Project 5: Network motifs

MCSB Bootcamp — Mathematical and Computational Track

Jun Allard

Below, five complex circuits are simulated.

1. Direct negative feedback
2. Indirect negative feedback
3. Double negative feedback
4. Incoherent feed-forward
5. The Repressilator

They are simulated in loops, across a few conditions (parameters, noise levels, …). Run each chunk to see its figures.

For each circuit:

- Describe the behavior exhibited by this circuit.
- When might this circuit be useful in biology? What functional role could it play?
- Think of examples when this circuit appears in a biological system.

One of the circuits from the slides is missing here:

6. Fast positive feedback plus slow negative feedback.

What functional role could it play? Hint: We have discussed it before. There is a textbox at the end of the notebook you can use to record your answer.

```
import numpy as np
import matplotlib.pyplot as plt
from scipy.integrate import solve_ivp
```

1. Direct negative feedback

Three parameter sets are simulated: No negative feedback, intermediate amount of negative feedback, and lots of negative feedback.

```
delta_ma = 0.05
gamma_pa = 0.02
delta_pa = 0.01

params = [
    {"Kaa": 0, "gamma_ma0": 1},
    {"Kaa": 0.1, "gamma_ma0": 5},
    {"Kaa": 1, "gamma_ma0": 40},
]

fig, axes = plt.subplots(1, len(params), figsize=(11, 4))

for i_param, ax in zip(range(len(params)), axes):

    Kaa = params[i_param]["Kaa"]  # Strength (IC50^-1) of inhibition of A by A
    gamma_ma0 = params[i_param]["gamma_ma0"]  # Baseline production rate mRNA for A

    def dxdt(t, state):
        """dma/dt and dpa/dt for a gene whose protein represses it."""
        ma, pa = state

        gamma_ma = gamma_ma0 / (1 + Kaa * pa)

        dma_dt = +gamma_ma - delta_ma * ma
        dpa_dt = +gamma_pa * ma - delta_pa * pa

        return [dma_dt, dpa_dt]

    sol = solve_ivp(dxdt, [0.0, 600], [0, 0])

    T = sol.t
    X = sol.y.T

    ax.plot(T, X[:, 0], "-r", linewidth=3)  # red for RNA
    ax.plot(T, X[:, 1], "-", color=[0.5, 0, 1], linewidth=3)  # purple for protein
    ax.set_xlim(0, 600)
    ax.set_ylim(0, 80)
    ax.set_box_aspect(1)
```

```
    ax.set_ylabel("RNA and Protein")
    ax.set_xlabel("Time (seconds)")

fig.tight_layout()
plt.show()
```

Your answer here.

Direct negative feedback exhibits…

This behavior could be useful in biological function when…

An example of where this motif occurs in biology is…

2. Indirect negative feedback

Two parameter sets are simulated: without negative feedback, and with negative feedback. In both cases, a signal is introduced at 60 seconds. And then, an even larger signal is introduced at 2000 seconds. For clarity, the plot is shown for the first 400 seconds, and then the same simulation is shown for the first 4000 seconds.

```
delta_ma = 0.08
gamma_pa = 1
delta_pa = 1

delta_mb = 1e-4
gamma_pb = 1
delta_pb = 1

params = [
    {"title": "No feedback", "Kba": 0, "Kab": 1e4, "t_max": 400},
    {"title": "Feedback", "Kba": 1.0, "Kab": 1e4, "t_max": 400},
    {"title": "Feedback, long timescale", "Kba": 1.0, "Kab": 1e4, "t_max": 4000},
]


# an input signal, e.g., externally-induced production
def input_signal(t):
    return 0 * (t < 60) + 100 * (t > 60) + 300 * (t > 2000)


for i_param in range(len(params)):
```

```
    Kba = params[i_param]["Kba"]  # Strength (IC50^-1) of inhibition of A by B
    Kab = params[i_param]["Kab"]  # Strength (EC50^-1) of activation of B by A
    t_max = params[i_param]["t_max"]

    def dxdt(t, state):
        """A driven by an input and repressed by B, B activated by A."""
        ma, pa, mb, pb = state

        gamma_ma = 1 / (1 + Kba * pb) * (1.0 + input_signal(t))
        gamma_mb = 1e-3 * (1 + Kab * pa)

        dma_dt = +gamma_ma - delta_ma * ma
        dpa_dt = +gamma_pa * ma - delta_pa * pa

        dmb_dt = +gamma_mb - delta_mb * mb
        dpb_dt = +gamma_pb * mb - delta_pb * pb

        return [dma_dt, dpa_dt, dmb_dt, dpb_dt]

    # Start well before t = 0 so the model has settled by the time the input arrives.
    sol = solve_ivp(dxdt, [-10000.0, t_max], [20, 20, 0, 0])

    # matplotlib autoscales the y-axis to everything handed to plot, not to what falls inside
    # Drop it here rather than relying on set_xlim to hide it.
    visible = sol.t >= 0
    T = sol.t[visible]
    X = sol.y.T[visible]

    fig, (ax_input, ax_a, ax_b) = plt.subplots(3, 1, figsize=(6, 7))
    ax_input.set_title(params[i_param]["title"])

    ax_input.plot(T, input_signal(T), "-k", linewidth=3)
    ax_input.set_ylabel("Input signal to A")
    ax_input.set_xlabel("Time (seconds)")
    ax_input.set_xlim(0, t_max)

    ax_a.plot(T, X[:, 1], "-", color=[0.5, 0, 1], linewidth=1)  # purple
    ax_a.set_ylabel("Protein A")
    ax_a.set_xlabel("Time (seconds)")
    ax_a.set_xlim(0, t_max)

    ax_b.plot(T, X[:, 3], "-", color=[0.5, 0, 1], linewidth=1)  # purple for protein
```

```
    ax_b.set_ylabel("Protein B")
    ax_b.set_xlabel("Time (seconds)")
    ax_b.set_xlim(0, t_max)

    fig.tight_layout()
    plt.show()
```

Your answer here.

Indirect negative feedback exhibits…

This behavior could be useful in biological function when…

An example of where this motif occurs in biology is…

3. Double negative feedback

Many simulations are performed at slightly different parameter sets (randomly selected). The steady states for all these simulations are plotted together.

```
rng = np.random.default_rng(6)

noisiness = 0.1
noisiness2 = 0.1

num_runs = 1000
storage_A = np.zeros(num_runs)
storage_B = np.zeros(num_runs)

fig, (ax_a, ax_b) = plt.subplots(2, 1, figsize=(6, 7))

for i_run in range(num_runs):

    delta_ma = 1 * (1 + noisiness * rng.random())
    gamma_pa = 1 * (1 + noisiness * rng.random())
    delta_pa = 1 * (1 + noisiness * rng.random())

    delta_mb = 1 * (1 + noisiness * rng.standard_normal())
    gamma_pb = 1 * (1 + noisiness * rng.random())
    delta_pb = 1 * (1 + noisiness * rng.random())

    Kba = 1 * (1 + noisiness2 * rng.standard_normal())  # Strength (IC50^-1) of inhibition o
    Kab = 1 * (1 + noisiness2 * rng.standard_normal())  # Strength (IC50^-1) of inhibition o
```

```
    def dxdt(t, state):
        """A and B, each repressing the other."""
        ma, pa, mb, pb = state

        gamma_ma = 3 / (1 + abs(Kba * pb) ** 3)
        gamma_mb = 3 / (1 + abs(Kab * pa) ** 3)

        dma_dt = +gamma_ma - delta_ma * ma
        dpa_dt = +gamma_pa * ma - delta_pa * pa

        dmb_dt = +gamma_mb - delta_mb * mb
        dpb_dt = +gamma_pb * mb - delta_pb * pb

        return [dma_dt, dpa_dt, dmb_dt, dpb_dt]

    initial_condition = 2 * rng.random(4)

    sol = solve_ivp(dxdt, [0.0, 60], initial_condition)

    T = sol.t
    X = sol.y.T

    if i_run < 9:
        ax_a.plot(T, X[:, 0], "-r")  # red for RNA
        ax_a.plot(T, X[:, 1], "-", color=[0.5, 0, 1])  # purple
        ax_a.set_ylabel("RNA and Protein A")
        ax_a.set_xlabel("Time (min)")

        ax_b.plot(T, X[:, 2], "-r")  # red for RNA
        ax_b.plot(T, X[:, 3], "-", color=[0.5, 0, 1])  # purple
        ax_b.set_ylabel("RNA and Protein B")
        ax_b.set_xlabel("Time (min)")

    storage_A[i_run] = X[-1, 1]
    storage_B[i_run] = X[-1, 3]

fig.tight_layout()
plt.show()
```

```
fig, (ax_a, ax_b) = plt.subplots(1, 2, figsize=(9, 4.5))
```

```
ax_a.hist(storage_A, 50)
ax_a.set_box_aspect(1)
ax_a.set_xlabel("Amount of Protein A")
ax_a.set_ylabel("Number of runs")

ax_b.hist(storage_B, 50)
ax_b.set_box_aspect(1)
ax_b.set_xlabel("Amount of Protein B")
ax_b.set_ylabel("Number of runs")

fig.tight_layout()
plt.show()
```

```
fig, ax = plt.subplots(figsize=(5, 5))
ax.scatter(storage_A, storage_B, 5)
ax.set_box_aspect(1)
ax.set_xlabel("Amount of Protein A")
ax.set_ylabel("Amount of Protein B")
plt.show()
```

Your answer here.

Double negative feedback exhibits...

This behavior could be useful in biological function when...

An example of where this motif occurs in biology is...

4. Incoherent feed-forward

```
delta_ma = 0.05
gamma_pa = 0.1
delta_pa = 0.02

delta_mb = 0.05
gamma_pb = 0.1
delta_pb = 0.02

delta_mc = 0.05
gamma_pc = 0.1
delta_pc = 0.02
```

```
gamma_ma = 10

Kab = 10  # Strength (EC50^-1) of activation of B by A
Kac = 10  # Strength (EC50^-1) of activation of C by A
Kbc = 10  # Strength (IC50^-1) of inhibition of C by B


def dxdt(t, state):
    """A drives B and C; B represses C."""
    ma, pa, mb, pb, mc, pc = state

    gamma_mb = 10 * (1 + (Kab * pa))
    gamma_mc = 10 * (1 + (Kac * pa)) / (1 + Kbc * pb)

    dma_dt = +gamma_ma - delta_ma * ma
    dpa_dt = +gamma_pa * ma - delta_pa * pa

    dmb_dt = +gamma_mb - delta_mb * mb
    dpb_dt = +gamma_pb * mb - delta_pb * pb

    dmc_dt = +gamma_mc - delta_mc * mc
    dpc_dt = +gamma_pc * mc - delta_pc * pc

    return [dma_dt, dpa_dt, dmb_dt, dpb_dt, dmc_dt, dpc_dt]


sol = solve_ivp(dxdt, [0.0, 400], [0, 0, 0, 0, 0, 0])

T = sol.t
X = sol.y.T

fig, (ax_a, ax_b, ax_c) = plt.subplots(3, 1, figsize=(6, 7))

for ax, (i_m, i_p), name in zip((ax_a, ax_b, ax_c), ((0, 1), (2, 3), (4, 5)), "ABC"):
    ax.plot(T, X[:, i_m], "-r")  # red for RNA
    ax.plot(T, X[:, i_p], "-", color=[0.5, 0, 1])  # purple
    ax.set_ylabel(f"RNA and Protein {name}")
    ax.set_xlabel("Time (seconds)")

fig.tight_layout()
plt.show()
```

Your answer here.

Incoherent feedforward exhibits…

This behavior could be useful in biological function when…

An example of where this motif occurs in biology is…

5. The Repressilator

```
delta_ma = 100 * 0.05
gamma_pa = 100 * 0.1
delta_pa = 100 * 0.02

delta_mb = 0.05
gamma_pb = 0.1
delta_pb = 0.02

delta_mc = 0.05
gamma_pc = 0.1
delta_pc = 0.02

Kca = 100  # Strength (IC50^-1) of inhibition of A by C
Kab = 100  # Strength (IC50^-1) of inhibition of B by A
Kbc = 100  # Strength (IC50^-1) of inhibition of C by B


def dxdt(t, state):
    """A ring of three repressors: C represses A, A represses B, B represses C."""
    ma, pa, mb, pb, mc, pc = state

    gamma_ma = 10 / (1 + (Kca * pc))
    gamma_mb = 10 / (1 + (Kab * pa))
    gamma_mc = 10 / (1 + (Kbc * pb))

    dma_dt = +gamma_ma - delta_ma * ma
    dpa_dt = +gamma_pa * ma - delta_pa * pa

    dmb_dt = +gamma_mb - delta_mb * mb
    dpb_dt = +gamma_pb * mb - delta_pb * pb

    dmc_dt = +gamma_mc - delta_mc * mc
```

```
    dpc_dt = +gamma_pc * mc - delta_pc * pc

    return [dma_dt, dpa_dt, dmb_dt, dpb_dt, dmc_dt, dpc_dt]


sol = solve_ivp(dxdt, [0, 1e5], [0, 0, 0, 0, 0, 0])

T = sol.t
X = sol.y.T

fig, (ax_a, ax_b, ax_c) = plt.subplots(3, 1, figsize=(6, 7))

for ax, (i_m, i_p), name in zip((ax_a, ax_b, ax_c), ((0, 1), (2, 3), (4, 5)), "ABC"):
    ax.plot(T, X[:, i_m], "-r")  # red for RNA
    ax.plot(T, X[:, i_p], "-", color=[0.5, 0, 1])  # purple
    ax.set_ylabel(f"RNA and Protein {name}")
    ax.set_xlabel("Time (seconds)")
    ax.set_xscale("log")
    ax.set_xlim(1e-4, 1e5)

fig.tight_layout()
plt.show()
```

Your answer here.

The Repressilator exhibits…

This behavior could be useful in biological function when…

An example of where this motif occurs in biology is…

6. Fast positive plus slow negative feedback

Your answer here.

Fast positive feedback plus slow negative feedback could play the functional role of…