    if name in TARGETS:
        # pick at random from True, False and original value
        return choice( [True, False, value] )
    else:
        return value

def run( text, nodes, repeat, steps ):
    """
    Runs the simulation and collects the nodes into a collector,
    a convenience class that can average the values that it collects.
    """
    coll = util.Collector()

    for i in xrange( repeat ):
        engine  = Model( mode='async', text=text )
        engine.RULE_GETVALUE = new_getvalue
        # minimalist initial conditions, missing nodes set to false
        engine.initialize( missing=util.false )
        engine.iterate( steps=steps)
        coll.collect( states=engine.states, nodes=nodes )

    print '- completed'
    avgs = coll.get_averages( normalize=True )
    return avgs

if __name__ == '__main__':
    text = file( 'Simulation1.txt').read()# Input file for Simulation 1
# this collects the state of all nodes
```

```
    NODES  = boolean2.all_nodes( text )

    #
    # REPEAT = 1000, STEPS=150
    REPEAT = 1000
    STEPS  = 150

    data = []

    print '- starting simulation with REPEAT=%s, STEPS=%s' % (REPEAT,
STEPS)
    mtext = boolean2.modify_states( text=text, turnon=['GRM5', 'ERK1'
'ERK2', 'miR335', 'miR4719', 'RUNX2', 'TAL1', 'REST', 'YAP1' ] ) #
highlighted genes positively regulating differentiation
    avgs = run( text=mtext, repeat=REPEAT, nodes=NODES, steps=STEPS)
    data.append( avgs ) # Perturbation condition 2
    mtext = boolean2.modify_states( text=text, turnoff=['GRM5', 'ERK1
'ERK2', 'miR335', 'miR4719', 'RUNX2', 'TAL1', 'REST', 'YAP1' ] )
    avgs = run( text=mtext, repeat=REPEAT, nodes=NODES, steps=STEPS)
    data.append( avgs ) # Perturbation condition 3
 # Single node perturbation
    factorslist = ['miR335', 'miR4719', 'RUNX2', 'TAL1', 'REST',
'YAP1'] # miRNAs and TFs regulating GRM5 interactome
for i in factorslist:
        mtext = boolean2.modify_states( text=text, turnon=[i] )
        avgs = run( text=mtext, repeat=REPEAT, nodes=NODES,
steps=STEPS)
        data.append( avgs ) # Perturbation condition 1
         mtext = boolean2.modify_states( text=text, turnoff=[i] )
        avgs = run( text=mtext, repeat=REPEAT, nodes=NODES,
steps=STEPS)
        data.append( avgs ) # Perturbation condition 1
mtext = boolean2.modify_states( text=text, turnon=['GRM5', 'ERK1',
'ERK2',i] )
        avgs = run( text=mtext, repeat=REPEAT, nodes=NODES,
steps=STEPS)
        data.append( avgs ) # Perturbation condition 4
        mtext = boolean2.modify_states( text=text, turnoff=['GRM5',
'ERK1', 'ERK2', i] )
        avgs = run( text=mtext, repeat=REPEAT, nodes=NODES,
steps=STEPS)
        data.append( avgs ) # Perturbation condition 5
        fname = 'GRM5_differentiation_factors.bin'
        util.bsave( data, fname=fname )
        print '- data saved into %s' % fname
       data = util.bload( 'GRM5_differentiation_factors.bin') #
Perturbation results
        avgs = data
        df = pd.DataFrame.from_dict(data)
        df.to_excel("differentaition_factors_sim1_clubbed.xlsx")
        #print (df)

        avgs = data
        for x in data:
            f = lambda x:'Value {}'.format(x+1)
           d = {x: pd.DataFrame(df[x].values.tolist(),
index=df.index).rename(columns=f)
          for x in df.columns}
```

```
          df_1 = pd.concat(d, axis=1)

# Effect of perturbations (PC1-PC5) on each stage of development at t=0
and t =  150
df_1.columns.get_loc("Proliferation")
df_1.columns.get_loc("Differentiation")
df_1.columns.get_loc("Synaptogenesis")
df_1.columns.get_loc("Neurodevelopment")


df_1= df_1.iloc[:, [2567, 2717, 604,754, 3322, 3472, 2114, 2264]]
df_1.to_excel('differentiation_factors_sim1.xlsx')
```

Script 3: Script for perturbation of genes regulating synaptogenesis on each stage of development (Simulation 1, PC1-PC5)

```
"""GRM5_ap simulator (Synaptogenesis)

It is also a demonstration on how the collector works

"""
import boolean2
from boolean2 import Model, util
from random import choice
import xlsxwriter
import pandas as pd
from pylab import *
from boolean2 import util
import matplotlib.pyplot as plt


# ocasionally randomized nodes
TARGETS = set( "Proliferation".split() )
TARGETS = set( "Differentiation".split() )
TARGETS = set( "Synaptogenesis".split() )
TARGETS = set( "Neurodevelopment".split() )


def new_getvalue( state, name, p):
    """
    Called every time a node value is used in an expression.
    It will override the value for the current step only.
    Returns random values for the node states
    """
    global TARGETS
    value = util.default_get_value( state, name, p )

    if name in TARGETS:
        # pick at random from True, False and original value
        return choice( [True, False, value] )
    else:
        return value

def run( text, nodes, repeat, steps ):
    """
    Runs the simulation and collects the nodes into a collector,
    a convenience class that can average the values that it collects.
    """
    coll = util.Collector()

    for i in xrange( repeat ):
        engine  = Model( mode='async', text=text )
        engine.RULE_GETVALUE = new_getvalue
        # minimalist initial conditions, missing nodes set to false
        engine.initialize( missing=util.false )
        engine.iterate( steps=steps)
        coll.collect( states=engine.states, nodes=nodes )

    print '- completed'
    avgs = coll.get_averages( normalize=True )
    return avgs

if __name__ == '__main__':
```

```
    text = file( 'Simulation1.txt').read() # Input file for Simulation
1   # this collects the state of all nodes
    NODES  = boolean2.all_nodes( text )

    # REPEAT = 1000, STEPS=150
    REPEAT = 1000
    STEPS  = 150

    data = []

    print '- starting simulation with REPEAT=%s, STEPS=%s' % (REPEAT,
STEPS)
    mtext = boolean2.modify_states( text=text, turnon=['CAM', 'GRM5',
'SHANK3', 'SIAH1',  'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2',
'miR137', 'PKC', 'UBE2I', 'DISC1', 'miR335', 'miR4719', 'RUNX2',
'TAL1', 'REST', 'YAP1'] )# Highlighted genes positively regulating
Synaptogenesis
    avgs = run( text=mtext, repeat=REPEAT, nodes=NODES, steps=STEPS)
    data.append( avgs ) # Perturbation condition 2
    mtext = boolean2.modify_states( text=text, turnoff=['CAM', 'GRM5',
'SHANK3', 'SIAH1',  'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2',
'miR137', 'PKC', 'UBE2I', 'DISC1', 'miR335', 'miR4719', 'RUNX2',
'TAL1', 'REST', 'YAP1' ] )
    avgs = run( text=mtext, repeat=REPEAT, nodes=NODES, steps=STEPS)
    data.append( avgs ) # Perturbation condition 3
# Single node perturbation
factorslist = ['miR335', 'miR4719', 'RUNX2', 'TAL1', 'REST', 'YAP1'] #
miRNAs and TFs regulating GRM5 interactome
for i in factorslist:
        mtext = boolean2.modify_states( text=text, turnon=[i] )
        avgs = run( text=mtext, repeat=REPEAT, nodes=NODES,
steps=STEPS)
        data.append( avgs ) # Perturbation condition 1
        mtext = boolean2.modify_states( text=text, turnoff=[i] )
        avgs = run( text=mtext, repeat=REPEAT, nodes=NODES,
steps=STEPS)
        data.append( avgs ) # Perturbation condition 1
        mtext = boolean2.modify_states( text=text, turnon=['CAM',
'GRM5',  'SHANK3', 'SIAH1',  'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2',
'miR137', 'PKC', 'UBE2I', 'DISC1',i] )
        avgs = run( text=mtext, repeat=REPEAT, nodes=NODES,
steps=STEPS)
        data.append( avgs ) # Perturbation condition 4
        mtext = boolean2.modify_states( text=text, turnoff=['CAM',
'GRM5',  'SHANK3', 'SIAH1',  'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2',
'miR137', 'PKC', 'UBE2I', 'DISC1', i] )# Perturbation condition 5
        avgs = run( text=mtext, repeat=REPEAT, nodes=NODES,
steps=STEPS)
        data.append( avgs )
        fname = 'GRM5_synaptogenesis_factors.bin' # Perturbation
results
        util.bsave( data, fname=fname )
        print '- data saved into %s' % fname
        avgs = data
        for  x in data:
            f = lambda x:'Value {}'.format(x+1)
            d = {x: pd.DataFrame(df[x].values.tolist(),
index=df.index).rename(columns=f)
```

```
        for x in df.columns}
        df_1 = pd.concat(d, axis=1)

# Effect of perturbations (PC1 - PC5) on each stage of development at t
= 0 and t =150
df_1.columns.get_loc("Proliferation")
df_1.columns.get_loc("Differentiation")
df_1.columns.get_loc("Synaptogenesis")
df_1.columns.get_loc("Neurodevelopment")

df_1= df_1.iloc[:, [2567, 2717, 604,754, 3322, 3472, 2114, 2264]]
df_1.to_excel('synaptogenesis_factors_sim1.xlsx')
```

Script 4: Script for perturbation of genes regulating overall neurodevelopmental process on each stage of neurodevelopment (Simulation 1, PC1-PC5)

```
"""GRM5_ap simulator (all stages)
It is also a demonstration on how the collector works

"""
import boolean2
from boolean2 import Model, util
from random import choice
import xlsxwriter
import pandas as pd
from pylab import *
from boolean2 import util
import matplotlib.pyplot as plt


# occasionally randomized nodes
TARGETS = set( "Proliferation".split() )
TARGETS = set( "Differentiation".split() )
TARGETS = set( "Synaptogenesis".split() )
TARGETS = set( "Neurodevelopment".split() )


def new_getvalue( state, name, p):
    """
    Called every time a node value is used in an expression.
    It will override the value for the current step only.
    Returns random values for the node states
    """
    global TARGETS
    value = util.default_get_value( state, name, p )

    if name in TARGETS:
        # pick at random from True, False and original value
        return choice( [True, False, value] )
    else:
        return value

def run( text, nodes, repeat, steps ):
    """
    Runs the simulation and collects the nodes into a collector,
    a convenience class that can average the values that it collects.
    """
    coll = util.Collector()

    for i in xrange( repeat ):
        engine  = Model( mode='async', text=text )
        engine.RULE_GETVALUE = new_getvalue
        # minimalist initial conditions, missing nodes set to false
        engine.initialize( missing=util.false )
        engine.iterate( steps=steps)
        coll.collect( states=engine.states, nodes=nodes )

    print '- completed'
    avgs = coll.get_averages( normalize=True )
    return avgs

if __name__ == '__main__':
    text = file( 'Simulation1.txt').read() # Input file for Simulation
1
```

```
    # this collects the state of all nodes
    NODES  = boolean2.all_nodes( text )

    # REPEAT = 1000, STEPS=150    REPEAT = 1000
    STEPS  = 150

    data = []

    print '- starting simulation with REPEAT=%s, STEPS=%s' % (REPEAT,
STEPS)
    mtext = boolean2.modify_states( text=text, turnon=['ERK1', 'ERK2',
'JNK2', 'CCND1','CAM', 'GRM5',  'SHANK3', 'SIAH1',  'GRASP', 'PKD',
'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', 'miR335',
'miR4719', 'RUNX2', 'TAL1', 'REST', 'YAP1'] )
    avgs = run( text=mtext, repeat=REPEAT, nodes=NODES, steps=STEPS)
    data.append( avgs ) # Perturbation condition 2
    mtext = boolean2.modify_states( text=text, turnoff=['ERK1', 'ERK2',
'JNK2', 'CCND1','CAM', 'GRM5',  'SHANK3', 'SIAH1',  'GRASP', 'PKD',
'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', 'miR335',
'miR4719', 'RUNX2', 'TAL1', 'REST', 'YAP1' ] )
    avgs = run( text=mtext, repeat=REPEAT, nodes=NODES, steps=STEPS)
    data.append( avgs ) # Perturbation condition 3
# Single node perturbation
    factorslist = ['miR335', 'miR4719', 'RUNX2', 'TAL1', 'REST',
'YAP1'] # miRNAs and TFs regulating GRM5 interactome)
for i in factorslist:
        mtext = boolean2.modify_states( text=text, turnon=[i] )
        avgs = run( text=mtext, repeat=REPEAT, nodes=NODES,
steps=STEPS)
        data.append( avgs ) # Perturbation condition 1
        mtext = boolean2.modify_states( text=text, turnoff=[i] )
        avgs = run( text=mtext, repeat=REPEAT, nodes=NODES,
steps=STEPS)
        data.append( avgs ) # Perturbation condition 1
        mtext = boolean2.modify_states( text=text, turnon=['ERK1',
'ERK2', 'JNK2', 'CCND1','CAM', 'GRM5',  'SHANK3', 'SIAH1',  'GRASP',
'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1',i]
)
        avgs = run( text=mtext, repeat=REPEAT, nodes=NODES,
steps=STEPS)
        data.append( avgs ) # Perturbation condition 4
        mtext = boolean2.modify_states( text=text, turnoff=['ERK1',
'ERK2', 'JNK2', 'CCND1','CAM', 'GRM5',  'SHANK3', 'SIAH1',  'GRASP',
'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', i]
)
        avgs = run( text=mtext, repeat=REPEAT, nodes=NODES,
steps=STEPS)
        data.append( avgs ) # Perturbation condition 5
        fname = 'GRM5_neurodevelopment_factors.bin'
        util.bsave( data, fname=fname )
        print '- data saved into %s' % fname
        data = util.bload( 'GRM5_neurodevelopment_factors.bin') #
Perturbation results
        avgs = data
        df = pd.DataFrame.from_dict(data)
        df.to_excel("neurodevelopment_factors_sim1_clubbed.xlsx")
        #print (df)
```

```
        avgs = data
        for x in data:
            f = lambda x:'Value {}'.format(x+1)
          d = {x: pd.DataFrame(df[x].values.tolist(),
index=df.index).rename(columns=f)
          for x in df.columns}
          df_1 = pd.concat(d, axis=1)

# Effect of perturbations (PC1- PC5) on each stage of development at t
= 0 and t = 150
df_1.columns.get_loc("Proliferation")
df_1.columns.get_loc("Differentiation")
df_1.columns.get_loc("Synaptogenesis")
df_1.columns.get_loc("Neurodevelopment")

df_1= df_1.iloc[:, [2567, 2717, 604,754, 3322, 3472, 2114, 2264]]
df_1.to_excel('neurodevelopment_factors_sim1.xlsx')
```

Step 3: Analysis of perturbation results

We exported the perturbation results in excel sheet and analyzed the effect of each perturbation condition (PC1-PC5) on stages of neurodevelopment. Effect of each perturbation on each neurodevelopmental stage is calculated as activation frequency. For example, if the neurodevelopmental stage was upregulated (ON) in all the 1000 simulations, the activation frequency is 1. If the neurodevelopmental stage was downregulated (OFF) in all the 1000 simulations, the activation frequency is 0. Out of 1000 simulations, if the neurodevelopmental stage was upregulated 514 times, then the activation frequency is 0.514. We have summarized the perturbation results at t=0 and t=150.

Table 1: Effect of perturbation of genes regulating proliferation on each stage of prenatal neurodevelopment (Simulation 1, PC1-PC5)

Activation frequency of each stage at t = 0 and t = 150 (1000 simulations at each time step)								Perturbed Nodes	Perturbation	Perturbation condition (PC)
Proliferation		Differentiation		Synaptogenesis		Neurodevelopment (all stages)				
Value 0	Value 150	Value 0	Value 150	Value 0	Value 150	Value 0	Value 150			
0	1	0	1	0	0	0	0	GRM5', 'ERK1', 'ERK2', 'JNK2','CCND1','miR335', 'miR4719', 'RUNX2', 'TAL1', 'REST', 'YAP1'	OE	PC2
0	0	0	0	0	0	0	0	GRM5', 'ERK1', 'ERK2', 'JNK2','CCND1','miR335', 'miR4719', 'RUNX2', 'TAL1', 'REST', 'YAP1'	KO	PC3
0	0	0	0	0	0	0	0	'miR335'	OE	PC1
0	0	0	0	0	0	0	0	'miR335'	KO	PC1
0	1	0	1	0	0	0	0	GRM5', 'ERK1', 'ERK2', 'JNK2','CCND1','miR335'	OE	PC4
0	0	0	0	0	0	0	0	GRM5', 'ERK1', 'ERK2', 'JNK2','CCND1','miR335'	KO	PC5
0	0	0	0	0	0	0	0	miR4719'	OE	PC1
0	0	0	0	0	0	0	0	miR4719'	KO	PC1
0	1	0	1	0	0	0	0	GRM5', 'ERK1', 'ERK2', 'JNK2','CCND1','miR4719'	OE	PC4
0	0	0	0	0	0	0	0	GRM5', 'ERK1', 'ERK2', 'JNK2','CCND1','miR4719'	KO	PC5
0	0	0	0	0	0	0	0	'RUNX2'	OE	PC1
0	0	0	0	0	0	0	0	'RUNX2'	KO	PC1
0	1	0	1	0	0	0	0	GRM5', 'ERK1', 'ERK2', 'JNK2','CCND1','RUNX2'	OE	PC4
0	0	0	0	0	0	0	0	GRM5', 'ERK1', 'ERK2', 'JNK2','CCND1','RUNX2'	KO	PC5
0	0	0	0	0	0	0	0	'TAL1'	OE	PC1
0	0	0	0	0	0	0	0	'TAL1'	KO	PC1

0	1	0	1	0	0	0	0	GRM5', 'ERK1', 'ERK2', 'JNK2','CCND1','TAL1'	OE	PC4
0	0	0	0	0	0	0	0	GRM5', 'ERK1', 'ERK2', 'JNK2','CCND1','TAL1'	KO	PC5
0	0	0	0	0	0	0	0	'REST'	OE	PC1
0	0	0	0	0	0	0	0	'REST'	KO	PC1
0	1	0	1	0	0	0	0	GRM5', 'ERK1', 'ERK2', 'JNK2','CCND1','REST'	OE	PC4
0	0	0	0	0	0	0	0	GRM5', 'ERK1', 'ERK2', 'JNK2','CCND1','REST'	KO	PC5
0	0	0	0	0	0	0	0	YAP1'	OE	PC1
0	0	0	0	0	0	0	0	YAP1'	KO	PC1
0	1	0	1	0	0	0	0	GRM5', 'ERK1', 'ERK2', 'JNK2','CCND1','YAP1'	OE	PC4
0	0	0	0	0	0	0	0	GRM5', 'ERK1', 'ERK2', 'JNK2','CCND1','YAP1'	KO	PC5

Table 2: Effect of perturbation of genes regulating differentiation on each stage of prenatal development (Simulation 1, PC1-PC5)

Activation frequency of each stage at *t* = 0 and *t* = 150 (1000 simulations at each time step)								**Perturbed Nodes**	**Perturbation**	**Perturbation condition (PC)**
Proliferation		**Differentiation**		**Synaptogenesis**		**Neurodevelopment (all stages)**				
Value 0	**Value 150**	**Value 0**	**Value 150**	**Value 0**	**Value 150**	**Value 0**	**Value 150**			
0	1	0	1	0	0	0	0	'GRM5', 'ERK1', 'ERK2', 'miR335', 'miR4719', 'RUNX2', 'TAL1', 'REST', 'YAP1'	OE	PC2
0	0	0	0	0	0	0	0	GRM5', 'ERK1', 'ERK2', 'miR335', 'miR4719', 'RUNX2', 'TAL1', 'REST', 'YAP1'	KO	PC3
0	0	0	0	0	0	0	0	'miR335'	OE	PC1
0	0	0	0	0	0	0	0	miR335'	KO	PC1

0	1	0	1	0	0	0	0	GRM5', 'ERK1', 'ERK2', 'miR335', 'miR4719', 'RUNX2', 'TAL1', 'REST', 'YAP1', 'miR335'	OE	PC4
0	0	0	0	0	0	0	0	GRM5', 'ERK1', 'ERK2', 'miR335', 'miR4719', 'RUNX2', 'TAL1', 'REST', 'YAP1', 'miR335'	KO	PC5
0	0	0	0	0	0	0	0	'miR4719'	OE	PC1
0	0	0	0	0	0	0	0	miR4719'	KO	PC1
0	1	0	1	0	0	0	0	GRM5', 'ERK1', 'ERK2', 'miR335'	OE	PC4
0	0	0	0	0	0	0	0	GRM5', 'ERK1', 'ERK2', 'miR335'	KO	PC5
0	0	0	0	0	0	0	0	'RUNX2'	OE	PC1
0	0	0	0	0	0	0	0	'RUNX2'	KO	PC1
0	1	0	1	0	0	0	0	GRM5', 'ERK1', 'ERK2', 'RUNX2'	OE	PC4
0	0	0	0	0	0	0	0	GRM5', 'ERK1', 'ERK2', 'RUNX2'	KO	PC5
0	0	0	0	0	0	0	0	TAL1'	OE	PC1
0	0	0	0	0	0	0	0	'TAL1'	KO	PC1
0	1	0	1	0	0	0	0	GRM5', 'ERK1', 'ERK2', 'TAL1'	OE	PC4
0	0	0	0	0	0	0	0	GRM5', 'ERK1', 'ERK2', 'TAL1'	KO	PC5
0	0	0	0	0	0	0	0	REST'	OE	PC1
0	0	0	0	0	0	0	0	REST'	KO	PC1
0	1	0	1	0	0	0	0	GRM5', 'ERK1', 'ERK2', 'REST'	OE	PC4
0	0	0	0	0	0	0	0	GRM5', 'ERK1', 'ERK2', 'REST'	KO	PC5
0	0	0	0	0	0	0	0	YAP1'	OE	PC1
0	0	0	0	0	0	0	0	'YAP1'	KO	PC1
0	1	0	1	0	0	0	0	GRM5', 'ERK1', 'ERK2', 'YAP1'	OE	PC4
0	0	0	0	0	0	0	0	GRM5', 'ERK1', 'ERK2','YAP1'	KO	PC5

Table 3: Effect of perturbation of genes regulating synaptogenesis on each stage of prenatal neurodevelopment (Simulation 1, PC1-PC5)

Activation frequency of each stage at *t* = 0 and *t* = 150 (1000 simulations at each time step)								Perturbed Nodes	Perturbation	Perturbation condition (PC)
Proliferation		Differentiation		Synaptogenesis		Neurodevelopment (all stages)				
Value 0	Value 150	Value 0	Value 150	Value 0	Value 150	Value 0	Value 150			
0	0	0	0	0	1	0	0	'CAM', 'GRM5', 'SHANK3', 'SIAH1', 'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', 'miR335', 'miR4719', 'RUNX2', 'TAL1', 'REST', 'YAP1'	OE	PC2
0	0	0	0	0	0	0	0	CAM', 'GRM5', 'SHANK3', 'SIAH1', 'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', 'miR335', 'miR4719', 'RUNX2', 'TAL1', 'REST', 'YAP1'	KO	PC3
0	0	0	0	0	0	0	0	miR335'	OE	PC1
0	0	0	0	0	0	0	0	'miR335'	KO	PC1
0	0	0	0	0	1	0	0	CAM', 'GRM5', 'SHANK3', 'SIAH1', 'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', 'miR335'	OE	PC4
0	0	0	0	0	0	0	0	CAM', 'GRM5', 'SHANK3', 'SIAH1', 'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', 'miR335'	KO	PC5
0	0	0	0	0	0	0	0	'miR4719'	OE	PC1
0	0	0	0	0	0	0	0	'miR4719'	KO	PC1
0	0.331	0	0.331	0	1	0	0.337	CAM', 'GRM5', 'SHANK3', 'SIAH1', 'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', 'miR4719'	OE	PC4
0	0	0	0	0	0	0	0	CAM', 'GRM5', 'SHANK3', 'SIAH1', 'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1','miR4719'	KO	PC5

0	0	0	0	0	0	0	0	'RUNX2'	OE	PC1
0	0	0	0	0	0	0	0	'RUNX2'	KO	PC1
0	0	0	0	0	1	0	0	CAM', 'GRM5', 'SHANK3', 'SIAH1', 'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', 'RUNX2'	OE	PC4
0	0	0	0	0	0	0	0	CAM', 'GRM5', 'SHANK3', 'SIAH1', 'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', 'RUNX2'	KO	PC5
0	0	0	0	0	0	0	0	'TAL1'	OE	PC1
0	0	0	0	0	0	0	0	'TAL1'	KO	PC1
0	0	0	0	0	1	0	0	CAM', 'GRM5', 'SHANK3', 'SIAH1', 'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', 'TAL1'	OE	PC4
0	0	0	0	0	0	0	0	CAM', 'GRM5', 'SHANK3', 'SIAH1', 'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', 'TAL1'	KO	PC5
0	0	0	0	0	0	0	0	'REST'	OE	PC1
0	0	0	0	0	0	0	0	REST'	KO	PC1
0	0	0	0	0	1	0	0	CAM', 'GRM5', 'SHANK3', 'SIAH1', 'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', 'REST'	OE	PC4
0	0	0	0	0	0	0	0	CAM', 'GRM5', 'SHANK3', 'SIAH1', 'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', 'REST'	KO	PC5
0	0	0	0	0	0	0	0	'YAP1	OE	PC1
0	0	0	0	0	0	0	0	'YAP1	KO	PC1
0	0	0	0	0	1	0	0	CAM', 'GRM5', 'SHANK3', 'SIAH1', 'GRASP', 'PKD', 'NECAB2', 'LRRC7',	OE	PC4

								'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', 'YAP1'		
0	0	0	0	0	0	0	0	CAM', 'GRM5', 'SHANK3', 'SIAH1', 'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', 'YAP1'	KO	PC5

Table 4: Effect of perturbation of genes regulating overall neurodevelopment process on each stage of prenatal neurodevelopment (Simulation 1, PC1-PC5)

Activation frequency of each stage at *t* = 0 and *t* =150 (1000 simulations at each time step)								**Perturbed Nodes**	**Perturbation**	**Perturbation condition (PC)**
Proliferation		**Differentiation**		**Synaptogenesis**		**Neurodevelopment (all stages)**				
Value 0	**Value 150**	**Value 0**	**Value 150**	**Value 0**	**Value 150**	**Value 0**	**Value 150**			
0	1	0	1	0	1	0	1	'ERK1', 'ERK2', 'JNK2', 'CCND1','CAM', 'GRM5', 'SHANK3', 'SIAH1', 'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', 'miR335', 'miR4719', 'RUNX2', 'TAL1', 'REST', 'YAP1'	OE	PC2
0	0	0	0	0	0	0	0	ERK1', 'ERK2', 'JNK2', 'CCND1','CAM', 'GRM5', 'SHANK3', 'SIAH1', 'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', 'miR335', 'miR4719', 'RUNX2', 'TAL1', 'REST', 'YAP1'	KO	PC3
0	0	0	0	0	0	0	0	'miR335'	OE	PC1
0	0	0	0	0	0	0	0	'miR335'	KO	PC1
0	1	0	1	0	1	0	1	'ERK1', 'ERK2', 'JNK2', 'CCND1','CAM', 'GRM5', 'SHANK3', 'SIAH1', 'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1',	OE	PC4

								'miR335		
0	0	0	0	0	0	0	0	'ERK1', 'ERK2', 'JNK2', 'CCND1','CAM', 'GRM5', 'SHANK3', 'SIAH1', 'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', 'miR335	KO	PC5
0	0	0	0	0	0	0	0	'miR4719'	OE	PC1
0	0	0	0	0	0	0	0	'miR4719'	KO	PC1
0	1	0	1	0	1	0	1	ERK1', 'ERK2', 'JNK2', 'CCND1','CAM', 'GRM5', 'SHANK3', 'SIAH1', 'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', 'miR4719'	OE	PC4
0	0	0	0	0	0	0	0	ERK1', 'ERK2', 'JNK2', 'CCND1','CAM', 'GRM5', 'SHANK3', 'SIAH1', 'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', 'miR4719'	KO	PC5
0	0	0	0	0	0	0	0	'RUNX2'	OE	PC1
0	0	0	0	0	0	0	0	'RUNX2'	KO	PC1
0	1	0	1	0	1	0	1	ERK1', 'ERK2', 'JNK2', 'CCND1','CAM', 'GRM5', 'SHANK3', 'SIAH1', 'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', 'RUNX2'	OE	PC4

0	0	0	0	0	0	0	0	ERK1', 'ERK2', 'JNK2', 'CCND1','CAM', 'GRM5', 'SHANK3', 'SIAH1', 'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', 'RUNX2'	KO	PC5
0	0	0	0	0	0	0	0	'TAL1'	OE	PC1
0	0	0	0	0	0	0	0	'TAL1'	KO	PC1
0	1	0	1	0	1	0	1	ERK1', 'ERK2', 'JNK2', 'CCND1','CAM', 'GRM5', 'SHANK3', 'SIAH1', 'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', 'TAL1'	OE	PC4
0	0	0	0	0	0	0	0	ERK1', 'ERK2', 'JNK2', 'CCND1','CAM', 'GRM5', 'SHANK3', 'SIAH1', 'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', 'TAL1'	KO	PC5
0	0	0	0	0	0	0	0	'REST'	OE	PC1
0	0	0	0	0	0	0	0	REST'	KO	PC1
0	1	0	1	0	1	0	1	ERK1', 'ERK2', 'JNK2', 'CCND1','CAM', 'GRM5', 'SHANK3', 'SIAH1', 'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', 'REST'	OE	PC4
0	0	0	0	0	0	0	0	ERK1', 'ERK2', 'JNK2', 'CCND1','CAM', 'GRM5', 'SHANK3', 'SIAH1', 'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', 'REST'	KO	PC5
0	0	0	0	0	0	0	0	YAP1'	OE	PC1

0	0	0	0	0	0	0	0	'YAP1'	KO	PC1
0	1	0	1	0	1	0	1	ERK1', 'ERK2', 'JNK2', 'CCND1','CAM', 'GRM5', 'SHANK3', 'SIAH1', 'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', 'YAP1'	OE	PC4
0	0	0	0	0	0	0	0	ERK1', 'ERK2', 'JNK2', 'CCND1','CAM', 'GRM5', 'SHANK3', 'SIAH1', 'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', 'YAP1'	KO	PC5

Summary table showing effect of perturbations on each stage of development (Simulation 1, PC1–PC5)

Perturbed stages	**Perturbations**	**Effect of perturbations on each stage at *t* =150**			
		Proliferation	**Differentiation**	**Synaptogenesis**	**Neurodevelopment (all stages)**
Proliferation	PC1	0	0	0	0
	PC2	1	1	0	0
	PC3	0	0	0	0
	PC4	1	1	0	0
	PC5	0	0	0	0
Differentiation	PC1	0	0	0	0
	PC2	1	1	0	0
	PC3	0	0	0	0
	PC4	1	1	0	0
	PC5	0	0	0	0
Synaptogenesis	PC1	0	0	0	0
	PC2	0	0	1	0
	PC3	0	0	0	0
	PC4	0.331 (miR4719 OE)*	0.331 (miR4719 OE)*	1	0.337 (miR4719 OE)*
	PC5	0	0	0	0
Neurodevelopment (all stages)	PC1	0	0	0	0
	PC2	1	1	1	1
	PC3	0	0	0	0
	PC4	1	1	1	1
	PC5	0	0	0	0

* At all perturbations, OE of each miRNA (except miR4719) showed downregulation of Proliferation, Differentiation and overall Neurodevelopment (all stages)

Simulation 2

Step1: Input file for Simulation 2 (TF represses gene and miRNA expression)

```
1: GRM5* = not(miR335 or miR4719 or RUNX2 or TAL1 or REST or YAP1) and
(CAM and SHANK3 and SIAH1 and GRASP and NECAB2 and LRRC7 and DNM2)
1: ERK1* = GRM5
1: ERK2* = not(miR335 or RUNX2) and (GRM5)
1: JNK2* = GRM5
1: MAPK14* = not(GRM5 or miR4719 or TAL1)
1: CCND1* = GRM5
1: RUNX2* = not(miR335)
1: miR335* = not(YAP1 or TAL1)
1: YAP1* = not(miR335)
1: TAL1* = not(miR4719)
1: miR4719* = miR4719
1: CAM* = CAM
1: SHANK3* = SHANK3
1: SIAH1* = SIAH1
1: GRASP* = not(miR335 or REST)
1: PKD* = GRM5
1: NECAB2* = NECAB2
1: LRRC7* = LRRC7
1: DISC1* = LRRC7
1: DNM2* =  DNM2
1: miR137* = GRM5
1: GRIA1* = not(miR137 or miR335 or REST or TAL1 or YAP1)
1: PKC* = GRM5
1: UBE2I* = PKC
1: REST* = not(miR335)
1: Proliferation* = not(MAPK14) and (GRM5 and ERK1 and ERK2 and JNK2
and CCND1)
1: Differentiation* = GRM5 and ERK1 and ERK2
1: Synaptogenesis* =  not(GRIA1) and (CAM and GRM5 and SHANK3 and SIAH1
and GRASP and PKD and NECAB2 and LRRC7 and DNM2 and miR137 and PKC and
UBE2I and DISC1)
1: Neurodevelopment* = not(GRIA1 or MAPK14) and (ERK1 and ERK2 and JNK2
and CCND1 and CAM and GRM5 and SHANK3 and SIAH1 and GRASP and PKD and
NECAB2 and LRRC7 and DNM2 and miR137 and PKC and UBE2I and DISC1)
```

Step 2: Specify the five perturbation conditions (PC1-PC5) for each stage regulated by GRM5 interactome

PC1: OE as well as KO of each miRNA and each TF regulated by GRM5 interactome

PC for the stage proliferation regulated by GRM5 interactome at Simulation 2

PC2: OE of genes positively regulating proliferation (regulated by GRM5 interactome) and OE of miRNAs and TFs regulating GRM5 interactome

PC3: KO of genes positively regulating proliferation (regulated by GRM5 interactome) and KO of miRNAs and TFs regulating GRM5 interactome

PC4: OE of genes positively regulating proliferation (regulated by GRM5 interactome) and OE of each factor (miRNA/ TF) regulating GRM5 interactome

PC5: KO of genes positively regulating proliferation (regulated by GRM5 interactome) and KO of each factor (miRNA/ TF) regulating GRM5 interactome

PC for the stage differentiation regulated by GRM5 interactome at Simulation 2

PC2: OE of genes positively regulating differentiation (regulated by GRM5 interactome) and OE of miRNAs and TFs regulating GRM5 interactome

PC3: KO of genes positively regulating differentiation (regulated by GRM5 interactome) and KO of miRNAs and TFs regulating GRM5 interactome

PC4: OE of genes positively regulating differentiation (regulated by GRM5 interactome) and OE of each factor (miRNA/ TF) regulating GRM5 interactome

PC5: KO of genes positively regulating differentiation (regulated by GRM5 interactome) and KO of each factor (miRNA/ TF) regulating GRM5 interactome

PC for the stage synaptogenesis regulated by GRM5 interactome at Simulation 2

PC2: OE of genes positively regulating synaptogenesis (regulated by GRM5 interactome) and OE of miRNAs and TFs regulating GRM5 interactome

PC3: KO of genes positively regulating synaptogenesis (regulated by GRM5 interactome) and KO of miRNAs and TFs regulating GRM5 interactome

PC4: OE of genes positively regulating synaptogenesis (regulated by GRM5 interactome) and OE of each factor (miRNA/ TF) regulating GRM5 interactome

PC5: KO of genes positively regulating synaptogenesis (regulated by GRM5 interactome) and KO of each factor (miRNA/ TF) regulating GRM5 interactome

PC for overall neurodevelopment process (all stages) regulated by GRM5 interactome at Simulation 2

PC2: OE of genes positively regulating overall neurodevelopment process (regulated by GRM5 interactome) and OE of miRNAs and TFs regulating GRM5 interactome

PC3: KO of genes positively regulating overall neurodevelopment process (regulated by GRM5 interactome) and KO of miRNAs and TFs regulating GRM5 interactome

PC4: OE of genes positively regulating overall neurodevelopment process (regulated by GRM5 interactome) and OE of each factor (miRNA/ TF) regulating GRM5 interactome

PC5: KO of genes positively regulating overall neurodevelopment process (regulated by GRM5 interactome) and KO of each factor (miRNA/ TF) regulating GRM5 interactome

Script 1: Script for perturbation of genes regulating proliferation on each stage of development (Simulation 2, PC1-PC5)

```
"""
GRM5_ap simulator (Proliferation)

It is also a demonstration on how the collector works

"""
import boolean2
from boolean2 import Model, util
from random import choice
import xlsxwriter
import pandas as pd
from pylab import *
from boolean2 import util
import matplotlib.pyplot as plt


# occasionally randomized nodes
TARGETS = set( "Proliferation".split() )
TARGETS = set( "Differentiation".split() )
TARGETS = set( "Synaptogenesis".split() )
TARGETS = set( "Neurodevelopment".split() )

def new_getvalue( state, name, p):
    """
    Called every time a node value is used in an expression.
    It will override the value for the current step only.
    Returns random values for the node states
    """
    global TARGETS
    value = util.default_get_value( state, name, p )

    if name in TARGETS:
        # pick at random from True, False and original value
        return choice( [True, False, value] )
    else:
        return value

def run( text, nodes, repeat, steps ):
    """
    Runs the simulation and collects the nodes into a collector,
    a convenience class that can average the values that it collects.
    """
    coll = util.Collector()

    for i in xrange( repeat ):
        engine  = Model( mode='async', text=text )
        engine.RULE_GETVALUE = new_getvalue
        # minimalist initial conditions, missing nodes set to false
        engine.initialize( missing=util.false )
        engine.iterate( steps=steps)
        coll.collect( states=engine.states, nodes=nodes )

    print '- completed'
    avgs = coll.get_averages( normalize=True )
    return avgs

if __name__ == '__main__':
```

```
    text = file( 'Simulation2.txt').read() # Input file for Simulation
2

    # this collects the state of all nodes
    NODES  = boolean2.all_nodes( text )


    # REPEAT = 1000, STEPS=150

    REPEAT = 1000
    STEPS  = 150

    data = []

    print '- starting simulation with REPEAT=%s, STEPS=%s' % (REPEAT,
STEPS)
    mtext = boolean2.modify_states( text=text, turnon=['GRM5', 'ERK1',
'ERK2', 'JNK2','CCND1','miR335', 'miR4719', 'RUNX2', 'TAL1', 'REST',
'YAP1' ] )
    avgs = run( text=mtext, repeat=REPEAT, nodes=NODES, steps=STEPS)
    data.append( avgs ) # Perturbation condition 2
    mtext = boolean2.modify_states( text=text, turnoff=['GRM5', 'ERK1',
'ERK2', 'JNK2','CCND1','miR335', 'miR4719', 'RUNX2', 'TAL1', 'REST',
'YAP1' ] )
    avgs = run( text=mtext, repeat=REPEAT, nodes=NODES, steps=STEPS)
    data.append( avgs ) # Perturbation condition 3
    # Single node perturbation
    factorslist = ['miR335', 'miR4719', 'RUNX2', 'TAL1', 'REST',
'YAP1'] # miRNAs and TFs regulating GRM5 interactome
    for i in factorslist:
        mtext = boolean2.modify_states( text=text, turnon=[i] )
        avgs = run( text=mtext, repeat=REPEAT, nodes=NODES,
steps=STEPS)
        data.append( avgs ) # Perturbation condition 1
        mtext = boolean2.modify_states( text=text, turnoff=[i] )
        avgs = run( text=mtext, repeat=REPEAT, nodes=NODES,
steps=STEPS)
        data.append( avgs ) # Perturbation condition 1
        mtext = boolean2.modify_states( text=text, turnon=['GRM5',
'ERK1', 'ERK2', 'JNK2','CCND1', i] )
        avgs = run( text=mtext, repeat=REPEAT, nodes=NODES,
steps=STEPS)
        data.append( avgs ) # Perturbation condition 4
        mtext = boolean2.modify_states( text=text, turnoff=['GRM5',
'ERK1', 'ERK2', 'JNK2','CCND1', i] )
        avgs = run( text=mtext, repeat=REPEAT, nodes=NODES,
steps=STEPS)
        data.append( avgs ) # Perturbation condition 5
        fname = 'GRM5_proliferation_factors.bin'
        util.bsave( data, fname=fname )
        print '- data saved into %s' % fname
        data = util.bload( 'GRM5_proliferation_factors.bin')
        avgs = data
        df = pd.DataFrame.from_dict(data)
        df.to_excel("proliferation_factors_sim2_clubbed.xlsx")
        #print (df)

        avgs = data
```

```
        for x in data:
            f = lambda x:'Value {}'.format(x+1)
            d = {x: pd.DataFrame(df[x].values.tolist(),
index=df.index).rename(columns=f)
          for x in df.columns}
          df_1 = pd.concat(d, axis=1)

# Effect of perturbations (PC1-PC5) on each stage of development at t=0
and t =150
df_1.columns.get_loc("Proliferation")
df_1.columns.get_loc("Differentiation")
df_1.columns.get_loc("Synaptogenesis")
df_1.columns.get_loc("Neurodevelopment")

df_1= df_1.iloc[:, [2567, 2717, 604,754, 3322, 3472, 2114, 2264]]
df_1.to_excel('proliferation_factors_sim2.xlsx')
```

Script 6: Script for perturbation of genes regulating differentiation on each stage of development (Simulation 2, PC1-PC5)

```
  """
  GRM5_ap simulator

  It is also a demonstration on how the collector works

  """
  import boolean2
  from boolean2 import Model, util
  from random import choice
  import xlsxwriter
  import pandas as pd
  from pylab import *
  from boolean2 import util
  import matplotlib.pyplot as plt


  # occasionally randomized nodes
  TARGETS = set( "Proliferation".split() )
  TARGETS = set( "Differentiation".split() )
  TARGETS = set( "Synaptogenesis".split() )
  TARGETS = set( "Neurodevelopment".split() )

  def new_getvalue( state, name, p):
      """
      Called every time a node value is used in an expression.
      It will override the value for the current step only.
      Returns random values for the node states
      """
      global TARGETS
      value = util.default_get_value( state, name, p )

      if name in TARGETS:
          # pick at random from True, False and original value
          return choice( [True, False, value] )
      else:
          return value

  def run( text, nodes, repeat, steps ):
      """
      Runs the simulation and collects the nodes into a collector,
      a convenience class that can average the values that it collects.
      """
      coll = util.Collector()

      for i in xrange( repeat ):
          engine  = Model( mode='async', text=text )
          engine.RULE_GETVALUE = new_getvalue
          # minimalist initial conditions, missing nodes set to false
          engine.initialize( missing=util.false )
          engine.iterate( steps=steps)
          coll.collect( states=engine.states, nodes=nodes )

      print '- completed'
      avgs = coll.get_averages( normalize=True )
      return avgs

  if __name__ == '__main__':
```

```
    text = file( 'Simulation2.txt').read() # Input file for Simulation
2

    # this collects the state of all nodes
    NODES  = boolean2.all_nodes( text )

    # REPEAT = 1000, STEPS=150
    #
    REPEAT = 1000
    STEPS  = 150

    data = []

    print '- starting simulation with REPEAT=%s, STEPS=%s' % (REPEAT,
STEPS)
    mtext = boolean2.modify_states( text=text, turnon=['GRM5', 'ERK1',
'ERK2', 'miR335', 'miR4719', 'RUNX2', 'TAL1', 'REST', 'YAP1' ] )
    avgs = run( text=mtext, repeat=REPEAT, nodes=NODES, steps=STEPS)
    data.append( avgs ) # Perturbation condition 2
    mtext = boolean2.modify_states( text=text, turnoff=['GRM5', 'ERK1',
'ERK2', 'miR335', 'miR4719', 'RUNX2', 'TAL1', 'REST', 'YAP1' ] )
    avgs = run( text=mtext, repeat=REPEAT, nodes=NODES, steps=STEPS)
    data.append( avgs ) # Perturbation condition 3
    # Single node perturbation
    factorslist = ['miR335', 'miR4719', 'RUNX2', 'TAL1', 'REST',
'YAP1']# miRNAs and TFs regulating GRM5 interactome
    for i in factorslist:
        mtext = boolean2.modify_states( text=text, turnon=[i] )
        avgs = run( text=mtext, repeat=REPEAT, nodes=NODES,
steps=STEPS)
        data.append( avgs ) # Perturbation condition 1
        mtext = boolean2.modify_states( text=text, turnoff=[i] )
        avgs = run( text=mtext, repeat=REPEAT, nodes=NODES,
steps=STEPS)
        data.append( avgs ) # Perturbation condition 1
        mtext = boolean2.modify_states( text=text, turnon=['GRM5',
'ERK1', 'ERK2',i] )
        avgs = run( text=mtext, repeat=REPEAT, nodes=NODES,
steps=STEPS)
        data.append( avgs ) # Perturbation condition 4
        mtext = boolean2.modify_states( text=text, turnoff=['GRM5',
'ERK1', 'ERK2', i] )
        avgs = run( text=mtext, repeat=REPEAT, nodes=NODES,
steps=STEPS)
        data.append( avgs ) # Perturbation condition 5
        fname = 'GRM5_differentiation_factors.bin'
        util.bsave( data, fname=fname )
        print '- data saved into %s' % fname

        data = util.bload( 'GRM5_differentiation_factors.bin')
        avgs = data
        df = pd.DataFrame.from_dict(data)
        df.to_excel("differentaition_factors_sim2_clubbed.xlsx")
        #print (df)

        avgs = data
        for x in data:
               f = lambda x:'Value {}'.format(x+1)
```

```
          d = {x: pd.DataFrame(df[x].values.tolist(),
index=df.index).rename(columns=f)
          for x in df.columns}
          df_1 = pd.concat(d, axis=1)

# Effect of perturbations (PC1-PC5) on each stage of development at t =
0 and t = 150
df_1.columns.get_loc("Proliferation")
df_1.columns.get_loc("Differentiation")
df_1.columns.get_loc("Synaptogenesis")
df_1.columns.get_loc("Neurodevelopment")


df_1= df_1.iloc[:, [2567, 2717, 604,754, 3322, 3472, 2114, 2264]]
df_1.to_excel('differentiation_factors_sim2.xlsx')
```

Script 7: Script for perturbation of genes regulating synaptogenesis process on each stage of development (Simulation 2, PC1-PC5)

```
"""
GRM5_ap simulator (Synaptogenesis)

It is also a demonstration on how the collector works

"""
import boolean2
from boolean2 import Model, util
from random import choice
import xlsxwriter
import pandas as pd
from pylab import *
from boolean2 import util
import matplotlib.pyplot as plt


# occasionally randomized nodes
TARGETS = set( "Proliferation".split() )
TARGETS = set( "Differentiation".split() )
TARGETS = set( "Synaptogenesis".split() )
TARGETS = set( "Neurodevelopment".split() )


def new_getvalue( state, name, p):
    """
    Called every time a node value is used in an expression.
    It will override the value for the current step only.
    Returns random values for the node states
    """
    global TARGETS
    value = util.default_get_value( state, name, p )

    if name in TARGETS:
        # pick at random from True, False and original value
        return choice( [True, False, value] )
    else:
        return value

def run( text, nodes, repeat, steps ):
    """
    Runs the simulation and collects the nodes into a collector,
    a convenience class that can average the values that it collects.
    """
    coll = util.Collector()

    for i in xrange( repeat ):
        engine  = Model( mode='async', text=text )
        engine.RULE_GETVALUE = new_getvalue
        # minimalist initial conditions, missing nodes set to false
        engine.initialize( missing=util.false )
        engine.iterate( steps=steps)
        coll.collect( states=engine.states, nodes=nodes )

    print '- completed'
```

```
    avgs = coll.get_averages( normalize=True )
    return avgs

if __name__ == '__main__':
    text = file( 'Simulation2.txt').read() # Input file for Simulation 2

    # this collects the state of all nodes
    NODES  = boolean2.all_nodes( text )

    # REPEAT = 1000, STEPS=150
    #
    REPEAT = 1000
    STEPS  = 150

    data = []

    print '- starting simulation with REPEAT=%s, STEPS=%s' % (REPEAT,
STEPS)
    mtext = boolean2.modify_states( text=text, turnon=['CAM', 'GRM5',
'SHANK3', 'SIAH1',  'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137',
'PKC', 'UBE2I', 'DISC1', 'miR335', 'miR4719', 'RUNX2', 'TAL1', 'REST',
'YAP1'] )
    avgs = run( text=mtext, repeat=REPEAT, nodes=NODES, steps=STEPS)
    data.append( avgs ) # Perturbation condition 2
    mtext = boolean2.modify_states( text=text, turnoff=['CAM', 'GRM5',
'SHANK3', 'SIAH1',  'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137',
'PKC', 'UBE2I', 'DISC1', 'miR335', 'miR4719', 'RUNX2', 'TAL1', 'REST',
'YAP1'] )
    avgs = run( text=mtext, repeat=REPEAT, nodes=NODES, steps=STEPS)
    data.append( avgs ) # Perturbation condition 3
    # Single node perturbation
    factorslist = ['miR335', 'miR4719', 'RUNX2', 'TAL1', 'REST', 'YAP1']#
miRNAs and TFs regulating GRM5 interactome
    for i in factorslist:
        mtext = boolean2.modify_states( text=text, turnon=[i] )
        avgs = run( text=mtext, repeat=REPEAT, nodes=NODES, steps=STEPS)
        data.append( avgs ) # Perturbation condition 1
        mtext = boolean2.modify_states( text=text, turnoff=[i] )
        avgs = run( text=mtext, repeat=REPEAT, nodes=NODES, steps=STEPS)
        data.append( avgs ) # Perturbation condition 1
        mtext = boolean2.modify_states( text=text, turnon=['CAM', 'GRM5',
'SHANK3', 'SIAH1',  'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137',
'PKC', 'UBE2I', 'DISC1',i] )
        avgs = run( text=mtext, repeat=REPEAT, nodes=NODES, steps=STEPS)
        data.append( avgs ) # Perturbation condition 4
        mtext = boolean2.modify_states( text=text, turnoff=['CAM', 'GRM5',
'SHANK3', 'SIAH1',  'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137',
'PKC', 'UBE2I', 'DISC1', i] )
        avgs = run( text=mtext, repeat=REPEAT, nodes=NODES, steps=STEPS)
        data.append( avgs ) # Perturbation condition 5
        fname = 'GRM5_synaptogenesis_factors.bin' (Perturbation results)
        util.bsave( data, fname=fname )
        print '- data saved into %s' % fname
        data = util.bload( 'GRM5_synaptogenesis_factors.bin')
```

```
        avgs = data
        df = pd.DataFrame.from_dict(data)
        df.to_excel("synaptogenesis_factors_sim2_clubbed.xlsx")
        #print (df)

        avgs = data
        for x in data:
                f = lambda x:'Value {}'.format(x+1)
          d = {x: pd.DataFrame(df[x].values.tolist(),
index=df.index).rename(columns=f)
          for x in df.columns}
          df_1 = pd.concat(d, axis=1)

# Effect of perturbations (PC1-PC5) on each stage of development at t = 0
and t =150
df_1.columns.get_loc("Proliferation")
df_1.columns.get_loc("Differentiation")
df_1.columns.get_loc("Synaptogenesis")
df_1.columns.get_loc("Neurodevelopment")

df_1= df_1.iloc[:, [2567, 2717, 604,754, 3322, 3472, 2114, 2264]]
df_1.to_excel('synaptogenesis_factors_sim2.xlsx')
```

Script 8: Script for perturbation of genes regulating overall neurodevelopment process on each stage of development (Simulation 2, PC1-PC5)

```
  """
  GRM5_ap simulator (Neurodevelopment)

  It is also a demonstration on how the collector works

  """
  import boolean2
  from boolean2 import Model, util
  from random import choice
  import xlsxwriter
  import pandas as pd
  from pylab import *
  from boolean2 import util
  import matplotlib.pyplot as plt


  # occasionally randomized nodes
  TARGETS = set( "Proliferation".split() )
  TARGETS = set( "Differentiation".split() )
  TARGETS = set( "Synaptogenesis".split() )
  TARGETS = set( "Neurodevelopment".split() )


  def new_getvalue( state, name, p):
      """
      Called every time a node value is used in an expression.
      It will override the value for the current step only.
      Returns random values for the node states
      """
      global TARGETS
      value = util.default_get_value( state, name, p )

      if name in TARGETS:
          # pick at random from True, False and original value
          return choice( [True, False, value] )
      else:
          return value

  def run( text, nodes, repeat, steps ):
      """
      Runs the simulation and collects the nodes into a collector,
      a convenience class that can average the values that it collects.
      """
      coll = util.Collector()

      for i in xrange( repeat ):
          engine  = Model( mode='async', text=text )
          engine.RULE_GETVALUE = new_getvalue
          # minimalist initial conditions, missing nodes set to false
          engine.initialize( missing=util.false )
          engine.iterate( steps=steps)
          coll.collect( states=engine.states, nodes=nodes )

      print '- completed'
      avgs = coll.get_averages( normalize=True )
      return avgs

  if __name__ == '__main__':
```

```
    text = file( 'Simulation2.txt').read() # Input file for Simulation2

    # this collects the state of all nodes
    NODES  = boolean2.all_nodes( text )

    # REPEAT = 1000, STEPS=150
    REPEAT = 1000
    STEPS  = 150

    data = []

    print '- starting simulation with REPEAT=%s, STEPS=%s' % (REPEAT,
STEPS)
    mtext = boolean2.modify_states( text=text, turnon=['ERK1', 'ERK2',
'JNK2', 'CCND1','CAM', 'GRM5',  'SHANK3', 'SIAH1',  'GRASP', 'PKD',
'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', 'miR335',
'miR4719', 'RUNX2', 'TAL1', 'REST', 'YAP1'] )
    avgs = run( text=mtext, repeat=REPEAT, nodes=NODES, steps=STEPS)
    data.append( avgs ) # Perturbation condition 2
    mtext = boolean2.modify_states( text=text, turnoff=['ERK1', 'ERK2',
'JNK2', 'CCND1','CAM', 'GRM5',  'SHANK3', 'SIAH1',  'GRASP', 'PKD',
'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', 'miR335',
'miR4719', 'RUNX2', 'TAL1', 'REST', 'YAP1'] )
    avgs = run( text=mtext, repeat=REPEAT, nodes=NODES, steps=STEPS)
    data.append( avgs ) # Perturbation condition 3
    # Single node perturbation
    factorslist = ['miR335', 'miR4719', 'RUNX2', 'TAL1', 'REST',
'YAP1'] # miRNAs and TFs regulating GRM5 interactome
    for i in factorslist:
        mtext = boolean2.modify_states( text=text, turnon=[i] )
        avgs = run( text=mtext, repeat=REPEAT, nodes=NODES,
steps=STEPS)
        data.append( avgs ) # Perturbation condition 1
        mtext = boolean2.modify_states( text=text, turnoff=[i] )
        avgs = run( text=mtext, repeat=REPEAT, nodes=NODES,
steps=STEPS)
        data.append( avgs ) # Perturbation condition 1
        mtext = boolean2.modify_states( text=text, turnon=['ERK1',
'ERK2', 'JNK2', 'CCND1','CAM', 'GRM5',  'SHANK3', 'SIAH1',  'GRASP',
'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1',i]
)
        avgs = run( text=mtext, repeat=REPEAT, nodes=NODES,
steps=STEPS)
        data.append( avgs ) # Perturbation condition 4
        mtext = boolean2.modify_states( text=text, turnoff=['ERK1',
'ERK2', 'JNK2', 'CCND1','CAM', 'GRM5',  'SHANK3', 'SIAH1',  'GRASP',
'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', i]
)
        avgs = run( text=mtext, repeat=REPEAT, nodes=NODES,
steps=STEPS)
        data.append( avgs ) # Perturbation condition 5
        fname = 'GRM5_neurodevelopment_factors.bin'
        util.bsave( data, fname=fname )
        print '- data saved into %s' % fname

        data = util.bload( 'GRM5_neurodevelopment_factors.bin')
        avgs = data
        df = pd.DataFrame.from_dict(data)
```

```
        df.to_excel("neurodevelopment_factors_sim2_clubbed.xlsx")
        #print (df)

        avgs = data
        for x in data:
               f = lambda x:'Value {}'.format(x+1)
          d = {x: pd.DataFrame(df[x].values.tolist(),
index=df.index).rename(columns=f)
          for x in df.columns}
          df_1 = pd.concat(d, axis=1)

# Effect of perturbations (PC1-PC5) on each stage of development at t =
0 and t = 150
df_1.columns.get_loc("Proliferation")
df_1.columns.get_loc("Differentiation")
df_1.columns.get_loc("Synaptogenesis")
df_1.columns.get_loc("Neurodevelopment")

df_1= df_1.iloc[:, [2567, 2717, 604,754, 3322, 3472, 2114, 2264]]
df_1.to_excel('neurodevelopment_factors_sim2.xlsx')
```

Step 3: Analysis of perturbation results

We exported the perturbation results in excel sheet and analyzed the effect of each perturbation condition (PC1-PC5) on stages of neurodevelopment. Effect of each perturbation on each neurodevelopmental stage is calculated as activation frequency. For example, if the neurodevelopmental stage was upregulated (ON) in all the 1000 simulations, the activation frequency is 1. If the neurodevelopmental stage was downregulated (OFF) in all the 1000 simulations, the activation frequency is 0. Out of 1000 simulations, if the neurodevelopmental stage was upregulated 514 times, then the activation frequency is 0.514. We have summarized the perturbation results at t=0 and t=150.

Table 5: Effect of perturbation of genes regulating proliferation on stages of prenatal neurodevelopment (Simulation 2, PC1-PC5)

Activation frequency of each stage at *t* = 0 and *t* =150 (1000 simulations at each time step)								Nodes perturbed	Perturbation	Perturbation condition (PC)
Proliferation		Differentiation		Synaptogenesis		Neurodevelopment (all stages)				
Value 0	Value 150	Value 0	Value 150	Value 0	Value 150	Value 0	Value 150			
0	1	0	1	0	0	0	0	GRM5', 'ERK1', 'ERK2', JNK2','CCND1','miR335', 'miR4719', 'RUNX2', 'TAL1', 'REST', 'YAP1'	OE	PC2
0	0	0	0	0	0	0	0	GRM5', 'ERK1', 'ERK2', 'JNK2','CCND1','miR335', 'miR4719', 'RUNX2', 'TAL1', 'REST', 'YAP1'	KO	PC3
0	0	0	0	0	0	0	0	'miR335'	OE	PC1
0	0	0	0	0	0	0	0	'miR335'	KO	PC1
0	1	0	1	0	0	0	0	GRM5', 'ERK1', 'ERK2', 'JNK2','CCND1','miR335'	OE	PC4
0	0	0	0	0	0	0	0	GRM5', 'ERK1', 'ERK2', 'JNK2','CCND1','miR335'	KO	PC5
0	0	0	0	0	0	0	0	miR4719'	OE	PC1
0	0	0	0	0	0	0	0	miR4719'	KO	PC1
0	1	0	1	0	0	0	0	GRM5', 'ERK1', 'ERK2', 'JNK2','CCND1','miR4719'	OE	PC4
0	0	0	0	0	0	0	0	GRM5', 'ERK1', 'ERK2', 'JNK2','CCND1','miR4719'	KO	PC5
0	0	0	0	0	0	0	0	'RUNX2'	OE	PC1
0	0	0	0	0	0	0	0	'RUNX2'	KO	PC1
0	1	0	1	0	0	0	0	GRM5', 'ERK1', 'ERK2', 'JNK2','CCND1','RUNX2'	OE	PC4
0	0	0	0	0	0	0	0	GRM5', 'ERK1', 'ERK2', 'JNK2','CCND1','RUNX2'	KO	PC5
0	0	0	0	0	0	0	0	'TAL1'	OE	PC1
0	0	0	0	0	0	0	0	'TAL1'	KO	PC1
0	1	0	1	0	0	0	0	GRM5', 'ERK1', 'ERK2', 'JNK2','CCND1','TAL1'	OE	PC4

0	0	0	0	0	0	0	0	GRM5', 'ERK1', 'ERK2', 'JNK2','CCND1','TAL1'	KO	PC5
0	0	0	0	0	0	0	0	'REST'	OE	PC1
0	0	0	0	0	0	0	0	'REST'	KO	PC1
0	1	0	1	0	0	0	0	GRM5', 'ERK1', 'ERK2', 'JNK2','CCND1','REST'	OE	PC4
0	0	0	0	0	0	0	0	GRM5', 'ERK1', 'ERK2', 'JNK2','CCND1','REST'	KO	PC5
0	0	0	0	0	0	0	0	YAP1'	OE	PC1
0	0	0	0	0	0	0	0	YAP1'	KO	PC1
0	1	0	1	0	0	0	0	GRM5', 'ERK1', 'ERK2', 'JNK2','CCND1','YAP1'	OE	PC4
0	0	0	0	0	0	0	0	GRM5', 'ERK1', 'ERK2', 'JNK2','CCND1','YAP1'	KO	PC5

Table 6: Effect of perturbation of genes regulating differentiation on stages of prenatal development (Simulation 2, PC1–PC5)

Activation frequency of each stage at *t* = 0 and *t* = 150 (1000 simulations at each time step)								**Nodes perturbed**	**Perturbation**	**Perturbation condition (PC)**
Proliferation		**Differentiation**		**Synaptogenesis**		**Neurodevelopment (all stages)**				
Value 0	**Value 150**	**Value 0**	**Value 150**	**Value 0**	**Value 150**	**Value 0**	**Value 150**			
0	1	0	1	0	0	0	0	'GRM5', 'ERK1', 'ERK2', 'miR335', 'miR4719', 'RUNX2', 'TAL1', 'REST', 'YAP1'	OE	PC2
0	0	0	0	0	0	0	0	GRM5', 'ERK1', 'ERK2', 'miR335', 'miR4719', 'RUNX2', 'TAL1', 'REST', 'YAP1'	KO	PC3
0	0	0	0	0	0	0	0	'miR335'	OE	PC1
0	0	0	0	0	0	0	0	miR335'	KO	PC1
0	1	0	1	0	0	0	0	GRM5', 'ERK1', 'ERK2', 'miR335', 'miR4719', 'RUNX2', 'TAL1', 'REST', 'YAP1', 'miR335'	OE	PC4
0	0	0	0	0	0	0	0	GRM5', 'ERK1', 'ERK2', 'miR335', 'miR4719', 'RUNX2', 'TAL1', 'REST', 'YAP1', 'miR335'	KO	PC5
0	0	0	0	0	0	0	0	'miR4719'	OE	PC1
0	0	0	0	0	0	0	0	miR4719'	KO	PC1

0	1	0	1	0	0	0	0	GRM5', 'ERK1', 'ERK2', 'miR335'	OE	PC4
0	0	0	0	0	0	0	0	GRM5', 'ERK1', 'ERK2', 'miR335'	KO	PC5
0	0	0	0	0	0	0	0	'RUNX2'	OE	PC1
0	0	0	0	0	0	0	0	'RUNX2'	KO	PC1
0	1	0	1	0	0	0	0	GRM5', 'ERK1', 'ERK2', 'RUNX2'	OE	PC4
0	0	0	0	0	0	0	0	GRM5', 'ERK1', 'ERK2', 'RUNX2'	KO	PC5
0	0	0	0	0	0	0	0	TAL1'	OE	PC1
0	0	0	0	0	0	0	0	'TAL1'	KO	PC1
0	1	0	1	0	0	0	0	GRM5', 'ERK1', 'ERK2', 'TAL1'	OE	PC4
0	0	0	0	0	0	0	0	GRM5', 'ERK1', 'ERK2', 'TAL1'	KO	PC5
0	0	0	0	0	0	0	0	REST'	OE	PC1
0	0	0	0	0	0	0	0	REST'	KO	PC1
0	1	0	1	0	0	0	0	GRM5', 'ERK1', 'ERK2', 'REST'	OE	PC4
0	0	0	0	0	0	0	0	GRM5', 'ERK1', 'ERK2', 'REST'	KO	PC5
0	0	0	0	0	0	0	0	YAP1'	OE	PC1
0	0	0	0	0	0	0	0	'YAP1'	KO	PC1
0	1	0	1	0	0	0	0	GRM5', 'ERK1', 'ERK2', 'YAP1'	OE	PC4
0	0	0	0	0	0	0	0	GRM5', 'ERK1', 'ERK2','YAP1'	KO	PC5

Table 7: Effect of perturbation of genes regulating synaptogenesis on stages of prenatal development (Simulation 2, PC1-PC5)

Activation frequency of each stage at *t* = 0 and *t* = 150 (1000 simulations at each time step)								**Nodes perturbed**	**Perturbation**	**Perturbation condition (PC)**
Proliferation		**Differentiation**		**Synaptogenesis**		**Neurodevelopment (all stages)**				
Value 0	**Value 150**	**Value 0**	**Value 150**	**Value 0**	**Value 150**	**Value 0**	**Value 150**			
0	0	0	0	0	1	0	0	'CAM', 'GRM5', 'SHANK3', 'SIAH1', 'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', 'miR335', 'miR4719', 'RUNX2', 'TAL1', 'REST', 'YAP1'	OE	PC2
0	0	0	0	0	0	0	0	CAM', 'GRM5', 'SHANK3', 'SIAH1', 'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', 'miR335', 'miR4719', 'RUNX2', 'TAL1', 'REST', 'YAP1'	KO	PC 3

0	0	0	0	0	0	0	0	miR335'	OE	PC1
0	0	0	0	0	0	0	0	'miR335'	KO	PC1
0	0	0	0	0	1	0	0	CAM', 'GRM5', 'SHANK3', 'SIAH1', 'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', 'miR335'	OE	PC4
0	0	0	0	0	0	0	0	CAM', 'GRM5', 'SHANK3', 'SIAH1', 'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', 'miR335'	KO	PC5
0	0	0	0	0	0	0	0	'miR4719'	OE	PC1
0	0	0	0	0	0	0	0	'miR4719'	KO	PC1
0	0	0	0	0	1	0	0	CAM', 'GRM5', 'SHANK3', 'SIAH1', 'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', 'miR4719'	OE	PC4
0	0	0	0	0	0	0	0	CAM', 'GRM5', 'SHANK3', 'SIAH1', 'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1','miR4719'	KO	PC5
0	0	0	0	0	0	0	0	'RUNX2'	OE	PC1
0	0	0	0	0	0	0	0	'RUNX2'	KO	PC1
0	0	0	0	0	1	0	0	CAM', 'GRM5', 'SHANK3', 'SIAH1', 'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', 'RUNX2'	OE	PC4
0	0	0	0	0	0	0	0	CAM', 'GRM5', 'SHANK3', 'SIAH1', 'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', 'RUNX2'	KO	PC5
0	0	0	0	0	0	0	0	'TAL1'	OE	PC1
0	0	0	0	0	0	0	0	'TAL1'	KO	PC1
0	0	0	0	0	1	0	0	CAM', 'GRM5', 'SHANK3', 'SIAH1', 'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', 'TAL1'	OE	PC4
0	0	0	0	0	0	0	0	CAM', 'GRM5', 'SHANK3', 'SIAH1', 'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', 'TAL1'	KO	PC5
0	0	0	0	0	0	0	0	'REST'	OE	PC1
0	0	0	0	0	0	0	0	REST'	KO	PC1

0	0	0	0	0	1	0	0	CAM', 'GRM5', 'SHANK3', 'SIAH1', 'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', 'REST'	OE	PC4
0	0	0	0	0	0	0	0	CAM', 'GRM5', 'SHANK3', 'SIAH1', 'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', 'REST'	KO	PC5
0	0	0	0	0	0	0	0	'YAP1	OE	PC1
0	0	0	0	0	0	0	0	'YAP1	KO	PC1
0	0	0	0	0	1	0	0	CAM', 'GRM5', 'SHANK3', 'SIAH1', 'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', 'YAP1'	OE	PC4
0	0	0	0	0	0	0	0	CAM', 'GRM5', 'SHANK3', 'SIAH1', 'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', 'YAP1'	KO	PC5

Table 8: Effect of perturbation of genes regulating overall neurodevelopmental process on each stage of prenatal neurodevelopment (Simulation 2, PC1–PC5)

Activation frequency of each stage at *t* = 0 and *t* = 150 (1000 simulations at each time step)								**Nodes perturbed**	**Perturbation**	**Perturbation condition (PC)**
Proliferation		**Differentiation**		**Synaptogenesis**		**Neurodevelopment (all stages)**				
Value 0	**Value 150**	**Value 0**	**Value 150**	**Value 0**	**Value 150**	**Value 0**	**Value 150**			
0	1	0	1	0	1	0	1	'ERK1', 'ERK2', 'JNK2', 'CCND1','CAM', 'GRM5', 'SHANK3', 'SIAH1', 'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', 'miR335', 'miR4719', 'RUNX2', 'TAL1', 'REST', 'YAP1'	OE	PC2
0	0	0	0	0	0	0	0	ERK1', 'ERK2', 'JNK2', 'CCND1','CAM', 'GRM5', 'SHANK3', 'SIAH1', 'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', 'miR335', 'miR4719', 'RUNX2', 'TAL1', 'REST', 'YAP1'	KO	PC3
0	0	0	0	0	0	0	0	'miR335'	OE	PC1
0	0	0	0	0	0	0	0	'miR335'	KO	PC1

0	1	0	1	0	1	0	1	'ERK1', 'ERK2', 'JNK2', 'CCND1','CAM', 'GRM5', 'SHANK3', 'SIAH1', 'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', 'miR335	OE	PC4
0	0	0	0	0	0	0	0	'ERK1', 'ERK2', 'JNK2', 'CCND1','CAM', 'GRM5', 'SHANK3', 'SIAH1', 'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', 'miR335	KO	PC5
0	0	0	0	0	0	0	0	'miR4719'	OE	PC1
0	0	0	0	0	0	0	0	'miR4719'	KO	PC1
0	1	0	1	0	1	0	1	ERK1', 'ERK2', 'JNK2', 'CCND1','CAM', 'GRM5', 'SHANK3', 'SIAH1', 'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', 'miR4719'	OE	PC4
0	0	0	0	0	0	0	0	ERK1', 'ERK2', 'JNK2', 'CCND1','CAM', 'GRM5', 'SHANK3', 'SIAH1', 'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', 'miR4719'	KO	PC5
0	0	0	0	0	0	0	0	'RUNX2'	OE	PC1
0	0	0	0	0	0	0	0	'RUNX2'	KO	PC1
0	1	0	1	0	1	0	1	ERK1', 'ERK2', 'JNK2', 'CCND1','CAM', 'GRM5', 'SHANK3', 'SIAH1', 'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', 'RUNX2'	OE	PC4
0	0	0	0	0	0	0	0	ERK1', 'ERK2', 'JNK2', 'CCND1','CAM', 'GRM5', 'SHANK3', 'SIAH1', 'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', 'RUNX2'	KO	PC5
0	0	0	0	0	0	0	0	'TAL1'	OE	PC1
0	0	0	0	0	0	0	0	'TAL1'	KO	PC1
0	1	0	1	0	1	0	1	ERK1', 'ERK2', 'JNK2', 'CCND1','CAM', 'GRM5', 'SHANK3', 'SIAH1', 'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', 'TAL1'	OE	PC4

0	0	0	0	0	0	0	0	ERK1', 'ERK2', 'JNK2', 'CCND1','CAM', 'GRM5', 'SHANK3', 'SIAH1', 'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', 'TAL1'	KO	PC5
0	0	0	0	0	0	0	0	'REST'	OE	PC1
0	0	0	0	0	0	0	0	REST'	KO	PC1
0	1	0	1	0	1	0	1	ERK1', 'ERK2', 'JNK2', 'CCND1','CAM', 'GRM5', 'SHANK3', 'SIAH1', 'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', 'REST'	OE	PC4
0	0	0	0	0	0	0	0	ERK1', 'ERK2', 'JNK2', 'CCND1','CAM', 'GRM5', 'SHANK3', 'SIAH1', 'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', 'REST'	KO	PC5
0	0	0	0	0	0	0	0	YAP1'	OE	PC1
0	0	0	0	0	0	0	0	'YAP1'	KO	PC1
0	1	0	1	0	1	0	1	ERK1', 'ERK2', 'JNK2', 'CCND1','CAM', 'GRM5', 'SHANK3', 'SIAH1', 'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', 'YAP1'	OE	PC4
0	0	0	0	0	0	0	0	ERK1', 'ERK2', 'JNK2', 'CCND1','CAM', 'GRM5', 'SHANK3', 'SIAH1', 'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', 'YAP1'	KO	PC5

Summary table showing effect of perturbations on each stage of neurodevelopment (Simulation 2, PC1-PC5)

Perturbed stages	**Perturbations**	**Effect of perturbations on each stage at *t* =150**			
		Proliferation	**Differentiation**	**Synaptogenesis**	**Neurodevelopment (all stages)**
Proliferation	PC1	0	0	0	0
	PC2	1	1	0	0
	PC3	0	0	0	0
	PC4	1	1	0	0
	PC5	0	0	0	0
Differentiation	PC1	0	0	0	0
	PC2	1	1	0	0

	PC3	0	0	0	0
	PC4	1	1	0	0
	PC5	0	0	0	0
Synaptogenesis	PC1	0	0	0	0
	PC2	0	0	1	0
	PC3	0	0	0	0
	PC4	0	0	1	0
	PC5	0	0	0	0
Neurodevelopment (all stages)	PC1	0	0	0	0
	PC2	1	1	1	1
	PC3	0	0	0	0
	PC4	1	1	1	1
	PC5	0	0	0	0

Simulation 3

Step1: Input file for Simulation 3 (TF represses miRNA and activates gene expression)

```
1: GRM5* = not(miR335 or miR4719) and (RUNX2 or TAL1 or REST or YAP1)
and (CAM and SHANK3 and SIAH1 and GRASP and NECAB2 and LRRC7 and DNM2)
1: ERK1* = GRM5
1: ERK2* = not(miR335) and (RUNX2) and (GRM5)
1: JNK2* = GRM5
1: MAPK14* = not(GRM5 or miR4719) and (TAL1)
1: CCND1* = GRM5
1: RUNX2* = not(miR335)
1: miR335* = not(YAP1 or TAL1)
1: YAP1* = not(miR335)
1: TAL1* = not(miR4719)
1: miR4719* = miR4719
1: CAM* = CAM
1: SHANK3* = SHANK3
1: SIAH1* = SIAH1
1: GRASP* = not(miR335) and (REST)
1: PKD* = GRM5
1: NECAB2* = NECAB2
1: LRRC7* = LRRC7
1: DISC1* = LRRC7
1: DNM2* =  DNM2
1: miR137* = GRM5
1: GRIA1* = not(miR137 or miR335) and (REST or TAL1 or YAP1)
1: PKC* = GRM5
1: UBE2I* = PKC
1: REST* = not(miR335)
1: Proliferation* = not(MAPK14) and (GRM5 and ERK1 and ERK2 and JNK2
and CCND1)
1: Differentiation* = GRM5 and ERK1 and ERK2
1: Synaptogenesis* =  not(GRIA1) and (CAM and GRM5 and SHANK3 and SIAH1
and GRASP and PKD and NECAB2 and LRRC7 and DNM2 and miR137 and PKC and
UBE2I and DISC1)
1: Neurodevelopment* = not(GRIA1 or MAPK14) and (ERK1 and ERK2 and JNK2
and CCND1 and CAM and GRM5 and SHANK3 and SIAH1 and GRASP and PKD and
NECAB2 and LRRC7 and DNM2 and miR137 and PKC and UBE2I and DISC1)
```

Step 2: Specify the five perturbation conditions (PC1-PC5) for each stage regulated by

GRM5 interactome

PC1: OE as well as KO of each miRNA and each TF regulated by GRM5 interactome (remains same for each stage)

PC for the stage proliferation regulated by GRM5 interactome at Simulation 3

PC2: OE of genes positively regulating proliferation (regulated by GRM5 interactome) and OE of miRNAs and TFs regulating GRM5 interactome

PC3: KO of genes positively regulating proliferation (regulated by GRM5 interactome) and KO of miRNAs and TFs regulating GRM5 interactome

PC4: OE of genes positively regulating proliferation (regulated by GRM5 interactome) and OE of each factor (miRNA/ TF) regulating GRM5 interactome

PC5: KO of genes positively regulating proliferation (regulated by GRM5 interactome) and KO of each factor (miRNA/ TF) regulating GRM5 interactome

PC for the stage differentiation regulated by GRM5 interactome at Simulation 3

PC2: OE of genes positively regulating differentiation (regulated by GRM5 interactome) and OE of miRNAs and TFs regulating GRM5 interactome

PC3: KO of genes positively regulating differentiation (regulated by GRM5 interactome) and KO of miRNAs and TFs regulating GRM5 interactome

PC4: OE of genes positively regulating differentiation (regulated by GRM5 interactome) and OE of each factor (miRNA/ TF) regulating GRM5 interactome

PC5: KO of genes positively regulating differentiation (regulated by GRM5 interactome) and KO of each factor (miRNA/ TF) regulating GRM5 interactome

PC for the stage synaptogenesis regulated by GRM5 interactome at Simulation 3

PC2: OE of genes positively regulating synaptogenesis (regulated by GRM5 interactome) and OE of miRNAs and TFs regulating GRM5 interactome

PC3: KO of genes positively regulating synaptogenesis (regulated by GRM5 interactome) and KO of miRNAs and TFs regulating GRM5 interactome

PC4: OE of genes positively regulating synaptogenesis (regulated by GRM5 interactome) and OE of each factor (miRNA/ TF) regulating GRM5 interactome

PC5: KO of genes positively regulating synaptogenesis (regulated by GRM5 interactome) and KO of each factor (miRNA/ TF) regulating GRM5 interactome

PC for overall neurodevelopment process (all stages) regulated by GRM5 interactome at Simulation 3

PC2: OE of genes positively regulating overall neurodevelopment process (regulated by GRM5 interactome) and OE of miRNAs and TFs regulating GRM5 interactome

PC3: KO of genes positively regulating overall neurodevelopment process (regulated by GRM5 interactome) and KO of miRNAs and TFs regulating GRM5 interactome

PC4: OE of genes positively regulating overall neurodevelopment process (regulated by GRM5 interactome) and OE of each factor (miRNA/ TF) regulating GRM5 interactome

PC5: KO of genes positively regulating overall neurodevelopment process (regulated by GRM5 interactome) and KO of each factor (miRNA/ TF) regulating GRM5 interactome

Script 9: Script for perturbation of genes regulating proliferation on each stage of development (Simulation 3, PC1-PC5)

```
"""
GRM5_ap simulator (Proliferation)

It is also a demonstration on how the collector works

"""
import boolean2
from boolean2 import Model, util
from random import choice
import xlsxwriter
import pandas as pd
from pylab import *
from boolean2 import util
import matplotlib.pyplot as plt


# occasionally randomized nodes
TARGETS = set( "Proliferation".split() )
TARGETS = set( "Differentiation".split() )
TARGETS = set( "Synaptogenesis".split() )
TARGETS = set( "Neurodevelopment".split() )

def new_getvalue( state, name, p):
    """
    Called every time a node value is used in an expression.
    It will override the value for the current step only.
    Returns random values for the node states
    """
    global TARGETS
    value = util.default_get_value( state, name, p )

    if name in TARGETS:
        # pick at random from True, False and original value
        return choice( [True, False, value] )
    else:
        return value

def run( text, nodes, repeat, steps ):
    """
    Runs the simulation and collects the nodes into a collector,
    a convenience class that can average the values that it collects.
    """
    coll = util.Collector()

    for i in xrange( repeat ):
        engine  = Model( mode='async', text=text )
        engine.RULE_GETVALUE = new_getvalue
        # minimalist initial conditions, missing nodes set to false
        engine.initialize( missing=util.false )
        engine.iterate( steps=steps)
        coll.collect( states=engine.states, nodes=nodes )

    print '- completed'
    avgs = coll.get_averages( normalize=True )
    return avgs

if __name__ == '__main__':
```

```
    text = file( 'Simulation3.txt').read() # Input file for Simulation
3

    # this collects the state of all nodes
    NODES  = boolean2.all_nodes( text )


    # REPEAT = 1000, STEPS=150

    REPEAT = 1000
    STEPS  = 150

    data = []

    print '- starting simulation with REPEAT=%s, STEPS=%s' % (REPEAT,
STEPS)
    mtext = boolean2.modify_states( text=text, turnon=['GRM5', 'ERK1',
'ERK2', 'JNK2','CCND1','miR335', 'miR4719', 'RUNX2', 'TAL1', 'REST',
'YAP1'] )
    avgs = run( text=mtext, repeat=REPEAT, nodes=NODES, steps=STEPS)
    data.append( avgs ) # Perturbation condition 2
    mtext = boolean2.modify_states( text=text, turnoff=['GRM5', 'ERK1'
'ERK2', 'JNK2','CCND1','miR335', 'miR4719', 'RUNX2', 'TAL1', 'REST',
'YAP1'] )
    avgs = run( text=mtext, repeat=REPEAT, nodes=NODES, steps=STEPS)
    data.append( avgs ) # Perturbation condition 3
   # Single node perturbation
    factorslist = ['miR335', 'miR4719', 'RUNX2', 'TAL1', 'REST',
'YAP1']
    for i in factorslist:
        mtext = boolean2.modify_states( text=text, turnon=[i] )
        avgs = run( text=mtext, repeat=REPEAT, nodes=NODES,
steps=STEPS)
        data.append( avgs ) # Perturbation condition 1
        mtext = boolean2.modify_states( text=text, turnoff=[i] )
        avgs = run( text=mtext, repeat=REPEAT, nodes=NODES,
steps=STEPS)
        data.append( avgs ) # Perturbation condition 1
        mtext = boolean2.modify_states( text=text, turnon=['GRM5',
'ERK1', 'ERK2', 'JNK2','CCND1', i] )
        avgs = run( text=mtext, repeat=REPEAT, nodes=NODES,
steps=STEPS)
        data.append( avgs ) # Perturbation condition 4
        mtext = boolean2.modify_states( text=text, turnoff=['GRM5',
'ERK1', 'ERK2', 'JNK2','CCND1', i] )
        avgs = run( text=mtext, repeat=REPEAT, nodes=NODES,
steps=STEPS)
        data.append( avgs ) # Perturbation condition 5
        fname = 'GRM5_proliferation_factors.bin'
        util.bsave( data, fname=fname )
        print '- data saved into %s' % fname
        data = util.bload( 'GRM5_proliferation_factors.bin')
        avgs = data
        df = pd.DataFrame.from_dict(data)
        df.to_excel("proliferation_factors_sim3_clubbed.xlsx")
        #print (df)

        avgs = data
```

```
        for x in data:
                f = lambda x:'Value {}'.format(x+1)
          d = {x: pd.DataFrame(df[x].values.tolist(),
index=df.index).rename(columns=f)
          for x in df.columns}
          df_1 = pd.concat(d, axis=1)

# Effect of perturbations (PC1-PC5) on each stage of development at t =
0 and t = 150
df_1.columns.get_loc("Proliferation")
df_1.columns.get_loc("Differentiation")
df_1.columns.get_loc("Synaptogenesis")
df_1.columns.get_loc("Neurodevelopment")

df_1= df_1.iloc[:, [2567, 2717, 604,754, 3322, 3472, 2114, 2264]]
df_1.to_excel('proliferation_factors_sim3.xlsx')
```

Script 10: Script for perturbation of genes regulating differentiation on each stage of neurodevelopment (Simulation 3, PC1-PC5)

```
"""
GRM5_ap simulator (Differentiation)

It is also a demonstration on how the collector works

"""
import boolean2
from boolean2 import Model, util
from random import choice
import xlsxwriter
import pandas as pd
from pylab import *
from boolean2 import util
import matplotlib.pyplot as plt


# ocasionally randomized nodes
TARGETS = set( "Proliferation".split() )
TARGETS = set( "Differentiation".split() )
TARGETS = set( "Synaptogenesis".split() )
TARGETS = set( "Neurodevelopment".split() )

def new_getvalue( state, name, p):
    """
    Called every time a node value is used in an expression.
    It will override the value for the current step only.
    Returns random values for the node states
    """
    global TARGETS
    value = util.default_get_value( state, name, p )

    if name in TARGETS:
        # pick at random from True, False and original value
        return choice( [True, False, value] )
    else:
        return value

def run( text, nodes, repeat, steps ):
    """
    Runs the simulation and collects the nodes into a collector,
    a convenience class that can average the values that it collects
    """
    coll = util.Collector()

    for i in xrange( repeat ):
        engine  = Model( mode='async', text=text )
        engine.RULE_GETVALUE = new_getvalue
        # minimalist initial conditions, missing nodes set to false
        engine.initialize( missing=util.false )
        engine.iterate( steps=steps)
        coll.collect( states=engine.states, nodes=nodes )

    print '- completed'
    avgs = coll.get_averages( normalize=True )
    return avgs

if __name__ == '__main__':
```

```
    text = file( 'Simulation3.txt').read() # Input file for Simulation
3
    # this collects the state of all nodes
    NODES  = boolean2.all_nodes( text )

    # REPEAT = 1000, STEPS=150
    REPEAT = 1000
    STEPS  = 150

    data = []

    print '- starting simulation with REPEAT=%s, STEPS=%s' % (REPEAT,
STEPS)
    mtext = boolean2.modify_states( text=text, turnon=['GRM5', 'ERK1',
'ERK2', 'miR335', 'miR4719', 'RUNX2', 'TAL1', 'REST', 'YAP1' ] )
    avgs = run( text=mtext, repeat=REPEAT, nodes=NODES, steps=STEPS)
    data.append( avgs ) # Perturbation condition 2
    mtext = boolean2.modify_states( text=text, turnoff=['GRM5', 'ERK1',
'ERK2', 'miR335', 'miR4719', 'RUNX2', 'TAL1', 'REST', 'YAP1' ] )
    avgs = run( text=mtext, repeat=REPEAT, nodes=NODES, steps=STEPS)
    data.append( avgs ) # Perturbation condition 3
    # Single node perturbation
    factorslist = ['miR335', 'miR4719', 'RUNX2', 'TAL1', 'REST',
'YAP1']
    for i in factorslist:
        mtext = boolean2.modify_states( text=text, turnon=[i] )
        avgs = run( text=mtext, repeat=REPEAT, nodes=NODES,
steps=STEPS)
        data.append( avgs ) # Perturbation condition 1
        mtext = boolean2.modify_states( text=text, turnoff=[i] )
        avgs = run( text=mtext, repeat=REPEAT, nodes=NODES,
steps=STEPS)
        data.append( avgs ) # Perturbation condition 1
        mtext = boolean2.modify_states( text=text, turnon=['GRM5',
'ERK1', 'ERK2',i] )
        avgs = run( text=mtext, repeat=REPEAT, nodes=NODES,
steps=STEPS)
        data.append( avgs ) # Perturbation condition 4
        mtext = boolean2.modify_states( text=text, turnoff=['GRM5',
'ERK1', 'ERK2', i] )
        avgs = run( text=mtext, repeat=REPEAT, nodes=NODES,
steps=STEPS)
        data.append( avgs ) # Perturbation condition 5
        fname = 'GRM5_differentiation_factors.bin'
        util.bsave( data, fname=fname )
        print '- data saved into %s' % fname
       data = util.bload(
'GRM5_differentiation_factors.bin')#Perturbation results
        avgs = data
        df = pd.DataFrame.from_dict(data)
        df.to_excel("differentaition_factors_sim3_clubbed.xlsx")
        #print (df)

        avgs = data
        for x in data:
            f = lambda x:'Value {}'.format(x+1)
            d = {x: pd.DataFrame(df[x].values.tolist(),
index=df.index).rename(columns=f)
```

```
            for x in df.columns}
            df_1 = pd.concat(d, axis=1)

# Effect of perturbation conditions (PC1-PC5) on each stage of
development at t = 0 and t = 150
df_1.columns.get_loc("Proliferation")
df_1.columns.get_loc("Differentiation")
df_1.columns.get_loc("Synaptogenesis")
df_1.columns.get_loc("Neurodevelopment")


df_1= df_1.iloc[:, [2567, 2717, 604,754, 3322, 3472, 2114, 2264]]
df_1.to_excel('differentiation_factors_sim3.xlsx')
```

Script 11: Script for perturbation of genes regulating synaptogenesis on each stage of development (Simulation 1, PC1-PC5)

```
"""
GRM5_ap simulator (Synaptogenesis)

It is also a demonstration on how the collector works

"""
import boolean2
from boolean2 import Model, util
from random import choice
import xlsxwriter
import pandas as pd
from pylab import *
from boolean2 import util
import matplotlib.pyplot as plt


# occasionally randomized nodes
TARGETS = set( "Proliferation".split() )
TARGETS = set( "Differentiation".split() )
TARGETS = set( "Synaptogenesis".split() )
TARGETS = set( "Neurodevelopment".split() )


def new_getvalue( state, name, p):
    """
    Called every time a node value is used in an expression.
    It will override the value for the current step only.
    Returns random values for the node states
    """
    global TARGETS
    value = util.default_get_value( state, name, p )

    if name in TARGETS:
        # pick at random from True, False and original value
        return choice( [True, False, value] )
    else:
        return value

def run( text, nodes, repeat, steps ):
    """
    Runs the simulation and collects the nodes into a collector,
    a convenience class that can average the values that it collects.
    """
    coll = util.Collector()

    for i in xrange( repeat ):
        engine  = Model( mode='async', text=text )
        engine.RULE_GETVALUE = new_getvalue
        # minimalist initial conditions, missing nodes set to false
        engine.initialize( missing=util.false )
        engine.iterate( steps=steps)
        coll.collect( states=engine.states, nodes=nodes )

    print '- completed'
    avgs = coll.get_averages( normalize=True )
    return avgs

if __name__ == '__main__':
```

```
    text = file( 'Simulation3.txt').read() # Input file for Simulation
3

    # this collects the state of all nodes
    NODES  = boolean2.all_nodes( text )

    # REPEAT = 1000, STEPS=150

    REPEAT = 1000
    STEPS  = 150

    data = []

    print '- starting simulation with REPEAT=%s, STEPS=%s' % (REPEAT,
STEPS)
    mtext = boolean2.modify_states( text=text, turnon=['CAM', 'GRM5',
'SHANK3', 'SIAH1',  'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2',
'miR137', 'PKC', 'UBE2I', 'DISC1', 'miR335', 'miR4719', 'RUNX2',
'TAL1', 'REST', 'YAP1'] )
    avgs = run( text=mtext, repeat=REPEAT, nodes=NODES, steps=STEPS)
    data.append( avgs ) # Perturbation condition 2
    mtext = boolean2.modify_states( text=text, turnoff=['CAM', 'GRM5',
'SHANK3', 'SIAH1',  'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2',
'miR137', 'PKC', 'UBE2I', 'DISC1', 'miR335', 'miR4719', 'RUNX2',
'TAL1', 'REST', 'YAP1'] )
    avgs = run( text=mtext, repeat=REPEAT, nodes=NODES, steps=STEPS)
    data.append( avgs ) # Perturbation condition  3
    factorslist = ['miR335', 'miR4719', 'RUNX2', 'TAL1', 'REST',
'YAP1']
    for i in factorslist:
        mtext = boolean2.modify_states( text=text, turnon=[i] )
        avgs = run( text=mtext, repeat=REPEAT, nodes=NODES,
steps=STEPS)
        data.append( avgs ) # Perturbation condition 1
        mtext = boolean2.modify_states( text=text, turnoff=[i] )
        avgs = run( text=mtext, repeat=REPEAT, nodes=NODES,
steps=STEPS)
        data.append( avgs ) # Perturbation condition 1
        mtext = boolean2.modify_states( text=text, turnon=['CAM',
'GRM5',  'SHANK3', 'SIAH1',  'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2',
'miR137', 'PKC', 'UBE2I', 'DISC1',i] )
        avgs = run( text=mtext, repeat=REPEAT, nodes=NODES,
steps=STEPS)
        data.append( avgs ) # Perturbation condition 4
        mtext = boolean2.modify_states( text=text, turnoff=['CAM',
'GRM5',  'SHANK3', 'SIAH1',  'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2',
'miR137', 'PKC', 'UBE2I', 'DISC1', i] )
        avgs = run( text=mtext, repeat=REPEAT, nodes=NODES,
steps=STEPS)
        data.append( avgs ) # Perturbation condition 5
        fname = 'GRM5_synaptogenesis_factors.bin'
        util.bsave( data, fname=fname )
        print '- data saved into %s' % fname

        data = util.bload( 'GRM5_synaptogenesis_factors.bin')
        avgs = data
        df = pd.DataFrame.from_dict(data)
        df.to_excel("synaptogenesis_factors_sim3_clubbed.xlsx")
```

```
        avgs = data
        for x in data:
            f = lambda x:'Value {}'.format(x+1)
          d = {x: pd.DataFrame(df[x].values.tolist(),
index=df.index).rename(columns=f)
          for x in df.columns}
          df_1 = pd.concat(d, axis=1)

# Effect of perturbations (PC1-PC5) on each stage of development at t =
0 and t = 150
df_1.columns.get_loc("Proliferation")
df_1.columns.get_loc("Differentiation")
df_1.columns.get_loc("Synaptogenesis")
df_1.columns.get_loc("Neurodevelopment")

df_1= df_1.iloc[:, [2567, 2717, 604,754, 3322, 3472, 2114, 2264]]
df_1.to_excel('synaptogenesis_factors_sim3.xlsx')
```

Script 12: Script for perturbation of genes involved in regulation of overall neurodevelopment process on each stage of development (Simulation 3, PC1-PC5)

```
"""
GRM5_ap simulator (all stages)

It is also a demonstration on how the collector works

"""
import boolean2
from boolean2 import Model, util
from random import choice
import xlsxwriter
import pandas as pd
from pylab import *
from boolean2 import util
import matplotlib.pyplot as plt


# ocasionally randomized nodes
TARGETS = set( "Proliferation".split() )
TARGETS = set( "Differentiation".split() )
TARGETS = set( "Synaptogenesis".split() )
TARGETS = set( "Neurodevelopment".split() )


def new_getvalue( state, name, p):
    """
    Called every time a node value is used in an expression.
    It will override the value for the current step only.
    Returns random values for the node states
    """
    global TARGETS
    value = util.default_get_value( state, name, p )

    if name in TARGETS:
        # pick at random from True, False and original value
        return choice( [True, False, value] )
    else:
        return value

def run( text, nodes, repeat, steps ):
    """
    Runs the simulation and collects the nodes into a collector,
    a convenience class that can average the values that it collects.
    """
    coll = util.Collector()

    for i in xrange( repeat ):
        engine  = Model( mode='async', text=text )
        engine.RULE_GETVALUE = new_getvalue
        # minimalist initial conditions, missing nodes set to false
        engine.initialize( missing=util.false )
        engine.iterate( steps=steps)
        coll.collect( states=engine.states, nodes=nodes )

    print '- completed'
    avgs = coll.get_averages( normalize=True )
    return avgs

if __name__ == '__main__':
```

```
    text = file( 'Simulation3.txt').read() # Input file for Simulation
3

    # this collects the state of all nodes
    NODES  = boolean2.all_nodes( text )

    # REPEAT = 1000, STEPS=150
    #
    REPEAT = 1000
    STEPS  = 150

    data = []

    print '- starting simulation with REPEAT=%s, STEPS=%s' % (REPEAT,
STEPS)
    mtext = boolean2.modify_states( text=text, turnon=['ERK1', 'ERK2',
'JNK2', 'CCND1','CAM', 'GRM5',  'SHANK3', 'SIAH1',  'GRASP', 'PKD',
'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', 'miR335',
'miR4719', 'RUNX2', 'TAL1', 'REST', 'YAP1'] )
    avgs = run( text=mtext, repeat=REPEAT, nodes=NODES, steps=STEPS)
    data.append( avgs ) # Perturbation condition 2
    mtext = boolean2.modify_states( text=text, turnoff=['ERK1', 'ERK2',
'JNK2', 'CCND1','CAM', 'GRM5',  'SHANK3', 'SIAH1',  'GRASP', 'PKD',
'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', 'miR335',
'miR4719', 'RUNX2', 'TAL1', 'REST', 'YAP1' ] )
    avgs = run( text=mtext, repeat=REPEAT, nodes=NODES, steps=STEPS)
    data.append( avgs ) # Perturbation condition 3
    #Single node perturbation
    factorslist = ['miR335', 'miR4719', 'RUNX2', 'TAL1', 'REST',
'YAP1']
    for i in factorslist:
        mtext = boolean2.modify_states( text=text, turnon=[i] )
        avgs = run( text=mtext, repeat=REPEAT, nodes=NODES,
steps=STEPS)
        data.append( avgs ) # Perturbation condition 1
        mtext = boolean2.modify_states( text=text, turnoff=[i] )
        avgs = run( text=mtext, repeat=REPEAT, nodes=NODES,
steps=STEPS)
        data.append( avgs ) # Perturbation condition 1
        mtext = boolean2.modify_states( text=text, turnon=['ERK1',
'ERK2', 'JNK2', 'CCND1','CAM', 'GRM5',  'SHANK3', 'SIAH1',  'GRASP',
'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1',i]
)
        avgs = run( text=mtext, repeat=REPEAT, nodes=NODES,
steps=STEPS)
        data.append( avgs ) # Perturbation condition 4
        mtext = boolean2.modify_states( text=text, turnoff=['ERK1',
'ERK2', 'JNK2', 'CCND1','CAM', 'GRM5',  'SHANK3', 'SIAH1',  'GRASP',
'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', i]
)
        avgs = run( text=mtext, repeat=REPEAT, nodes=NODES,
steps=STEPS)
        data.append( avgs ) # Perturbation condition 5
        fname = 'GRM5_neurodevelopment_factors.bin'
        util.bsave( data, fname=fname )
        print '- data saved into %s' % fname
```

```
        data = util.bload( 'GRM5_neurodevelopment_factors.bin') #
Perturbation results
        avgs = data
        df = pd.DataFrame.from_dict(data)
        df.to_excel("neurodevelopment_factors_sim3_clubbed.xlsx")
        #print (df)

        avgs = data
        for x in data:
            f = lambda x:'Value {}'.format(x+1)
          d = {x: pd.DataFrame(df[x].values.tolist(),
index=df.index).rename(columns=f)
           for x in df.columns}
           df_1 = pd.concat(d, axis=1)

# Effect of perturbations (PC1-PC5) on each stage of development at t =
0 and t =150
df_1.columns.get_loc("Proliferation")
df_1.columns.get_loc("Differentiation")
df_1.columns.get_loc("Synaptogenesis")
df_1.columns.get_loc("Neurodevelopment")

df_1= df_1.iloc[:, [2567, 2717, 604,754, 3322, 3472, 2114, 2264]]
df_1.to_excel('neurodevelopment_factors_sim3.xlsx')
```

Step 3: Analysis of perturbation results

We exported the perturbation results in excel sheet and analyzed the effect of each perturbation condition (PC1-PC5) on stages of neurodevelopment. Effect of each perturbation on each neurodevelopmental stage is calculated as activation frequency. For example, if the neurodevelopmental stage was upregulated (ON) in all the 1000 simulations, the activation frequency is 1. If the neurodevelopmental stage was downregulated (OFF) in all the 1000 simulations, the activation frequency is 0. Out of 1000 simulations, if the neurodevelopmental stage was upregulated 514 times, then the activation frequency is 0.514. We have summarized the perturbation results at t=0 and t=150.

Table 9: Effect of perturbation of genes regulating proliferation on each stage of prenatal neurodevelopment (Simulation 3, PC1-PC5)

Activation frequency of each stage at *t* = 0 and *t* =150 (1000 simulations at each time step)								Nodes perturbed	Perturbation	Perturbation condition (PC)
Proliferation		Differentiation		Synaptogenesis		Neurodevelopment (all stages)				
Value 0	Value 150	Value 0	Value 150	Value 0	Value 150	Value 0	Value 150			
0	1	0	1	0	0	0	0	GRM5', 'ERK1', 'ERK2', JNK2','CCND1','miR335', 'miR4719', 'RUNX2', 'TAL1', 'REST', 'YAP1'	OE	PC2
0	0	0	0	0	0	0	0	GRM5', 'ERK1', 'ERK2', 'JNK2','CCND1','miR335', 'miR4719', 'RUNX2', 'TAL1', 'REST', 'YAP1'	KO	PC3
0	0	0	0	0	0	0	0	'miR335'	OE	PC1
0	0	0	0	0	0	0	0	'miR335'	KO	PC1
0	1	0	1	0	0	0	0	GRM5', 'ERK1', 'ERK2', 'JNK2','CCND1','miR335'	OE	PC4
0	0	0	0	0	0	0	0	GRM5', 'ERK1', 'ERK2', 'JNK2','CCND1','miR335'	KO	PC5
0	0	0	0	0	0	0	0	miR4719'	OE	PC1
0	0	0	0	0	0	0	0	miR4719'	KO	PC1
0	1	0	1	0	0	0	0	GRM5', 'ERK1', 'ERK2', 'JNK2','CCND1','miR4719'	OE	PC4
0	0	0	0	0	0	0	0	GRM5', 'ERK1', 'ERK2', 'JNK2','CCND1','miR4719'	KO	PC5
0	0	0	0	0	0	0	0	'RUNX2'	OE	PC1
0	0	0	0	0	0	0	0	'RUNX2'	KO	PC1
0	1	0	1	0	0	0	0	GRM5', 'ERK1', 'ERK2', 'JNK2','CCND1','RUNX2'	OE	PC4
0	0	0	0	0	0	0	0	GRM5', 'ERK1', 'ERK2', 'JNK2','CCND1','RUNX2'	KO	PC5
0	0	0	0	0	0	0	0	'TAL1'	OE	PC1
0	0	0	0	0	0	0	0	'TAL1'	KO	PC1

0	1	0	1	0	0	0	0	GRM5', 'ERK1', 'ERK2', 'JNK2','CCND1','TAL1'	OE	PC4
0	0	0	0	0	0	0	0	GRM5', 'ERK1', 'ERK2', 'JNK2','CCND1','TAL1'	KO	PC5
0	0	0	0	0	0	0	0	'REST'	OE	PC1
0	0	0	0	0	0	0	0	'REST'	KO	PC1
0	1	0	1	0	0	0	0	GRM5', 'ERK1', 'ERK2', 'JNK2','CCND1','REST'	OE	PC4
0	0	0	0	0	0	0	0	GRM5', 'ERK1', 'ERK2', 'JNK2','CCND1','REST'	KO	PC5
0	0	0	0	0	0	0	0	YAP1'	OE	PC1
0	0	0	0	0	0	0	0	YAP1'	KO	PC1
0	1	0	1	0	0	0	0	GRM5', 'ERK1', 'ERK2', 'JNK2','CCND1','YAP1'	OE	PC4
0	0	0	0	0	0	0	0	GRM5', 'ERK1', 'ERK2', 'JNK2','CCND1','YAP1'	KO	PC5

Table 10: Effect of perturbation of genes regulating differentiation on each stage of prenatal neurodevelopment (Simulation 3, PC1–PC5)

Activation frequency of each stage at *t* = 0 and *t* = 150 (1000 simulations at each time step)								**Nodes perturbed**	**Perturbation**	**Perturbation condition (PC)**
Proliferation		**Differentiation**		**Synaptogenesis**		**Neurodevelopment (all stages)**				
Value 0	**Value 150**	**Value 0**	**Value 150**	**Value 0**	**Value 150**	**Value 0**	**Value 150**			
0	1	0	1	0	0	0	0	'GRM5', 'ERK1', 'ERK2', 'miR335', 'miR4719', 'RUNX2', 'TAL1', 'REST', 'YAP1'	OE	PC2
0	0	0	0	0	0	0	0	GRM5', 'ERK1', 'ERK2', 'miR335', 'miR4719', 'RUNX2', 'TAL1', 'REST', 'YAP1'	KO	PC3
0	0	0	0	0	0	0	0	'miR335'	OE	PC1
0	0	0	0	0	0	0	0	miR335'	KO	PC1

0	1	0	1	0	0	0	0	GRM5', 'ERK1', 'ERK2', 'miR335', 'miR4719', 'RUNX2', 'TAL1', 'REST', 'YAP1', 'miR335'	OE	PC4
0	0	0	0	0	0	0	0	GRM5', 'ERK1', 'ERK2', 'miR335', 'miR4719', 'RUNX2', 'TAL1', 'REST', 'YAP1', 'miR335'	KO	PC5
0	0	0	0	0	0	0	0	'miR4719'	OE	PC1
0	0	0	0	0	0	0	0	miR4719'	KO	PC1
0	1	0	1	0	0	0	0	GRM5', 'ERK1', 'ERK2', 'miR335'	OE	PC4
0	0	0	0	0	0	0	0	GRM5', 'ERK1', 'ERK2', 'miR335'	KO	PC5
0	0	0	0	0	0	0	0	'RUNX2'	OE	PC1
0	0	0	0	0	0	0	0	'RUNX2'	KO	PC1
0	1	0	1	0	0	0	0	GRM5', 'ERK1', 'ERK2', 'RUNX2'	OE	PC4
0	0	0	0	0	0	0	0	GRM5', 'ERK1', 'ERK2', 'RUNX2'	KO	PC5
0	0	0	0	0	0	0	0	TAL1'	OE	PC1
0	0	0	0	0	0	0	0	'TAL1'	KO	PC1
0	1	0	1	0	0	0	0	GRM5', 'ERK1', 'ERK2', 'TAL1'	OE	PC4
0	0	0	0	0	0	0	0	GRM5', 'ERK1', 'ERK2', 'TAL1'	KO	PC5
0	0	0	0	0	0	0	0	REST'	OE	PC1
0	0	0	0	0	0	0	0	REST'	KO	PC1
0	1	0	1	0	0	0	0	GRM5', 'ERK1', 'ERK2', 'REST'	OE	PC4
0	0	0	0	0	0	0	0	GRM5', 'ERK1', 'ERK2', 'REST'	KO	PC5
0	0	0	0	0	0	0	0	YAP1'	OE	PC1
0	0	0	0	0	0	0	0	'YAP1'	KO	PC1
0	1	0	1	0	0	0	0	GRM5', 'ERK1', 'ERK2', 'YAP1'	OE	PC4
0	0	0	0	0	0	0	0	GRM5', 'ERK1', 'ERK2','YAP1'	KO	PC5

Table 10: Effect of perturbation of genes regulating synaptogenesis on each stage of prenatal neurodevelopment (Simulation 3, PC1-PC5)

Activation frequency of each stage at *t* = 0 and *t* = 150 (1000 simulations at each time step)								Nodes perturbed	Perturbation	Perturbation condition (PC)
Proliferation		Differentiation		Synaptogenesis		Neurodevelopment (all stages)				
Value 0	Value 150	Value 0	Value 150	Value 0	Value 150	Value 0	Value 150			
0	0	0	0	0	1	0	0	'CAM', 'GRM5', 'SHANK3', 'SIAH1', 'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', 'miR335', 'miR4719', 'RUNX2', 'TAL1', 'REST', 'YAP1'	OE	PC2
0	0	0	0	0	0	0	0	CAM', 'GRM5', 'SHANK3', 'SIAH1', 'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', 'miR335', 'miR4719', 'RUNX2', 'TAL1', 'REST', 'YAP1'	KO	PC3
0	0	0	0	0	0	0	0	miR335'	OE	PC1
0	0	0	0	0	0	0	0	'miR335'	KO	PC1
0	0	0	0	0	1	0	0	CAM', 'GRM5', 'SHANK3', 'SIAH1', 'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', 'miR335'	OE	PC4
0	0	0	0	0	0	0	0	CAM', 'GRM5', 'SHANK3', 'SIAH1', 'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', 'miR335'	KO	PC5
0	0	0	0	0	0	0	0	'miR4719'	OE	PC1
0	0	0	0	0	0	0	0	'miR4719'	KO	PC1

0	0.514	0	0.514	0	1	0	0.514	CAM', 'GRM5', 'SHANK3', 'SIAH1', 'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', 'miR4719'	OE	PC4
0	0	0	0	0	0	0	0	CAM', 'GRM5', 'SHANK3', 'SIAH1', 'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1','miR4719'	KO	PC5
0	0	0	0	0	0	0	0	'RUNX2'	OE	PC1
0	0	0	0	0	0	0	0	'RUNX2'	KO	PC1
0	1	0	1	0	1	0	1	CAM', 'GRM5', 'SHANK3', 'SIAH1', 'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', 'RUNX2'	OE	PC4
0	0	0	0	0	0	0	0	CAM', 'GRM5', 'SHANK3', 'SIAH1', 'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', 'RUNX2'	KO	PC5
0	0	0	0	0	0	0	0	'TAL1'	OE	PC1
0	0	0	0	0	0	0	0	'TAL1'	KO	PC1
0	1	0	1	0	1	0	1	CAM', 'GRM5', 'SHANK3', 'SIAH1', 'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', 'TAL1'	OE	PC4
0	0	0	0	0	0	0	0	CAM', 'GRM5', 'SHANK3', 'SIAH1', 'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', 'TAL1'	KO	PC5
0	0	0	0	0	0	0	0	'REST'	OE	PC1

0	0	0	0	0	0	0	0	REST'	KO	PC1
0	1	0	1	0	1	0	1	CAM', 'GRM5', 'SHANK3', 'SIAH1', 'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', 'REST'	OE	PC4
0	0	0	0	0	0	0	0	CAM', 'GRM5', 'SHANK3', 'SIAH1', 'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', 'REST'	KO	PC5
0	0	0	0	0	0	0	0	'YAP1	OE	PC1
0	0	0	0	0	0	0	0	'YAP1	KO	PC1
0	1	0	1	0	1	0	1	CAM', 'GRM5', 'SHANK3', 'SIAH1', 'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', 'YAP1'	OE	PC4
0	0	0	0	0	0	0	0	CAM', 'GRM5', 'SHANK3', 'SIAH1', 'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', 'YAP1'	KO	PC5

Table 11: Effect of perturbation of genes regulating overall neurodevelopment process on stages of prenatal neurodevelopment (Simulation 3, PC1-PC5)

Activation frequency of each stage at *t* = 0 and *t* = 150 (1000 simulations at each time step)								**Nodes perturbed**	**Perturbation**	**Perturbation condition (PC)**
Proliferation		**Differentiation**		**Synaptogenesis**		**Neurodevelopment (all stages)**				
Value 0	**Value 150**	**Value 0**	**Value 150**	**Value 0**	**Value 150**	**Value 0**	**Value 150**			

0	1	0	1	0	1	0	1	'ERK1', 'ERK2', 'JNK2', 'CCND1','CAM', 'GRM5', 'SHANK3', 'SIAH1', 'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', 'miR335', 'miR4719', 'RUNX2', 'TAL1', 'REST', 'YAP1'	OE	PC2
0	0	0	0	0	0	0	0	ERK1', 'ERK2', 'JNK2', 'CCND1','CAM', 'GRM5', 'SHANK3', 'SIAH1', 'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', 'miR335', 'miR4719', 'RUNX2', 'TAL1', 'REST', 'YAP1'	KO	PC3
0	0	0	0	0	0	0	0	'miR335'	OE	PC1
0	0	0	0	0	0	0	0	'miR335'	KO	PC1
0	1	0	1	0	1	0	1	'ERK1', 'ERK2', 'JNK2', 'CCND1','CAM', 'GRM5', 'SHANK3', 'SIAH1', 'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', 'miR335	OE	PC4
0	0	0	0	0	0	0	0	'ERK1', 'ERK2', 'JNK2', 'CCND1','CAM', 'GRM5', 'SHANK3', 'SIAH1', 'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', 'miR335	KO	PC5
0	0	0	0	0	0	0	0	'miR4719'	OE	PC1
0	0	0	0	0	0	0	0	'miR4719'	KO	PC1

0	1	0	1	0	1	0	1	ERK1', 'ERK2', 'JNK2', 'CCND1','CAM', 'GRM5', 'SHANK3', 'SIAH1', 'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', 'miR4719'	OE	PC4
0	0	0	0	0	0	0	0	ERK1', 'ERK2', 'JNK2', 'CCND1','CAM', 'GRM5', 'SHANK3', 'SIAH1', 'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', 'miR4719'	KO	PC5
0	0	0	0	0	0	0	0	'RUNX2'	OE	PC1
0	0	0	0	0	0	0	0	'RUNX2'	KO	PC1
0	1	0	1	0	1	0	1	ERK1', 'ERK2', 'JNK2', 'CCND1','CAM', 'GRM5', 'SHANK3', 'SIAH1', 'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', 'RUNX2'	OE	PC4
0	0	0	0	0	0	0	0	ERK1', 'ERK2', 'JNK2', 'CCND1','CAM', 'GRM5', 'SHANK3', 'SIAH1', 'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', 'RUNX2'	KO	PC5
0	0	0	0	0	0	0	0	'TAL1'	OE	PC1
0	0	0	0	0	0	0	0	'TAL1'	KO	PC1
0	1	0	1	0	1	0	1	ERK1', 'ERK2', 'JNK2', 'CCND1','CAM', 'GRM5', 'SHANK3', 'SIAH1', 'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', 'TAL1'	OE	PC4

0	0	0	0	0	0	0	0	ERK1', 'ERK2', 'JNK2', 'CCND1','CAM', 'GRM5', 'SHANK3', 'SIAH1', 'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', 'TAL1'	KO	PC5
0	0	0	0	0	0	0	0	'REST'	OE	PC1
0	0	0	0	0	0	0	0	REST'	KO	PC1
0	1	0	1	0	1	0	1	ERK1', 'ERK2', 'JNK2', 'CCND1','CAM', 'GRM5', 'SHANK3', 'SIAH1', 'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', 'REST'	OE	PC4
0	0	0	0	0	0	0	0	ERK1', 'ERK2', 'JNK2', 'CCND1','CAM', 'GRM5', 'SHANK3', 'SIAH1', 'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', 'REST'	KO	PC5
0	0	0	0	0	0	0	0	YAP1'	OE	PC1
0	0	0	0	0	0	0	0	'YAP1'	KO	PC1
0	1	0	1	0	1	0	1	ERK1', 'ERK2', 'JNK2', 'CCND1','CAM', 'GRM5', 'SHANK3', 'SIAH1', 'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', 'YAP1'	OE	PC4
0	0	0	0	0	0	0	0	ERK1', 'ERK2', 'JNK2', 'CCND1','CAM', 'GRM5', 'SHANK3', 'SIAH1', 'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', 'YAP1'	KO	PC5

Summary table showing effect of perturbation on each stage of neurodevelopment (Simulation 3, PC1-PC5)

Perturbed stages	Perturbation	Effect of perturbation on each stage at *t* =150			
		Proliferation	**Differentiation**	**Synaptogenesis**	**Neurodevelopment (all stages)**
Proliferation	PC1	0	0	0	0
	PC2	1	1	0	0
	PC3	0	0	0	0
	PC4	1	1	0	0
	PC5	0	0	0	0
Differentiation	PC1	0	0	0	0
	PC2	1	1	0	0
	PC3	0	0	0	0
	PC4	1	1	0	0
	PC5	0	0	0	0
Synaptogenesis	PC1	0	0	0	0
	PC2	0	0	1	0
	PC3	0	0	0	0
	PC4	0.514 (miR4719 OE)*	0.514 (miR4719 OE)*	1	0.514 (miR4719 OE)*
	PC5	0	0	0	0
Neurodevelopment (all stages)	PC1	0	0	0	0
	PC2	1	1	1	1
	PC3	0	0	0	0
	PC4	1	1	1	1
	PC5	0	0	0	0

* OE of each TF showed upregulation (100%) of Proliferation, Differentiation and Neurodevelopment (all stages)

Simulation 4

Step1: Input file for Simulation 4 (TF represses gene and activates miRNA expression)

```
1: GRM5* = not(miR335 or miR4719 or RUNX2 or TAL1 or REST or YAP1) and
(CAM and SHANK3 and SIAH1 and GRASP and NECAB2 and LRRC7 and DNM2)
1: ERK1* = GRM5
1: ERK2* = not(miR335 or RUNX2) and (GRM5)
1: JNK2* = GRM5
1: MAPK14* = not(GRM5 or miR4719 or TAL1)
1: CCND1* = GRM5
1: RUNX2* = not(miR335)
1: miR335* = YAP1 or TAL1
1: YAP1* = not(miR335)
1: TAL1* = not(miR4719)
1: miR4719* = miR4719
1: CAM* = CAM
1: SHANK3* = SHANK3
1: SIAH1* = SIAH1
1: GRASP* = not(miR335 or REST)
1: PKD* = GRM5
1: NECAB2* = NECAB2
1: LRRC7* = LRRC7
1: DISC1* = LRRC7
1: DNM2* =  DNM2
1: miR137* = GRM5
1: GRIA1* = not(miR137 or miR335 or REST or TAL1 or YAP1)
1: PKC* = GRM5
1: UBE2I* = PKC
1: REST* = not(miR335)
1: Proliferation* = not(MAPK14) and (GRM5 and ERK1 and ERK2 and JNK2
and CCND1)
1: Differentiation* = GRM5 and ERK1 and ERK2
1: Synaptogenesis* =  not(GRIA1) and (CAM and GRM5 and SHANK3 and SIAH1
and GRASP and PKD and NECAB2 and LRRC7 and DNM2 and miR137 and PKC and
UBE2I and DISC1)
1: Neurodevelopment* = not(GRIA1 or MAPK14) and (ERK1 and ERK2 and JNK2
and CCND1 and CAM and GRM5 and SHANK3 and SIAH1 and GRASP and PKD and
NECAB2 and LRRC7 and DNM2 and miR137 and PKC and UBE2I and DISC1)
```

Step 2: Specify the five perturbation conditions (PC1-PC5) for each stage regulated by

GRM5 interactome

PC1: OE as well as KO of each miRNA and each TF regulated by GRM5 interactome (remains same for each stage)

PC for the stage proliferation regulated by GRM5 interactome at Simulation 4

PC2: OE of genes positively regulating proliferation (regulated by GRM5 interactome) and OE of miRNAs and TFs regulating GRM5 interactome

PC3: KO of genes positively regulating proliferation (regulated by GRM5 interactome) and KO of miRNAs and TFs regulating GRM5 interactome

PC4: OE of genes positively regulating proliferation (regulated by GRM5 interactome) and OE of each factor (miRNA/ TF) regulating GRM5 interactome

PC5: KO of genes positively regulating proliferation (regulated by GRM5 interactome) and KO of each factor (miRNA/ TF) regulating GRM5 interactome

PC for the stage differentiation regulated by GRM5 interactome at Simulation 4

PC2: OE of genes positively regulating differentiation (regulated by GRM5 interactome) and OE of miRNAs and TFs regulating GRM5 interactome

PC3: KO of genes positively regulating differentiation (regulated by GRM5 interactome) and KO of miRNAs and TFs regulating GRM5 interactome

PC4: OE of genes positively regulating differentiation (regulated by GRM5 interactome) and OE of each factor (miRNA/ TF) regulating GRM5 interactome

PC5: KO of genes positively regulating differentiation (regulated by GRM5 interactome) and KO of each factor (miRNA/ TF) regulating GRM5 interactome

PC for the stage synaptogenesis regulated by GRM5 interactome at Simulation 4

PC2: OE of genes positively regulating synaptogenesis (regulated by GRM5 interactome) and OE of miRNAs and TFs regulating GRM5 interactome

PC3: KO of genes positively regulating synaptogenesis (regulated by GRM5 interactome) and KO of miRNAs and TFs regulating GRM5 interactome

PC4: OE of genes positively regulating synaptogenesis (regulated by GRM5 interactome) and OE of each factor (miRNA/ TF) regulating GRM5 interactome

PC5: KO of genes positively regulating synaptogenesis (regulated by GRM5 interactome) and KO of each factor (miRNA/ TF) regulating GRM5 interactome

PC for overall neurodevelopment process (all stages) regulated by GRM5 interactome at Simulation 4

PC2: OE of genes positively regulating overall neurodevelopment process (regulated by GRM5 interactome) and OE of miRNAs and TFs regulating GRM5 interactome

PC3: KO of genes positively regulating overall neurodevelopment process (regulated by GRM5 interactome) and KO of miRNAs and TFs regulating GRM5 interactome

PC4: OE of genes positively regulating overall neurodevelopment process (regulated by GRM5 interactome) and OE of each factor (miRNA/ TF) regulating GRM5 interactome

PC5: KO of genes positively regulating overall neurodevelopment process (regulated by GRM5 interactome) and KO of each factor (miRNA/ TF) regulating GRM5 interactome

Script 13: Script for perturbation of genes regulating proliferation on each stage of development (Simulation 4, PC1-PC5)

```
"""
GRM5_ap simulator (Proliferation)

It is also a demonstration on how the collector works

"""
import boolean2
from boolean2 import Model, util
from random import choice
import xlsxwriter
import pandas as pd
from pylab import *
from boolean2 import util
import matplotlib.pyplot as plt


# occasionally randomized nodes
TARGETS = set( "Proliferation".split() )
TARGETS = set( "Differentiation".split() )
TARGETS = set( "Synaptogenesis".split() )
TARGETS = set( "Neurodevelopment".split() )

def new_getvalue( state, name, p):
    """
    Called every time a node value is used in an expression.
    It will override the value for the current step only.
    Returns random values for the node states
    """
    global TARGETS
    value = util.default_get_value( state, name, p )

    if name in TARGETS:
        # pick at random from True, False and original value
        return choice( [True, False, value] )
    else:
        return value

def run( text, nodes, repeat, steps ):
    """
    Runs the simulation and collects the nodes into a collector,
    a convenience class that can average the values that it collects.
    """
    coll = util.Collector()

    for i in xrange( repeat ):
        engine  = Model( mode='async', text=text )
        engine.RULE_GETVALUE = new_getvalue
        # minimalist initial conditions, missing nodes set to false
        engine.initialize( missing=util.false )
        engine.iterate( steps=steps)
        coll.collect( states=engine.states, nodes=nodes )

    print '- completed'
    avgs = coll.get_averages( normalize=True )
    return avgs

if __name__ == '__main__':
```

```
    text = file( 'Simulation4.txt').read() # Input file for Simulatic
4

    # this collects the state of all nodes
    NODES  = boolean2.all_nodes( text )

    # REPEAT = 1000, STEPS=150
    REPEAT = 1000
    STEPS  = 150

    data = []

    print '- starting simulation with REPEAT=%s, STEPS=%s' % (REPEAT,
STEPS)
    mtext = boolean2.modify_states( text=text, turnon=['GRM5', 'ERK1'
'ERK2', 'JNK2','CCND1','miR335', 'miR4719', 'RUNX2', 'TAL1', 'REST',
'YAP1' ] )
    avgs = run( text=mtext, repeat=REPEAT, nodes=NODES, steps=STEPS)
    data.append( avgs ) # Perturbation condition 2
    mtext = boolean2.modify_states( text=text, turnoff=['GRM5', 'ERK1
'ERK2', 'JNK2','CCND1','miR335', 'miR4719', 'RUNX2', 'TAL1', 'REST',
'YAP1' ] )
    avgs = run( text=mtext, repeat=REPEAT, nodes=NODES, steps=STEPS)
    data.append( avgs ) # Perturbation condition 3
    # Single node perturbation
    factorslist = ['miR335', 'miR4719', 'RUNX2', 'TAL1', 'REST',
'YAP1']
    for i in factorslist:
        mtext = boolean2.modify_states( text=text, turnon=[i] )
        avgs = run( text=mtext, repeat=REPEAT, nodes=NODES,
steps=STEPS)
        data.append( avgs ) # Perturbation condition 1
        mtext = boolean2.modify_states( text=text, turnoff=[i] )
        avgs = run( text=mtext, repeat=REPEAT, nodes=NODES,
steps=STEPS)
        data.append( avgs ) # Perturbation condition 1
        mtext = boolean2.modify_states( text=text, turnon=['GRM5',
'ERK1', 'ERK2', 'JNK2','CCND1', i] )
        avgs = run( text=mtext, repeat=REPEAT, nodes=NODES,
steps=STEPS)
        data.append( avgs ) # Perturbation condition 4
        mtext = boolean2.modify_states( text=text, turnoff=['GRM5',
'ERK1', 'ERK2', 'JNK2','CCND1', i] )
        avgs = run( text=mtext, repeat=REPEAT, nodes=NODES,
steps=STEPS)
        data.append( avgs ) # Perturbation condition 5
        fname = 'GRM5_proliferation_factors.bin'
        util.bsave( data, fname=fname )
        print '- data saved into %s' % fname
       data = util.bload( 'GRM5_proliferation_factors.bin') #
Perturbation results
        avgs = data
        df = pd.DataFrame.from_dict(data)
        df.to_excel("proliferation_factors_sim4_clubbed.xlsx")
        #print (df)

        avgs = data
        for x in data:
```

```
                f = lambda x:'Value {}'.format(x+1)
          d = {x: pd.DataFrame(df[x].values.tolist(),
index=df.index).rename(columns=f)
          for x in df.columns}
          df_1 = pd.concat(d, axis=1)

# Effect of perturbations (PC1-PC5) on each stage of development at t =
0 and t = 150
df_1.columns.get_loc("Proliferation")
df_1.columns.get_loc("Differentiation")
df_1.columns.get_loc("Synaptogenesis")
df_1.columns.get_loc("Neurodevelopment")

df_1= df_1.iloc[:, [2567, 2717, 604,754, 3322, 3472, 2114, 2264]]
df_1.to_excel('proliferation_factors_sim4.xlsx')
```

Script 13: Script for perturbation of genes regulating differentiation on each stage of development (Simulation 4, PC1-PC5)

```
"""GRM5_ap simulator (Differentiation)
It is also a demonstration on how the collector works

"""
import boolean2
from boolean2 import Model, util
from random import choice
import xlsxwriter
import pandas as pd
from pylab import *
from boolean2 import util
import matplotlib.pyplot as plt


# occasionally randomized nodes
TARGETS = set( "Proliferation".split() )
TARGETS = set( "Differentiation".split() )
TARGETS = set( "Synaptogenesis".split() )
TARGETS = set( "Neurodevelopment".split() )

def new_getvalue( state, name, p):
    """
    Called every time a node value is used in an expression.
    It will override the value for the current step only.
    Returns random values for the node states
    """
    global TARGETS
    value = util.default_get_value( state, name, p )

    if name in TARGETS:
        # pick at random from True, False and original value
        return choice( [True, False, value] )
    else:
        return value

def run( text, nodes, repeat, steps ):
    """
    Runs the simulation and collects the nodes into a collector,
    a convenience class that can average the values that it collects.
    """
    coll = util.Collector()

    for i in xrange( repeat ):
        engine  = Model( mode='async', text=text )
        engine.RULE_GETVALUE = new_getvalue
        # minimalist initial conditions, missing nodes set to false
        engine.initialize( missing=util.false )
        engine.iterate( steps=steps)
        coll.collect( states=engine.states, nodes=nodes )

    print '- completed'
    avgs = coll.get_averages( normalize=True )
    return avgs

if __name__ == '__main__':
    text = file( 'Simulation4.txt').read() # Input file for Simulation
4
 # this collects the state of all nodes
```

```
    NODES  = boolean2.all_nodes( text )

    # REPEAT = 1000, STEPS=150
    REPEAT = 1000
    STEPS  = 150

    data = []

    print '- starting simulation with REPEAT=%s, STEPS=%s' % (REPEA
STEPS)
    mtext = boolean2.modify_states( text=text, turnon=['GRM5', 'ERK
'ERK2', 'miR335', 'miR4719', 'RUNX2', 'TAL1', 'REST', 'YAP1' ] )
    avgs = run( text=mtext, repeat=REPEAT, nodes=NODES, steps=STEPS
    data.append( avgs ) # Perturbation condition 2
    mtext = boolean2.modify_states( text=text, turnoff=['GRM5', 'ER
'ERK2', 'miR335', 'miR4719', 'RUNX2', 'TAL1', 'REST', 'YAP1' ] )
    avgs = run( text=mtext, repeat=REPEAT, nodes=NODES, steps=STEPS
    data.append( avgs ) # Perturbation condition 3
# Single node perturbation
factorslist = ['miR335', 'miR4719', 'RUNX2', 'TAL1', 'REST', 'YAP1'
    for i in factorslist:
        mtext = boolean2.modify_states( text=text, turnon=[i] )
        avgs = run( text=mtext, repeat=REPEAT, nodes=NODES,
steps=STEPS)
        data.append( avgs ) # Perturbation condition 1
        mtext = boolean2.modify_states( text=text, turnoff=[i] )
        avgs = run( text=mtext, repeat=REPEAT, nodes=NODES,
steps=STEPS)
        data.append( avgs ) # Perturbation condition 1
        mtext = boolean2.modify_states( text=text, turnon=['GRM5',
'ERK1', 'ERK2',i] )
        avgs = run( text=mtext, repeat=REPEAT, nodes=NODES,
steps=STEPS)
        data.append( avgs ) # Perturbation condition 4
        mtext = boolean2.modify_states( text=text, turnoff=['GRM5',
'ERK1', 'ERK2', i] )
        avgs = run( text=mtext, repeat=REPEAT, nodes=NODES,
steps=STEPS)
        data.append( avgs ) # Perturbation condition 5
        fname = 'GRM5_differentiation_factors.bin'
        util.bsave( data, fname=fname )
        print '- data saved into %s' % fname
        data = util.bload( 'GRM5_differentiation_factors.bin') #
Perturbation results
        avgs = data
        df = pd.DataFrame.from_dict(data)
        df.to_excel("differentaition_factors_sim4_clubbed.xlsx")
        #print (df)

        avgs = data
        for x in data:
            f = lambda x:'Value {}'.format(x+1)
            d = {x: pd.DataFrame(df[x].values.tolist(),
index=df.index).rename(columns=f)
         for x in df.columns}
         df_1 = pd.concat(d, axis=1)
```

```
# Effect of perturbations (PC1-PC5) on each stage of development at t =
0 and t = 150
df_1.columns.get_loc("Proliferation")
df_1.columns.get_loc("Differentiation")
df_1.columns.get_loc("Synaptogenesis")
df_1.columns.get_loc("Neurodevelopment")


df_1= df_1.iloc[:, [2567, 2717, 604,754, 3322, 3472, 2114, 2264]]
df_1.to_excel('differentiation_factors_sim4.xlsx')
```

Script 15: Script for perturbation of genes regulating synaptogenesis on each stage of development (Simulation 4, PC1-PC5)

```
"""
GRM5_ap simulator (Synaptogenesis)

It is also a demonstration on how the collector works

"""
import boolean2
from boolean2 import Model, util
from random import choice
import xlsxwriter
import pandas as pd
from pylab import *
from boolean2 import util
import matplotlib.pyplot as plt


# occasionally randomized nodes
TARGETS = set( "Proliferation".split() )
TARGETS = set( "Differentiation".split() )
TARGETS = set( "Synaptogenesis".split() )
TARGETS = set( "Neurodevelopment".split() )


def new_getvalue( state, name, p):
    """
    Called every time a node value is used in an expression.
    It will override the value for the current step only.
    Returns random values for the node states
    """
    global TARGETS
    value = util.default_get_value( state, name, p )

    if name in TARGETS:
        # pick at random from True, False and original value
        return choice( [True, False, value] )
    else:
        return value

def run( text, nodes, repeat, steps ):
    """
    Runs the simulation and collects the nodes into a collector,
    a convenience class that can average the values that it collects.
    """
    coll = util.Collector()

    for i in xrange( repeat ):
        engine  = Model( mode='async', text=text )
        engine.RULE_GETVALUE = new_getvalue
        # minimalist initial conditions, missing nodes set to false
        engine.initialize( missing=util.false )
        engine.iterate( steps=steps)
        coll.collect( states=engine.states, nodes=nodes )

    print '- completed'
    avgs = coll.get_averages( normalize=True )
    return avgs

if __name__ == '__main__':
```

```
    text = file( 'Simulation4.txt').read() # Input file for Simulation
4

    # this collects the state of all nodes
    NODES  = boolean2.all_nodes( text )

    # REPEAT = 1000, STEPS=150
    REPEAT = 1000
    STEPS  = 150

    data = []

    print '- starting simulation with REPEAT=%s, STEPS=%s' % (REPEAT,
STEPS)
    mtext = boolean2.modify_states( text=text, turnon=['CAM', 'GRM5',
'SHANK3', 'SIAH1',  'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2',
'miR137', 'PKC', 'UBE2I', 'DISC1', 'miR335', 'miR4719', 'RUNX2',
'TAL1', 'REST', 'YAP1'] )
    avgs = run( text=mtext, repeat=REPEAT, nodes=NODES, steps=STEPS)
    data.append( avgs ) # Perturbation condition 2
    mtext = boolean2.modify_states( text=text, turnoff=['CAM', 'GRM5',
'SHANK3', 'SIAH1',  'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2',
'miR137', 'PKC', 'UBE2I', 'DISC1', 'miR335', 'miR4719', 'RUNX2',
'TAL1', 'REST', 'YAP1'] )
    avgs = run( text=mtext, repeat=REPEAT, nodes=NODES, steps=STEPS)
    data.append( avgs ) # Perturbation condition 3
    # Single node perturbation
    factorslist = ['miR335', 'miR4719', 'RUNX2', 'TAL1', 'REST',
'YAP1', 'GRIA1']
    for i in factorslist:
        mtext = boolean2.modify_states( text=text, turnon=[i] )
        avgs = run( text=mtext, repeat=REPEAT, nodes=NODES,
steps=STEPS)
        data.append( avgs ) # Perturbation condition 1
        mtext = boolean2.modify_states( text=text, turnoff=[i] )
        avgs = run( text=mtext, repeat=REPEAT, nodes=NODES,
steps=STEPS)
        data.append( avgs ) # Perturbation condition 1
        mtext = boolean2.modify_states( text=text, turnon=['CAM',
'GRM5',  'SHANK3', 'SIAH1',  'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2',
'miR137', 'PKC', 'UBE2I', 'DISC1',i] )
        avgs = run( text=mtext, repeat=REPEAT, nodes=NODES,
steps=STEPS)
        data.append( avgs ) # Perturbation condition 4
        mtext = boolean2.modify_states( text=text, turnoff=['CAM',
'GRM5',  'SHANK3', 'SIAH1',  'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2',
'miR137', 'PKC', 'UBE2I', 'DISC1', i] )
        avgs = run( text=mtext, repeat=REPEAT, nodes=NODES,
steps=STEPS)
        data.append( avgs ) # Perturbation condition 5
        fname = 'GRM5_synaptogenesis_factors.bin' # Perturbation
results
        util.bsave( data, fname=fname )
        print '- data saved into %s' % fname
        data = util.bload( 'GRM5_synaptogenesis_factors.bin')
        avgs = data
        df = pd.DataFrame.from_dict(data)
        df.to_excel("synaptogenesis_factors_sim4_clubbed.xlsx")
```

```
        #print (df)

        avgs = data
        for x in data:
            f = lambda x:'Value {}'.format(x+1)
            d = {x: pd.DataFrame(df[x].values.tolist(),
index=df.index).rename(columns=f)
          for x in df.columns}
          df_1 = pd.concat(d, axis=1)

# Effect of perturbations (PC1-PC5) on each stage of development at t =
0 and t = 150
df_1.columns.get_loc("Proliferation")
df_1.columns.get_loc("Differentiation")
df_1.columns.get_loc("Synaptogenesis")
df_1.columns.get_loc("Neurodevelopment")

df_1= df_1.iloc[:, [2567, 2717, 604,754, 3322, 3472, 2114, 2264]]
df_1.to_excel('synaptogenesis_factors_sim4.xlsx')
```

Script 16: Script for perturbation of genes regulating overall neurodevelopment process on each stage of development (Simulation 4, PC1-PC5)

```
"""
GRM5_ap simulator (all stages)

It is also a demonstration on how the collector works

"""
import boolean2
from boolean2 import Model, util
from random import choice
import xlsxwriter
import pandas as pd
from pylab import *
from boolean2 import util
import matplotlib.pyplot as plt


# occasionally randomized nodes
TARGETS = set( "Proliferation".split() )
TARGETS = set( "Differentiation".split() )
TARGETS = set( "Synaptogenesis".split() )
TARGETS = set( "Neurodevelopment".split() )


def new_getvalue( state, name, p):
    """
    Called every time a node value is used in an expression.
    It will override the value for the current step only.
    Returns random values for the node states
    """
    global TARGETS
    value = util.default_get_value( state, name, p )

    if name in TARGETS:
        # pick at random from True, False and original value
        return choice( [True, False, value] )
    else:
        return value

def run( text, nodes, repeat, steps ):
    """
    Runs the simulation and collects the nodes into a collector,
    a convenience class that can average the values that it collects.
    """
    coll = util.Collector()

    for i in xrange( repeat ):
        engine  = Model( mode='async', text=text )
        engine.RULE_GETVALUE = new_getvalue
        # minimalist initial conditions, missing nodes set to false
        engine.initialize( missing=util.false )
        engine.iterate( steps=steps)
        coll.collect( states=engine.states, nodes=nodes )

    print '- completed'
    avgs = coll.get_averages( normalize=True )
    return avgs

if __name__ == '__main__':
```

```
    text = file( 'Simulation4.txt').read() # Input file for Simulation
4
    # this collects the state of all nodes
    NODES  = boolean2.all_nodes( text )

    # REPEAT = 1000, STEPS=150
    REPEAT = 1000
    STEPS  = 150

    data = []

    print '- starting simulation with REPEAT=%s, STEPS=%s' % (REPEAT,
STEPS)
    mtext = boolean2.modify_states( text=text, turnon=['ERK1', 'ERK2',
'JNK2', 'CCND1','CAM', 'GRM5',  'SHANK3', 'SIAH1',  'GRASP', 'PKD',
'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', 'miR335',
'miR4719', 'RUNX2', 'TAL1', 'REST', 'YAP1'] )
    avgs = run( text=mtext, repeat=REPEAT, nodes=NODES, steps=STEPS)
    data.append( avgs ) # Perturbation condition 2
    mtext = boolean2.modify_states( text=text, turnoff=['ERK1', 'ERK2',
'JNK2', 'CCND1','CAM', 'GRM5',  'SHANK3', 'SIAH1',  'GRASP', 'PKD',
'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', 'miR335',
'miR4719', 'RUNX2', 'TAL1', 'REST', 'YAP1' ] )
    avgs = run( text=mtext, repeat=REPEAT, nodes=NODES, steps=STEPS)
    data.append( avgs ) # Perturbation condition 3
    # Single node perturbation
    factorslist = ['miR335', 'miR4719', 'RUNX2', 'TAL1', 'REST',
'YAP1']
    for i in factorslist:
        mtext = boolean2.modify_states( text=text, turnon=[i] )
        avgs = run( text=mtext, repeat=REPEAT, nodes=NODES,
steps=STEPS)
        data.append( avgs ) # Perturbation condition 1
        mtext = boolean2.modify_states( text=text, turnoff=[i] )
        avgs = run( text=mtext, repeat=REPEAT, nodes=NODES,
steps=STEPS)
        data.append( avgs ) # Perturbation condition 1
        mtext = boolean2.modify_states( text=text, turnon=['ERK1',
'ERK2', 'JNK2', 'CCND1','CAM', 'GRM5',  'SHANK3', 'SIAH1',  'GRASP',
'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1',i]
)
        avgs = run( text=mtext, repeat=REPEAT, nodes=NODES,
steps=STEPS)
        data.append( avgs ) # Perturbation condition 4
        mtext = boolean2.modify_states( text=text, turnoff=['ERK1',
'ERK2', 'JNK2', 'CCND1','CAM', 'GRM5',  'SHANK3', 'SIAH1',  'GRASP',
'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', i]
)
        avgs = run( text=mtext, repeat=REPEAT, nodes=NODES,
steps=STEPS)
        data.append( avgs ) # Perturbation condition 5
        fname = 'GRM5_neurodevelopment_factors.bin'
        util.bsave( data, fname=fname )
        print '- data saved into %s' % fname
        data = util.bload( 'GRM5_neurodevelopment_factors.bin') #
Perturbation results
        avgs = data
        df = pd.DataFrame.from_dict(data)
```

```
        df.to_excel("neurodevelopment_factors_sim4_clubbed.xlsx")
        #print (df)

        avgs = data
        for x in data:
             f = lambda x:'Value {}'.format(x+1)
            d = {x: pd.DataFrame(df[x].values.tolist(),
index=df.index).rename(columns=f)
          for x in df.columns}
          df_1 = pd.concat(d, axis=1)

# Effect of perturbations (PC1-PC5) on each stage of development at t =
0 and t = 150
df_1.columns.get_loc("Proliferation")
df_1.columns.get_loc("Differentiation")
df_1.columns.get_loc("Synaptogenesis")
df_1.columns.get_loc("Neurodevelopment")

df_1= df_1.iloc[:, [2567, 2717, 604,754, 3322, 3472, 2114, 2264]]
df_1.to_excel('neurodevelopment_factors_sim4.xlsx')
```

Step 3: Analysis of perturbation results

We exported the perturbation results in excel sheet and analyzed the effect of each perturbation condition (PC1-PC5) on stages of neurodevelopment. Effect of each perturbation on each neurodevelopmental stage is calculated as activation frequency. For example, if the neurodevelopmental stage was upregulated (ON) in all the 1000 simulations, the activation frequency is 1. If the neurodevelopmental stage was downregulated (OFF) in all the 1000 simulations, the activation frequency is 0. Out of 1000 simulations, if the neurodevelopmental stage was upregulated 514 times, then the activation frequency is 0.514. We have summarized the perturbation results at t=0 and t=150.

Table 13: Effect of perturbation of genes regulating proliferation on each stage of prenatal neurodevelopment (Simulation 4, PC1-PC5)

Activation frequency of each stage at *t* = 0 and *t* =150 (1000 simulations at each time step)								Nodes perturbed	Perturbation	Perturbation condition (PC)
Proliferation		Differentiation		Synaptogenesis		Neurodevelopment (all stages)				
Value 0	Value 150	Value 0	Value 150	Value 0	Value 150	Value 0	Value 150			
0	1	0	1	0	0	0	0	GRM5', 'ERK1', 'ERK2', JNK2','CCND1','miR335', 'miR4719', 'RUNX2', 'TAL1', 'REST', 'YAP1'	OE	PC2
0	0	0	0	0	0	0	0	GRM5', 'ERK1', 'ERK2', 'JNK2','CCND1','miR335', 'miR4719', 'RUNX2', 'TAL1', 'REST', 'YAP1'	KO	PC3
0	0	0	0	0	0	0	0	'miR335'	OE	PC1
0	0	0	0	0	0	0	0	'miR335'	KO	PC1
0	1	0	1	0	0	0	0	GRM5', 'ERK1', 'ERK2', 'JNK2','CCND1','miR335'	OE	PC4
0	0	0	0	0	0	0	0	GRM5', 'ERK1', 'ERK2', 'JNK2','CCND1','miR335'	KO	PC5
0	0	0	0	0	0	0	0	miR4719'	OE	PC1
0	0	0	0	0	0	0	0	miR4719'	KO	PC1
0	1	0	1	0	0	0	0	GRM5', 'ERK1', 'ERK2', 'JNK2','CCND1','miR4719'	OE	PC4
0	0	0	0	0	0	0	0	GRM5', 'ERK1', 'ERK2', 'JNK2','CCND1','miR4719'	KO	PC5
0	0	0	0	0	0	0	0	'RUNX2'	OE	PC1
0	0	0	0	0	0	0	0	'RUNX2'	KO	PC1
0	1	0	1	0	0	0	0	GRM5', 'ERK1', 'ERK2', 'JNK2','CCND1','RUNX2'	OE	PC4
0	0	0	0	0	0	0	0	GRM5', 'ERK1', 'ERK2', 'JNK2','CCND1','RUNX2'	KO	PC5
0	0	0	0	0	0	0	0	'TAL1'	OE	PC1
0	0	0	0	0	0	0	0	'TAL1'	KO	PC1

0	1	0	1	0	0	0	0	GRM5', 'ERK1', 'ERK2', 'JNK2','CCND1','TAL1'	OE	PC4
0	0	0	0	0	0	0	0	GRM5', 'ERK1', 'ERK2', 'JNK2','CCND1','TAL1'	KO	PC5
0	0	0	0	0	0	0	0	'REST'	OE	PC1
0	0	0	0	0	0	0	0	'REST'	KO	PC1
0	1	0	1	0	0	0	0	GRM5', 'ERK1', 'ERK2', 'JNK2','CCND1','REST'	OE	PC4
0	0	0	0	0	0	0	0	GRM5', 'ERK1', 'ERK2', 'JNK2','CCND1','REST'	KO	PC5
0	0	0	0	0	0	0	0	YAP1'	OE	PC1
0	0	0	0	0	0	0	0	YAP1'	KO	PC1
0	1	0	1	0	0	0	0	GRM5', 'ERK1', 'ERK2', 'JNK2','CCND1','YAP1'	OE	PC4
0	0	0	0	0	0	0	0	GRM5', 'ERK1', 'ERK2', 'JNK2','CCND1','YAP1'	KO	PC5

Table 14: Effect of perturbation of genes regulating differentiation on each stage of prenatal neurodevelopment (Simulation 4, PC1-PC5)

Activation frequency of each stage at *t* = 0 and *t* = 150 (1000 simulations at each time step)								Nodes perturbed	Perturbation	Perturbation condition (PC)
Proliferation		Differentiation		Synaptogenesis		Neurodevelopment (all stages)				
Value 0	Value 150	Value 0	Value 150	Value 0	Value 150	Value 0	Value 150			
0	1	0	1	0	0	0	0	'GRM5', 'ERK1', 'ERK2', 'miR335', 'miR4719', 'RUNX2', 'TAL1', 'REST', 'YAP1'	OE	PC2
0	0	0	0	0	0	0	0	GRM5', 'ERK1', 'ERK2', 'miR335', 'miR4719', 'RUNX2', 'TAL1', 'REST', 'YAP1'	KO	PC3
0	0	0	0	0	0	0	0	'miR335'	OE	PC1
0	0	0	0	0	0	0	0	miR335'	KO	PC1
0	1	0	1	0	0	0	0	GRM5', 'ERK1', 'ERK2', 'miR335', 'miR4719', 'RUNX2', 'TAL1', 'REST', 'YAP1', 'miR335'	OE	PC4
0	0	0	0	0	0	0	0	GRM5', 'ERK1', 'ERK2', 'miR335', 'miR4719', 'RUNX2', 'TAL1', 'REST', 'YAP1', 'miR335'	KO	PC5
0	0	0	0	0	0	0	0	'miR4719'	OE	PC1
0	0	0	0	0	0	0	0	miR4719'	KO	PC1
0	1	0	1	0	0	0	0	GRM5', 'ERK1', 'ERK2', 'miR335'	OE	PC4
0	0	0	0	0	0	0	0	GRM5', 'ERK1', 'ERK2', 'miR335'	KO	PC5
0	0	0	0	0	0	0	0	'RUNX2'	OE	PC1
0	0	0	0	0	0	0	0	'RUNX2'	KO	PC1
0	1	0	1	0	0	0	0	GRM5', 'ERK1', 'ERK2', 'RUNX2'	OE	PC4

0	0	0	0	0	0	0	0	GRM5', 'ERK1', 'ERK2', 'RUNX2'	KO	PC5
0	0	0	0	0	0	0	0	TAL1'	OE	PC1
0	0	0	0	0	0	0	0	'TAL1'	KO	PC1
0	1	0	1	0	0	0	0	GRM5', 'ERK1', 'ERK2', 'TAL1'	OE	PC4
0	0	0	0	0	0	0	0	GRM5', 'ERK1', 'ERK2', 'TAL1'	KO	PC5
0	0	0	0	0	0	0	0	REST'	OE	PC1
0	0	0	0	0	0	0	0	REST'	KO	PC1
0	1	0	1	0	0	0	0	GRM5', 'ERK1', 'ERK2', 'REST'	OE	PC4
0	0	0	0	0	0	0	0	GRM5', 'ERK1', 'ERK2', 'REST'	KO	PC5
0	0	0	0	0	0	0	0	YAP1'	OE	PC1
0	0	0	0	0	0	0	0	'YAP1'	KO	PC1
0	1	0	1	0	0	0	0	GRM5', 'ERK1', 'ERK2', 'YAP1'	OE	PC4
0	0	0	0	0	0	0	0	GRM5', 'ERK1', 'ERK2','YAP1'	KO	PC5

Table 15: Effect of perturbation of genes regulating synaptogenesis on each stage of prenatal neurodevelopment (Simulation 4, PC1-PC5)

Activation frequency of each stage at *t* = 0 and *t* = 150 (1000 simulations at each time step)								Nodes perturbed	Perturbation	Perturbation condition (PC)
Proliferation		Differentiation		Synaptogenesis		Neurodevelopment				
Value 0	Value 150	Value 0	Value 150	Value 0	Value 150	Value 0	Value 150			
0	0	0	0	0	1	0	0	'CAM', 'GRM5', 'SHANK3', 'SIAH1', 'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', 'miR335', 'miR4719', 'RUNX2', 'TAL1', 'REST', 'YAP1'	OE	PC2
0	0	0	0	0	0	0	0	CAM', 'GRM5', 'SHANK3', 'SIAH1', 'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', 'miR335', 'miR4719', 'RUNX2', 'TAL1', 'REST', 'YAP1'	KO	PC3
0	0	0	0	0	0	0	0	miR335'	OE	PC1
0	0	0	0	0	0	0	0	'miR335'	KO	PC1
0	0	0	0	0	1	0	0	CAM', 'GRM5', 'SHANK3', 'SIAH1', 'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', 'miR335'	OE	PC4
0	0	0	0	0	0	0	0	CAM', 'GRM5', 'SHANK3', 'SIAH1', 'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', 'miR335'	KO	PC5
0	0	0	0	0	0	0	0	'miR4719'	OE	PC1
0	0	0	0	0	0	0	0	'miR4719'	KO	PC1

0	0.18	0	0.187	0	1	0	0.175	CAM', 'GRM5', 'SHANK3', 'SIAH1', 'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', 'miR4719'	OE	PC4
0	0	0	0	0	0	0	0	CAM', 'GRM5', 'SHANK3', 'SIAH1', 'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1','miR4719'	KO	PC5
0	0	0	0	0	0	0	0	'RUNX2'	OE	PC1
0	0	0	0	0	0	0	0	'RUNX2'	KO	PC1
0	0	0	0	0	1	0	0	CAM', 'GRM5', 'SHANK3', 'SIAH1', 'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', 'RUNX2'	OE	PC4
0	0	0	0	0	0	0	0	CAM', 'GRM5', 'SHANK3', 'SIAH1', 'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', 'RUNX2'	KO	PC5
0	0	0	0	0	0	0	0	'TAL1'	OE	PC1
0	0	0	0	0	0	0	0	'TAL1'	KO	PC1
0	0	0	0	0	1	0	0	CAM', 'GRM5', 'SHANK3', 'SIAH1', 'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', 'TAL1'	OE	PC4
0	0	0	0	0	0	0	0	CAM', 'GRM5', 'SHANK3', 'SIAH1', 'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', 'TAL1'	KO	PC5
0	0	0	0	0	0	0	0	'REST'	OE	PC1
0	0	0	0	0	0	0	0	REST'	KO	PC1

0	0	0	0	0	1	0	0	CAM', 'GRM5', 'SHANK3', 'SIAH1', 'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', 'REST'	OE	PC4
0	0	0	0	0	0	0	0	CAM', 'GRM5', 'SHANK3', 'SIAH1', 'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', 'REST'	KO	PC5
0	0	0	0	0	0	0	0	'YAP1	OE	PC1
0	0	0	0	0	0	0	0	'YAP1	KO	PC1
0	0	0	0	0	1	0	0	CAM', 'GRM5', 'SHANK3', 'SIAH1', 'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', 'YAP1'	OE	PC4
0	0	0	0	0	0	0	0	CAM', 'GRM5', 'SHANK3', 'SIAH1', 'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', 'YAP1'	KO	PC5

Table 15: Effect of perturbation of genes regulating overall neurodevelopment process on each stage of prenatal neurodevelopment (Simulation 4, PC1-PC5)

Activation frequency of each stage at *t* = 0 and *t* = 150								Nodes perturbed	Perturbation	Perturbation condition (PC)
Proliferation		Differentiation		Synaptogenesis		Neurodevelopment (all stages)				
Value 0	Value 150	Value 0	Value 150	Value 0	Value 150	Value 0	Value 150			
0	1	0	1	0	1	0	1	'ERK1', 'ERK2', 'JNK2', 'CCND1','CAM', 'GRM5', 'SHANK3', 'SIAH1', 'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', 'miR335', 'miR4719', 'RUNX2', 'TAL1', 'REST', 'YAP1'	OE	PC2
0	0	0	0	0	0	0	0	ERK1', 'ERK2', 'JNK2', 'CCND1','CAM', 'GRM5', 'SHANK3', 'SIAH1', 'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', 'miR335', 'miR4719', 'RUNX2', 'TAL1', 'REST', 'YAP1'	KO	PC3
0	0	0	0	0	0	0	0	'miR335'	OE	PC1
0	0	0	0	0	0	0	0	'miR335'	KO	PC1
0	1	0	1	0	1	0	1	'ERK1', 'ERK2', 'JNK2', 'CCND1','CAM', 'GRM5', 'SHANK3', 'SIAH1', 'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', 'miR335	OE	PC4
0	0	0	0	0	0	0	0	'ERK1', 'ERK2', 'JNK2', 'CCND1','CAM', 'GRM5', 'SHANK3', 'SIAH1', 'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', 'miR335	KO	PC5

0	0	0	0	0	0	0	0	'miR4719'	OE	PC1
0	0	0	0	0	0	0	0	'miR4719'	KO	PC1
0	1	0	1	0	1	0	1	ERK1', 'ERK2', 'JNK2', 'CCND1','CAM', 'GRM5', 'SHANK3', 'SIAH1', 'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', 'miR4719'	OE	PC4
0	0	0	0	0	0	0	0	ERK1', 'ERK2', 'JNK2', 'CCND1','CAM', 'GRM5', 'SHANK3', 'SIAH1', 'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', 'miR4719'	KO	PC5
0	0	0	0	0	0	0	0	'RUNX2'	OE	PC1
0	0	0	0	0	0	0	0	'RUNX2'	KO	PC1
0	1	0	1	0	1	0	1	ERK1', 'ERK2', 'JNK2', 'CCND1','CAM', 'GRM5', 'SHANK3', 'SIAH1', 'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', 'RUNX2'	OE	PC4
0	0	0	0	0	0	0	0	ERK1', 'ERK2', 'JNK2', 'CCND1','CAM', 'GRM5', 'SHANK3', 'SIAH1', 'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', 'RUNX2'	KO	PC5
0	0	0	0	0	0	0	0	'TAL1'	OE	PC1
0	0	0	0	0	0	0	0	'TAL1'	KO	PC1
0	1	0	1	0	1	0	1	ERK1', 'ERK2', 'JNK2', 'CCND1','CAM', 'GRM5', 'SHANK3', 'SIAH1', 'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC',	OE	PC4

								'UBE2I', 'DISC1', 'TAL1'		
0	0	0	0	0	0	0	0	ERK1', 'ERK2', 'JNK2', 'CCND1','CAM', 'GRM5', 'SHANK3', 'SIAH1', 'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', 'TAL1'	KO	PC5
0	0	0	0	0	0	0	0	'REST'	OE	PC1
0	0	0	0	0	0	0	0	REST'	KO	PC1
0	1	0	1	0	1	0	1	ERK1', 'ERK2', 'JNK2', 'CCND1','CAM', 'GRM5', 'SHANK3', 'SIAH1', 'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', 'REST'	OE	PC4
0	0	0	0	0	0	0	0	ERK1', 'ERK2', 'JNK2', 'CCND1','CAM', 'GRM5', 'SHANK3', 'SIAH1', 'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', 'REST'	KO	PC5
0	0	0	0	0	0	0	0	YAP1'	OE	PC1
0	0	0	0	0	0	0	0	'YAP1'	KO	PC1
0	1	0	1	0	1	0	1	ERK1', 'ERK2', 'JNK2', 'CCND1','CAM', 'GRM5', 'SHANK3', 'SIAH1', 'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', 'YAP1'	OE	PC4

0	0	0	0	0	0	0	0	ERK1', 'ERK2', 'JNK2', 'CCND1','CAM', 'GRM5', 'SHANK3', 'SIAH1', 'GRASP', 'PKD', 'NECAB2', 'LRRC7', 'DNM2', 'miR137', 'PKC', 'UBE2I', 'DISC1', 'YAP1'	KO	PC5

Summary table showing effect of perturbations on each stage of neurodevelopment (Simulation 4, PC1-PC5)

Perturbed stages	**Perturbations**	**Effect of perturbations on each stage at *t* =150**			
		Proliferation	**Differentiation**	**Synaptogenesis**	**Neurodevelopment (all stages)**
Proliferation	PC1	0	0	0	0
	PC2	1	1	0	0
	PC3	0	0	0	0
	PC4	1	1	0	0
	PC5	0	0	0	0
Differentiation	PC1	0	0	0	0
	PC2	1	1	0	0
	PC3	0	0	0	0
	PC4	1	1	0	0
	PC5	0	0	0	0
Synaptogenesis	PC1	0	0	0	0
	PC2	0	0	1	0
	PC3	0	0	0	0
	PC4	0.18 (miR4719 OE)*	0.187 (miR4719 OE)*	1	0.175 (miR4719 OE)*
	PC5	0	0	0	0
Neurodevelopment (all stages)	PC1	0	0	0	0
	PC2	1	1	1	1
	PC3	0	0	0	0
	PC4	1	1	1	1
	PC5	0	0	0	0

* At all perturbations, OE of each miRNA (except miR4719) showed downregulation of Proliferation, Differentiation and overall Neurodevelopment (all stages)

Summary of perturbation results

Single node perturbation

Perturbation condition 1: OE and KO of each miRNA and each TF regulating GRM5 interactome

Simulations 1-4: OE and KO of each miRNA and each TF regulating GRM5 interactome downregulated (0%) each stage of neurodevelopment and the overall neurodevelopment process (all the stages regulated by GRM5 are combined as one stage).

Multiple node perturbation

Perturbation condition 2: OE of genes positively regulating each stage of neurodevelopment, miRNAs and TFs regulating GRM5 interactome

Simulations 1-4: OE of genes positively regulating each stage of neurodevelopment (proliferation/ neurite growth/ synaptogenesis) and OE of miRNAs and TFs, upregulated each stage of neurodevelopment (100%). OE of genes positively regulating neurodevelopment and OE of miRNAs and TFs, upregulated all the stages of neurodevelopment (100%).

Perturbation condition 3: KO of all nodes (gene, miRNAs and TFs)

Simulations 1-4: KO of genes positively regulating each stage of neurodevelopment (proliferation/ neurite growth/ synaptogenesis) and KO of miRNAs and TFs, downregulated each stage of neurodevelopment (0%).

Perturbation condition 4: OE of all genes positively regulating each stage of neurodevelopment along with OE of each miRNA/TF regulating GRM5 interactome

a. Proliferation

Simulations 1-4: OE of genes positively regulating proliferation/ differentiation/ overall neurodevelopment process and OE of each miRNA/TF, upregulated proliferation (100%).

Simulations 1, 3 and 4: OE of genes positively regulating synaptogenesis and OE of miR4719 regulated proliferation (between 0%-100%).

b. Differentiation

Simulations 1-4: OE of genes positively regulating proliferation/ differentiation / overall neurodevelopment process and OE of each miRNA/TF, upregulated differentiation (100%).

Simulations 1, 3 and 4: OE of genes positively regulating synaptogenesis and OE of miR4719 regulated differentiation (between 0%-100%).

c. Synaptogenesis

Simulations 1-4: OE of genes positively regulating synaptogenesis/ overall neurodevelopment process and OE of each miRNA/TF, upregulated synaptogenesis (100%).

d. Neurodevelopment

Simulations 1-4: OE of genes positively regulating overall neurodevelopment process and OE of each miRNA/TF, upregulated neurodevelopment (100%).

Simulations 1, 3 and 4: OE of genes positively regulating synaptogenesis and OE of miR4719 regulated neurodevelopment (between 0%-100%).