UNIVERSITÀ
DEGLI STUDI
DI PADOVA

DIPARTIMENTO
DI INGEGNERIA
DELL'INFORMAZIONE

DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

CORSO DI LAUREA IN INGEGNERIA BIOMEDICA

"The fast component of Rho mixed-mode oscillations: a critical analysis of Tong et al. and the failure of alternative approaches"

Relatore: Prof. Morten Gram Pedersen

Laureanda: Camilla Varotto

ANNO ACCADEMICO 2024 – 2025
Data di laurea 24/07/2025

Abstract

This thesis investigates the phosphoinositide–Rho GTPase signaling network with the goal of developing a mathematical model capable of reproducing the mixed-mode oscillations of Rho observed experimentally, in order to better understand how cell contractility is regulated by a nonlinear signaling network. Starting from the incoherent feedforward loop proposed by Tong et al. to explain the fast component of these oscillations, this work shows that such a structure is insufficient to sustain periodic behavior. Several modifications are introduced, including feedback loops inspired by experimental observations and nonlinear terms based on Hill-type kinetics. Four distinct models are proposed and analyzed using phase plane analysis and the Routh–Hurwitz stability criterion. Despite these modifications, none of the models exhibited a stable limit cycle, an unstable equilibrium corresponding to sustained oscillatory behavior. Instead, all system trajectories converged toward a stable steady state, suggesting that the simplified ordinary differential equations used in this study fail to capture the complex dynamics observed in vivo. Future directions are discussed, including the incorporation of time delays, spatial dynamics and stochastic fluctuations to better reflect the biological complexity of the system.

Riassunto

In questa tesi viene approfondito lo studio della rete di segnalazione fosfoinositide–Rho GTPasi, con l'obiettivo di sviluppare un modello matematico in grado di riprodurre le oscillazioni a modalità mista di Rho osservate sperimentalmente, al fine di chiarire in che modo la contrattilità cellulare sia regolata da una rete di segnalazione non lineare. Partendo dal circuito di feedforward incoerente proposto da Tong et al. per spiegare la componente veloce di tali oscillazioni, si dimostra che questa struttura non è sufficiente a sostenere un comportamento periodico. Sono quindi state introdotte diverse modifiche alla formulazione iniziale, tra cui anelli di retroazione ispirati da osservazioni sperimentali e termini non lineari basati su cinetiche di tipo Hill. Quattro diversi modelli sono stati proposti e analizzati attraverso lo studio del piano di fase e l'applicazione del criterio di stabilità di Routh–Hurwitz. Nonostante tali modifiche, nessuno dei modelli ha mostrato la presenza di un ciclo limite stabile, ossia un equilibrio instabile associato a un comportamento oscillatorio. Al contrario, tutte le traiettorie del sistema convergono verso uno stato stazionario stabile, suggerendo che le equazioni differenziali ordinarie semplificate adottate in questo lavoro non riescono a catturare la complessità dinamica osservata in vivo. Vengono infine discusse possibili prospettive future, tra cui l'inclusione di ritardi temporali, dinamiche spaziali e fluttuazioni stocastiche, al fine di rappresentare più fedelmente la complessità biologica del sistema.

TABLE OF CONTENTS

1. Biological background: phosphoinositide-Rho GTPase network.....5

2. Tong et al. article.....6

2.1. Mixed-mode oscillation circuit.....6

2.2. Fast oscillations circuit: incoherent feed-forward loop and its limitations.....9

3. Modified model and computational approach.....12

3.1. Initial modified model.....12

3.2. Introduction of non-linear terms.....14

3.3. Stability analysis.....16

3.3.1. Phase plane and nullcline analysis.....16

3.3.2. Routh–Hurwitz stability criterion.....20

4. Conclusions and critical analysis of model failure.....23

5. Bibliography.....26

6. Figure Sources.....27

7. Appendix A – Calculations to implement Routh-Hurwitz criterion.....28

8. Appendix B – Python code.....34

1. Biological background: phosphoinositide-Rho GTPase network

The phosphoinositide–Rho GTPase network controls cell contractility spatially and temporally and it is essential for morphogenetic events such as cytokinesis. Cytokinesis is the process by which a eukaryotic cell's cytoplasm is partitioned into two daughter cells during mitosis [2]. Actomyosin networks generate contractile forces in the cell when myosin II motors slide antiparallel actin filaments. The assembly of these networks is regulated by Rho GTPase activity. During mitosis, Rho recruits and assembles local furrowing units that compose the cytokinetic ring at the plasma membrane, driving ring contraction and cell division. However, dynamic waves, pulses and oscillations have been observed in various mitotic cell types, providing evidence of a nonlinear signaling network governing Rho.

Early actomyosin-centric mechanical models described these contractile pulses focusing on the mechanical property of the network, but they lacked insight on the chemical state of the cytogel. The cytogel is defined as the meshwork of protein filaments and regulators that gives the cytoplasm mechanical strength and contractile ability [3].

Recent advances using fluorescent reporters suggest that Rho presents an oscillatory behavior as it regulates myosin II activity and drives cycles of contraction. To oscillate, a system requires a local activator and an inhibitor: Rho guanine nucleotide exchange factors (RhoGEFs) and Rho GTPase–activating proteins (RhoGAPs), respectively. Specifically, positive feedback via RhoGEFs amplifies Rho, which in turn drives actin-filament assembly and boosts myosin II activity. The accumulation of actin and myosin II triggers, after a delay, the recruitment of RhoGAPs, shutting off Rho and generating pulses of contractile activity [4].

Nonetheless, experimental studies showed that Rho oscillation periods can shift significantly within the same system ranging from tens to hundreds of seconds; this variability cannot be fully explained by RhoGEF/RhoGAP interactions alone. Interestingly, many Rho regulators are recruited to the plasma membrane through their binding to phosphoinositides, phospholipids within the cell membrane that act as signaling molecules for many physiological processes [5]. Phosphoinositides constitute a dynamic metabolic network of kinases and phosphatases whose coupling can produce complex dynamical behaviors, suggesting that oscillations in phosphoinositide levels could modulate Rho signaling like a pacemaker. Understanding how phosphoinositides interact with Rho GTPase is therefore critical for comprehending the mechanisms that govern the assembly of contractile units in cytokinesis and, more broadly, for revealing the principles by which nonlinear signaling networks regulate contractile behavior and signal transduction in the cell[1].

2. Tong et al. article

2.1 Mixed-mode oscillation circuit

In 2023, Chee San Tong, X.J. Xŭ and Min Wu from Yale University investigated the dynamical behavior of the phosphoinositide-Rho GTPase signaling network and published their findings in an article titled *"Periodicity, mixed-mode oscillations, and multiple timescales in a phosphoinositide-Rho GTPase network"* [1]. For their research, they used mitotic rat basophilic leukemia (RBL) cells treated with nocodazole, a microtubule-depolymerizing drug known from the literature to enhance oscillatory behavior. Moreover, nocodazole-treated mitotic rat cells displayed Rho oscillations with periods ranging from 30 seconds to 5 minutes, covering the range of periodicities reported across previous experimental systems.

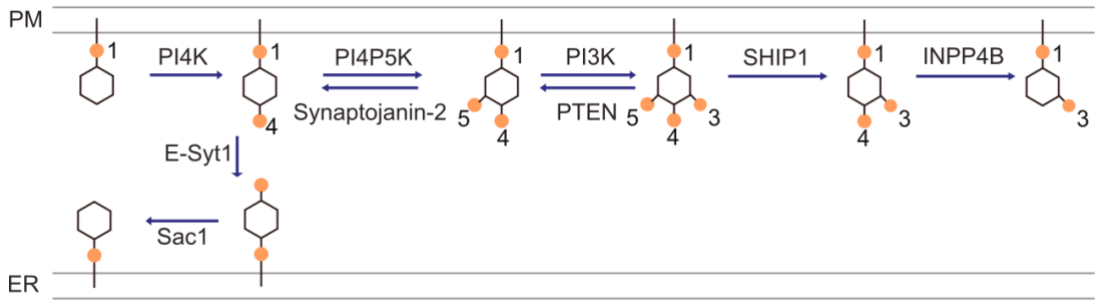


Figure 1: Phosphoinositide metabolism pathways.

Initially, in order to better understand the molecular network governing cell contractility, they searched for experimental conditions that could generate or perturb Rho simple oscillations. However, due to the heterogeneity of oscillation periodicities, it proved challenging to identify a single regulator of this behavior. Therefore, the researchers took an alternative approach. They noticed that mixed-mode oscillations (MMOs) were occasionally displayed and decided to investigate these events more closely.

Mixed-mode oscillations represent a complex dynamic regime characterized by alternating slow and fast oscillations, in which pulses are not controlled by a single regulatory mechanism but rather by the interplay of two distinct feedback loops operating at different timescales. The coupling between the two mechanisms generating mixed-mode oscillations can reveal information on the structure of a higher dimensional signaling network.

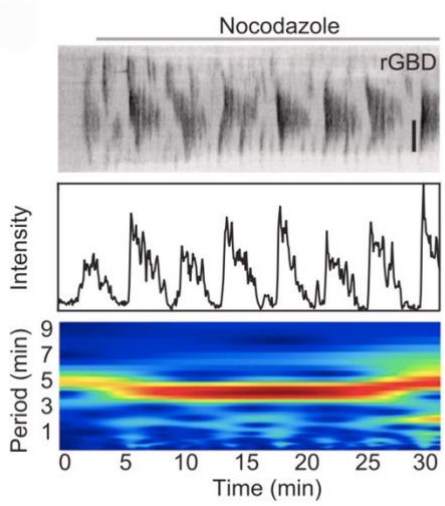


Figure 2: Kymographs, intensity profiles, and wavelet analyses of representative movies depicting Rho mixed-mode oscillations in wild-type cells.

With the aim of better understanding the nonlinear phosphoinositide network underlying Rho mixed-mode oscillations (Figure 1), the researchers conducted a series of experiments and ultimately concluded that the fast oscillations are regulated by PIP_3 and $PI(3,4)P_2$ through an activator-delayed inhibition circuit, while the slow oscillations are controlled by PI(4)P and $PI(4,5)P_2$ via an activator-substrate depletion mechanism.

In wild-type cells, mixed-mode oscillations were rarely observed (Figure 2), therefore, any condition that led to their emergence was considered significant (Figure 3). One of the first impactful perturbations was the knockdown of PTEN, a phosphatase that converts PIP_3 into $PI(4,5)P_2$, which resulted in an increased prevalence of mixed-mode oscillations. It is likely that elevated PIP_3 levels enhance the fast oscillatory component of Rho dynamics (Figure 4B). In contrast, the knockdown of Synaptojanin-2, an enzyme that dephosphorylates $PI(4,5)P_2$ to yield PI(4)P, suppressed mixed-mode oscillations. However, the most striking effect was observed with the knockdown of E-Syt1, a membrane contact site protein that facilitates lipid transfer and reduces PI(4)P levels at the plasma membrane by transporting it to the endoplasmic reticulum. E-Syt1 depletion dramatically increased the appearance of mixed-mode oscillations, with over half of the cells displaying such patterns (Figure 4B). These findings suggest a critical role for PI(4)P accumulation at the plasma membrane in enabling complex oscillatory behavior.

The well-defined high-frequency pulses in E-Syt1 knockdown cells displaying mixed-mode oscillations provided an opportunity to explore the circuit underlying fast oscillations. As PTEN negatively regulates PIP_3, the authors hypothesized that the sharp, nested peaks were PIP_3-dependent. To test this, they performed double knockdowns and found that high-frequency peaks disappeared only when INPP4B, the enzyme responsible for degrading $PI(3,4)P_2$, was also knocked down (Figure 4D), not when SHIP1, which converts PIP_3 into $PI(3,4)P_2$, was removed.

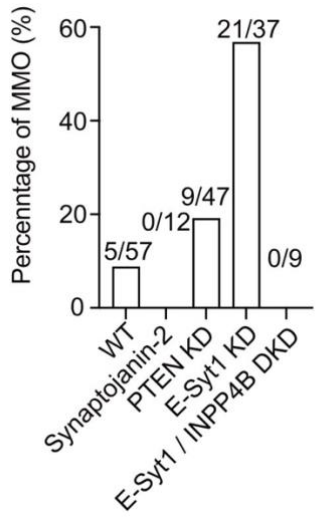


Figure 3: Bar plot indicating the percentage of cells displaying mixed-mode Rho oscillations under different conditions.

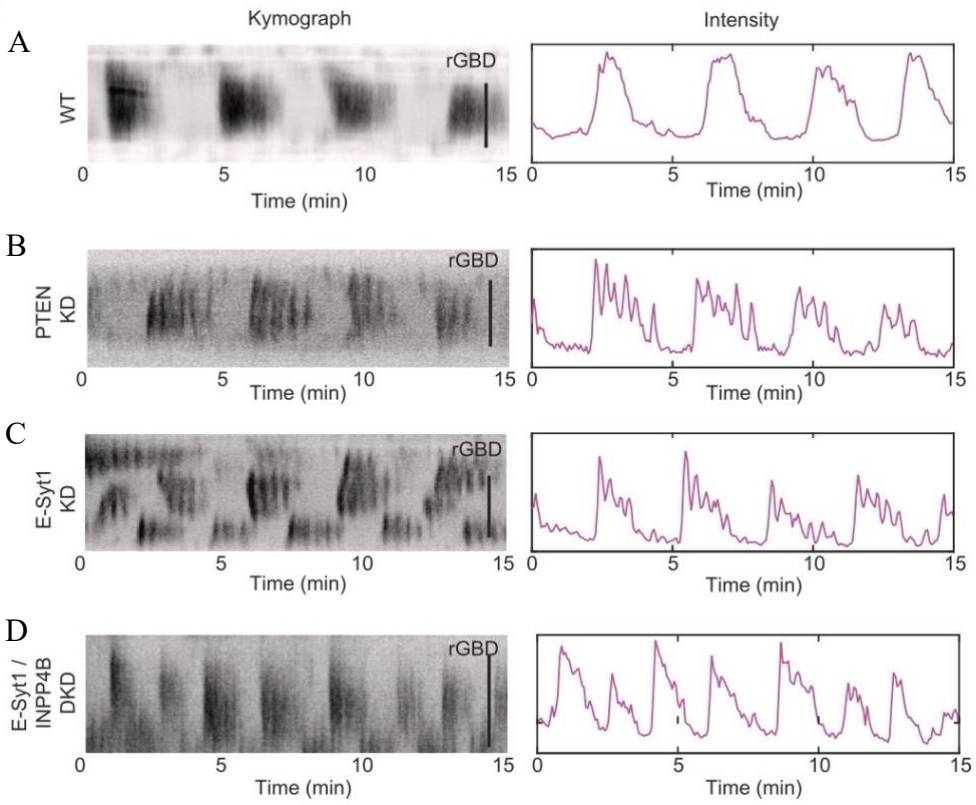


Figure 4: Representative kymographs and intensity profiles of Rho oscillations in WT cells (A), PTEN KD cells (B), E-Syt1 KD cells (C) and E-Syt1/ INPP4B double KD (DKD) cells (D).

Additionally, reducing PIP_3 levels using the PI3K-inhibitor LY294002 abolished fast oscillations altogether. These results suggested a regulatory mechanism in which PIP_3 promotes fast oscillations by accelerating $PI(3,4)P_2$ turnover through a PIP_3/INPP4B pathway. In this framework, high-levels of PIP_3 lead to a faster degradation of $PI(3,4)P_2$ and sharp high-frequency peaks, whereas lower PIP_3 levels lead to a slower degradation of $PI(3,4)P_2$, causing multiple nested peaks to merge into single peaks.

Regarding the slow oscillatory circuit, the authors perturbed PI(4)P metabolism by inhibiting PI4K with GSK-A1 in E-Syt1 knockdown cells exhibiting MMOs. This led to a transition from complex to simple oscillatory regimes and eventually to the complete cessation of Rho oscillations (Figure 5), implicating that PI(4)P depletion limits Rho activation cycle. This behavior supports the activator-substrate depletion model, in which $PI(4,5)P_2$, synthesized from PI(4)P by PI4P5K, acts as a substrate for Rho activation. To further test this mechanism, the authors co-imaged Rho activity with fluorescent reporters for PI4P5K and $PI(4,5)P_2$ and observed that both signals decreased following Rho activation. This confirmed that substrate levels were being depleted during each Rho pulse. These findings support the idea that the rate of replenishment of PI(4)P, which controls $PI(4,5)P_2$ synthesis, regulates the periodicity of Rho

activation, consistent with the notion that PI(4)P accumulation is necessary for enabling complex Rho dynamics.

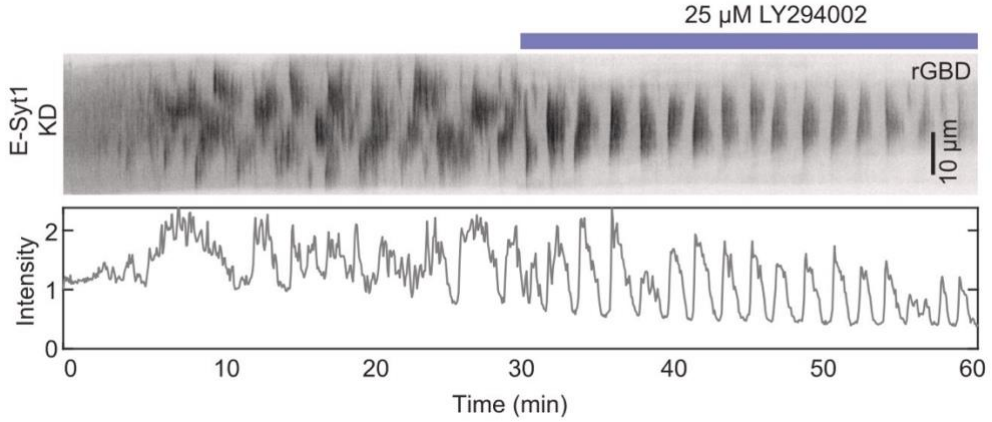


Figure 5: Representative kymographs and intensity profile of Rho oscillations in E-Syt1 KD cells treated with LY294002.

Overall, these results reveal that mixed-mode oscillations in the phosphoinositide-Rho network arise from the coupling of two regulatory circuits: a fast PIP_3/INPP4B-dependent loop and a slower PI(4)P/$PI(4,5)P_2$ substrate-limited cycle. This dual feedback model enables the system to operate near critical bifurcation points, where small changes in lipid metabolism can shift the network between simple and complex dynamical states.

2.2 Fast oscillations circuit: incoherent feed-forward loop and its limitations

In their article, the authors state that their findings suggest the presence of an incoherent feedforward loop (IFFL) regulating the fast component of the mixed-mode oscillations. Specifically, this is based on the observation that PIP_3 acts both as a precursor for $PI(3,4)P_2$ and promotes its degradation through activation of the enzyme INPP4B (Figure 6).

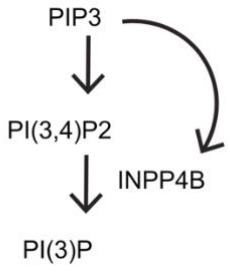
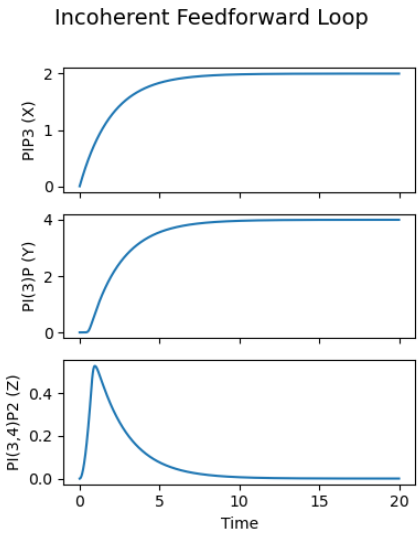


Figure 6: Incoherent feedforward loop regulating fast oscillations proposed by Tong et al.

However, IFFL are well-known pulse generators and are not capable of producing oscillations unless an additional mechanism ensures the periodic replenishment of PIP_3. In fact, as PIP3 is synthesized, $PI(3,4)P_2$ levels begin to rise. After a time delay, once PIP_3 reaches a threshold concentration, INPP4B is recruited to the plasma membrane, where it accelerates the degradation of $PI(3,4)P_2$. This results in a single peak: an initial increase in $PI(3,4)P_2$ concentration, followed by a sharp decline. Once $PI(3,4)P_2$ is degraded, the system stabilizes and remains at a steady-state equilibrium unless PIP_3 levels are reduced below the INPP4B recruitment threshold, allowing $PI(3,4)P_2$ to accumulate again.

To support this claim, the dynamics of the incoherent feedforward loop were simulated in Python using a set of ordinary differential equations (ODEs) that reflects the characteristic interactions of an IFFL (Figure B.1):



$$\frac{dX}{dt} = \alpha - \delta_X X$$

$$\frac{dY}{dt} = \beta_Y \cdot \frac{X^n}{X^n + K^n} - \delta_Y Y$$

$$\frac{dZ}{dt} = \gamma_Z \cdot X \cdot \frac{{K_Y}^n}{X^n + {K_Y}^n} - \delta_Z Z$$

Figure 7: Python simulation of the IFFL regulating the fast component of the circuit.

As anticipated, PI(3,4)P_2 concentration exhibits a single transient peak, rather than sustained high-frequency oscillations (Figure 7).

Furthermore, in the attempt of reproducing the complex dynamics observed experimentally, the authors simulated mixed-mode oscillations using a system of differential equations developed in ecology: the Hastings–Powell model, which describes the interactions between a prey, a predator and a superpredator. This model is known to generate complex patterns, including mixed-mode oscillations and chaotic dynamics, but its application in this study raises some concerns. For instance, the authors do not specify which molecules of the phosphoinositide–Rho GTPase signaling network correspond to the variables used in the model (typically *x*, *y*, and *z*). This lack of correspondence causes the simulation to be disconnected from the biological system under study. In systems biology, the use of analog models is acceptable only when accompanied by a clear interpretation. Without such context, the model serves merely as a mathematical illustration rather than a biologically meaningful explanation of the observed behavior. Moreover, representing molecular signaling circuits with ecological models presents some inconsistencies, given the differences in timescales and regulatory mechanisms between the two domains. Therefore, while the ecological model succeeds in replicating mixed-mode oscillations, its use without a justified molecular mapping limits its explanatory power and may be considered inadequate for representing the true mechanistic regulating the phosphoinositide–Rho GTPase circuit. In the following chapter, alternative models will be formulated with the

aim of reproducing the observed complex dynamics while reflecting the actual molecular components involved in the system.

3. Modified model and computational approach

3.1 Initial modified model

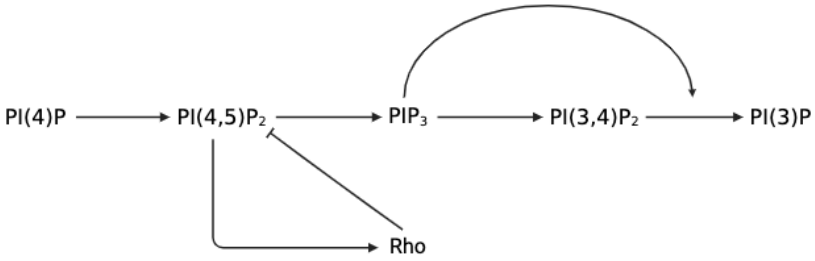


Figure 8: Phosphoinositide-Rho GTPase network proposed by Tong et al..

As the initial model, including only the incoherent feedforward loop (Figure 8), failed to produce fast oscillations and instead generated a single transient peak, it became necessary to introduce additional interactions, specifically, feedback loops, in order to enable sustained high-frequency oscillations.

In the literature concerning this specific IFFL, the hypothesis has been raised that $PI(3,4)P_2$ acts as an inhibitor of Rho activation, likely because RhoGAPs, which negatively regulate Rho, are recruited through the presence of $PI(3,4)P_2$ [6]. For this reason, I added an inhibitory interaction from $PI(3,4)P_2$ to Rho in the model (Figure 9).

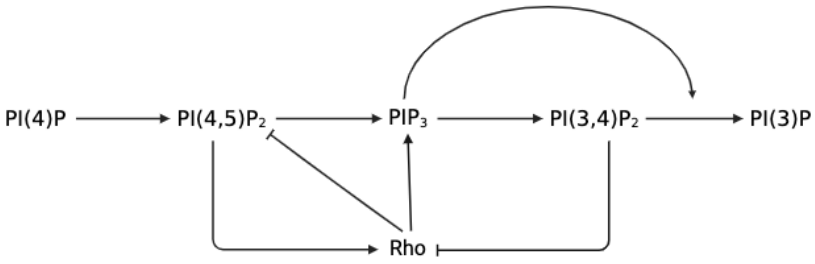


Figure 9: Proposed model describing the phosphoinositide-Rho GTPase network.

Furthermore, in the article by Tong et al., the authors suggest that in the activator–substrate depletion mechanism controlling the slow component, $PI(4,5)P_2$, the substrate, is converted into PIP_3 following interaction with Rho, the activator. To support this, they co-imaged PH_{Grp1}, a biosensor that binds to PIP_3, along with a Rho activity sensor and observed that PIP_3 levels were elevated relative to baseline when Rho was activated (Figure 10). Based on this evidence, a positive regulatory arrow going from Rho to PIP_3 was included in the proposed model (Figure 9), indicating that high Rho activity promotes PIP_3 accumulation.

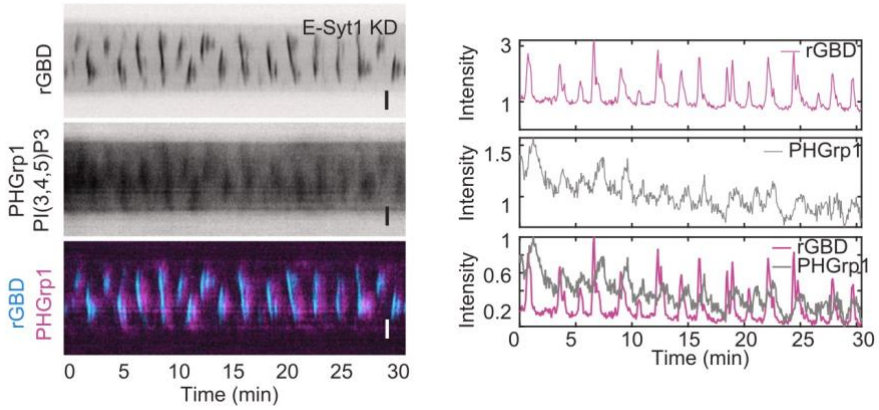


Figure 10: Kymographs and intensity profiles of Rho in E-Syt1 KD cell co-imaged with PH_{Grp1} (PIP_3).

Together, these modifications introduce a feedback structure that may allow the fast component to generate sustained oscillations, addressing the limitations of the original incoherent feedforward loop.

Linear model

From the resulting circuit (Figure 11), the following system of three ordinary differential equations (ODEs) describing the fast component of the mixed-mode oscillation was derived. A represents PIP_3, B represents $PI(3,4)P_2$ and R* represents active Rho:

$$\frac{dA}{dt} = \alpha_A R^* - k_{AB}A - \delta_A A$$

$$\frac{dB}{dt} = k_{AB}A - k_{deg}AB - \delta_B B$$

$$\frac{dR^*}{dt} = \beta(1 - R^*) - \gamma B R^*$$

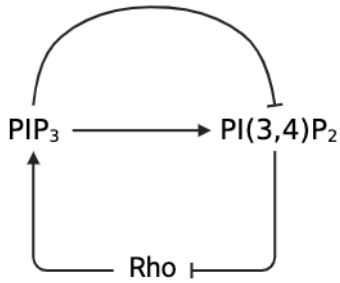


Figure 11: Proposed circuit regulating fast oscillations.

The term α_A represents the rate at which active Rho promotes the conversion of $PI(4,5)P_2$ to PIP_3, increasing A levels. Similarly, k_{AB} denotes the rate at which PIP_3 is converted to $PI(3,4)P_2$, while k_{deg} corresponds to the rate at which PIP_3 accelerates the degradation of $PI(3,4)P_2$. The parameter β represents the basal activation rate of Rho and γ quantifies the rate at which $PI(3,4)P_2$ recruits RhoGAPs to inactivate Rho.

To evaluate whether this system can generate oscillatory behavior and correctly represent the fast component of the phosphoinositide–Rho GTPase signaling network, a Python script was implemented to numerically integrate the ODEs over time trying different values for the model parameters (Figure B.2). In all cases tested, the system failed to produce sustained oscillations,

instead converging toward a steady state in which all system variables (A, B and R*) remain constant over time (Figure 12). In the next section, nonlinear terms will be introduced into the model in an attempt to enable oscillatory behavior. The absence of oscillations in the current system may be attributed to its linear nature. Nonlinearities are often necessary to amplify sensitivity and destabilize steady states, creating conditions that promote oscillatory behavior.

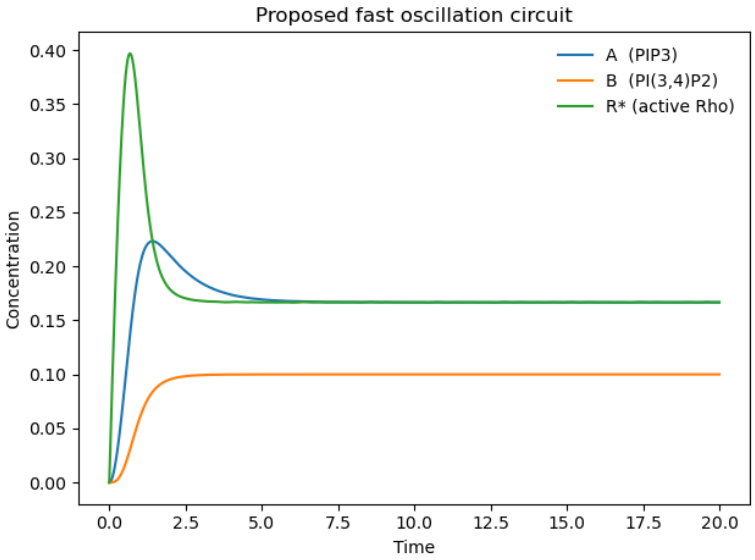


Figure 12: Simulation of the proposed fast oscillation component without non-linear terms.

3.2 Introduction of nonlinear terms

In this paragraph, nonlinear terms in the form of Hill-type kinetics are incorporated into the initial linear model to increase dynamic sensitivity and induce instabilities that could give rise to periodic oscillations.

Proposed nonlinearity 1

In this first modification, a Hill-type term is introduced in the inactivation of Rho (R*) mediated by PI(3,4)P_2 (B):

$$\frac{dA}{dt} = \alpha_A R^* - k_{AB} A - \delta_A A$$

$$\frac{dB}{dt} = k_{AB} A - k_{deg} AB - \delta_B B$$

$$\frac{dR^*}{dt} = \beta(1 - R^*) - \gamma \frac{B^n}{K^n + B^n} R^*$$

The Hill-type term makes Rho inactivation ultrasensitive to changes in the concentration of $PI(3,4)P_2$ (B). That is, Rho inactivation will occur only when B reaches the threshold concentration K, while below that value, B will have little to no effect on Rho. For sufficiently high values of n:

If B < K:

$$\lim_{n\to\infty} \frac{B^n}{K^n + B^n} = 0$$

If B > K:

$$\lim_{n\to\infty} \frac{B^n}{K^n + B^n} = 1$$

Proposed nonlinearity 2

In this formulation, a Hill-type term is added in the degradation of $PI(3,4)P_2$ (B) promoted by PIP_3 (A), making B degradation ultrasensitive to its own concentration. The Hill term regulating Rho inactivation from the previous model is maintained:

$$\frac{dA}{dt} = \alpha_A R^* - k_{AB} A - \delta_A A$$

$$\frac{dB}{dt} = k_{AB} A - k_{deg} A \frac{B^n}{K^n + B^n} - \delta_B B$$

$$\frac{dR^*}{dt} = \beta(1 - R^*) - \gamma \frac{B^n}{K^n + B^n} R^*$$

In this case, both Rho inactivation and B degradation become ultrasensitive to the concentration of $PI(3,4)P_2$ (B). This means that degradation of B will occur only once B exceeds the threshold K.

Proposed nonlinearity 3

Finally, in the third formulation, a Hill-type term is introduced in the degradation of $PI(3,4)P_2$ (B) promoted by PIP_3 (A), this time making B degradation ultrasensitive to A concentration:

$$\frac{dA}{dt} = \alpha_A R^* - k_{AB} A - \delta_A A$$

$$\frac{dB}{dt} = k_{AB} A - k_{deg} \frac{A^n}{K^n + A^n} B - \delta_B B$$

$$\frac{dR^*}{dt} = \beta(1 - R^*) - \gamma B R^*$$

Here, degradation of $PI(3,4)P_2$ (B) will happen only when PIP_3 (A) exceeds the threshold K, for sufficiently large n.

The following paragraph examines whether these nonlinear models are capable of generating oscillations, using nullcline analysis in the two-dimensional phase plane and the Routh–Hurwitz stability criterion.

3.3 Stability analysis

3.3.1 Phase plane and nullcline analysis

As an initial step in investigating the dynamic behavior of the proposed nonlinear models, the phase planes of the systems were analyzed by plotting the nullclines of two variables while assuming the third variable to be at its steady state value.

The phase plane is a graphical representation in which the x- and y-axes correspond to the concentrations of two system variables and each point on the plane represents a possible state of the system defined by these two concentrations. Nullclines are the loci of points in the phase plane where the derivative of one of the variables is zero.

Therefore, to plot the phase plane and nullclines for a system of three ordinary differential equations, the following approach was taken:

- One of the three variables was fixed at its steady state value. For example, for nonlinear model 1, the phase plane given by $PI(3,4)P_2$ (B) and Rho (R*) was constructed assuming PIP_3 (A) to be at its steady state:

$$\frac{dA}{dt} = \alpha_A R^* - k_{AB} A - \delta_A A = 0 \;\rightarrow\; A = \frac{\alpha_A}{k_{AB} + \delta_A} R^* = mR^*$$

- The nullclines of the two remaining variables were found. For nonlinear model 1, PI(3,4)P_2 (B) and Rho (R*).

  B-nullcline:

$$\frac{dB}{dt} = k_{AB}A - k_{deg}AB - \delta_B B = 0 \rightarrow R^* = \frac{\delta_B B}{k_{AB}m - k_{deg}mB}$$

  R-nullcline:

$$\frac{dR^*}{dt} = \beta(1 - R^*) - \gamma \frac{B^n}{K^n + B^n} R^* = 0 \rightarrow R^* = \frac{\beta}{\beta + \gamma \frac{B^n}{K^n + B^n}}$$

The intersection points of the nullclines correspond to the system's fixed points, that is, the concentrations at which both derivatives are zero and the system is at equilibrium. By overlaying the vector field and the trajectory obtained by numerically integrating the ODEs, it is possible to assess the stability of these points. A stable equilibrium is characterized by trajectories that converge to the fixed point, whereas a stable limit cycle, which corresponds to sustained oscillations, manifests as a closed orbit surrounding an unstable fixed point.

Nonlinear model 1

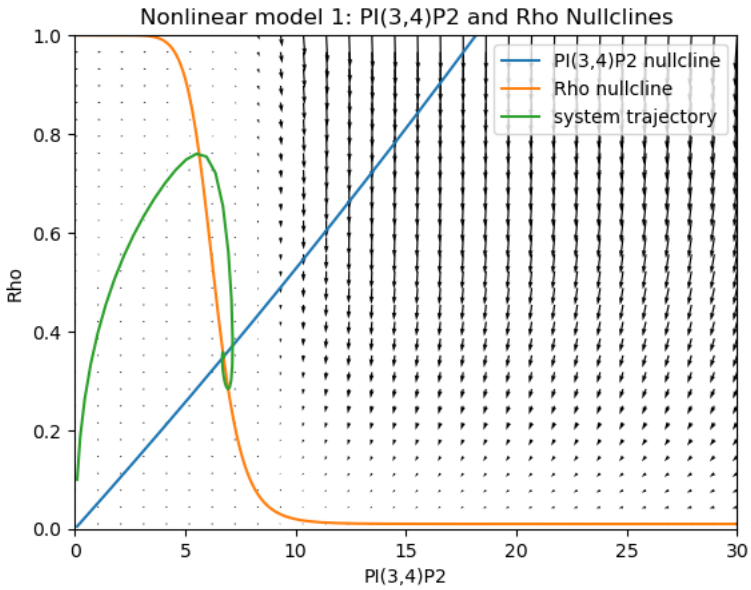


Figure 13: Phase plane, nullclines of PI(3,4)P_2 and Rho, vector field and system trajectory for nonlinear model 1, obtained using Python.

As anticipated, for nonlinear model 1, the phase plane of PI(3,4)P_2 (B) and Rho (R*) was computed using the nullcline equations shown before (Figure 13 and B.3).
The nullclines intersect at a single point, which is the system's fixed point. As shown by the system trajectory and direction field, this fixed point is attractive as all trajectories converge toward it. This behavior indicates that the system stabilizes at a steady state, with no emergence of sustained oscillatory behavior, regardless of initial conditions.

Nonlinear model 2

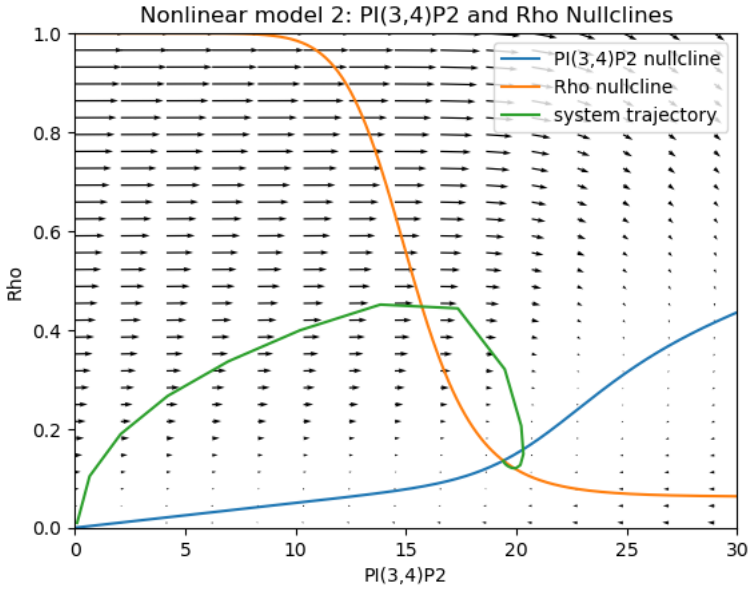


Figure 14: Phase plane, nullclines of PI(3,4)P_2 and Rho, vector field and system trajectory for nonlinear model 2, obtained using Python.

For nonlinear model 2, the phase plane of PI(3,4)P_2 (B) and Rho (R*) was plotted (Figure 14 and B.4), using the following nullclines equations, where $m = \frac{\alpha_A}{k_{AB}+\delta_A}$:

B-nullcline: $R^* = \frac{\delta_B B}{k_{AB} m - k_{deg} m \frac{B^n}{K^n+B^n}}$

R-nullcline: $R^* = \frac{\beta}{\beta+\gamma \frac{B^n}{K^n+B^n}}$

As with nonlinear model 1, the nullclines intersect at a single stable fixed point. The trajectory spirals toward this point without forming a closed loop, confirming the presence of a stable equilibrium and the absence of limit cycle oscillations.

Nonlinear model 3

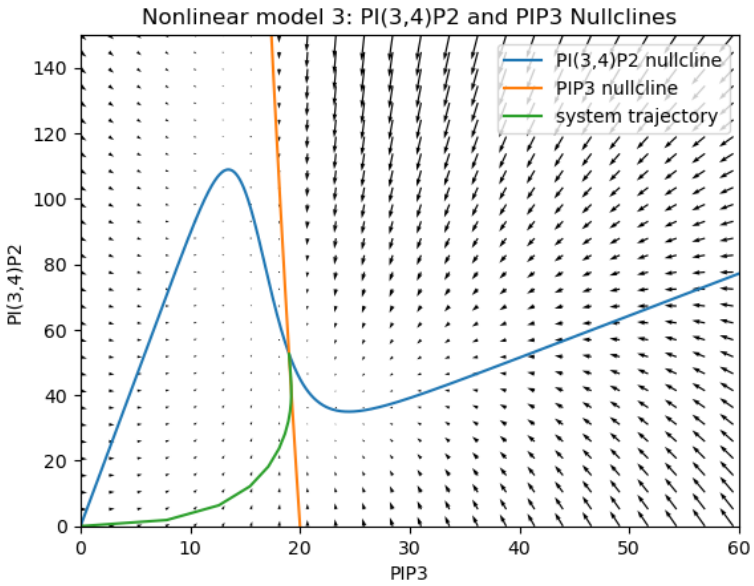


Figure 15: Phase plane, nullclines of PIP_3 and $PI(3,4)P_2$, vector field and system trajectory for nonlinear model 3, obtained using Python.

In this case, the phase plane defined by PIP_3 (A) and $PI(3,4)P_2$ (B) (Figure 15 and B.5) was analyzed considering Rho at its steady state value: $R^* = \frac{m}{A}$, where $m = \frac{\alpha_A}{k_{AB}+\delta_A}$. The nullclines used are:

A-nullcline: $B = \frac{\beta}{\gamma\left(\frac{m}{A}-1\right)}$

B-nullcline: $B = \frac{k_{AB}A}{k_{deg}\frac{A^n}{K^n+A^n}+\delta_B}$

As observed in the previous models, the intersection of the two nullclines identifies the system's unique fixed point. The system trajectory converges toward this point and no oscillatory loops are observed. The direction field also shows all arrows pointing inward, confirming the system's convergence toward a stable equilibrium.

In conclusion, none of the three nonlinear models exhibited the emergence of a stable limit cycle. In all cases, system trajectories converged toward a stable fixed point, indicating that, despite the introduction of nonlinear Hill-type terms, the system was unable to produce sustained oscillations observed experimentally. However, it is important to note that phase plane and nullcline analysis is only a qualitative method. It is useful for visualizing the dynamics of the system for a specific set of parameters, but it does not allow for a

comprehensive analysis across a broader parameter range. Therefore, these results should be complemented by more precise mathematical tools, such as the Routh-Hurwitz criterion, which will be used in the following section to assess the stability and potential for bifurcation of the various proposed formulations.

3.3.2 Routh–Hurwitz stability criterion

This paragraph presents the mathematical investigation conducted to determine whether a Hopf bifurcation can occur in one of the proposed models. A Hopf bifurcation is a phenomenon in which a small change in a parameter causes the system's behavior to transition from a stable steady state to a stable limit cycle with oscillatory behavior. The condition for a Hopf bifurcation to occur is that the Jacobian matrix of the system has a pair of purely imaginary eigenvalues:

$$\lambda_{\pm} = \pm i\omega$$

The general Jacobian matrix describing a negative feedback loop is expressed as:

$$\boldsymbol{J}_{-} = \begin{bmatrix} -\alpha & 0 & -\phi \\ c_1 & -\beta & 0 \\ 0 & c_2 & -\gamma \end{bmatrix}$$

where all the constants $\alpha, \beta, \gamma, \phi, c_1, c_2$ are positive. This structure correponds to the Jacobian matrix obtained from the proposed systems of ordinary differential equations. For instance, considering the initial linear model, the Jacobian takes the form:

$$\boldsymbol{J}_{-} = \begin{bmatrix} -\beta - \gamma B & 0 & -\gamma R^{*} \\ \alpha_A & -k_{AB} - \delta_A & 0 \\ 0 & k_{AB} - k_{deg} & -k_{deg} A - \delta_B \end{bmatrix}$$

The eigenvalues of J- are the roots of the characteristic equation, which is:

$$G(\lambda) = \lambda^3 + (\alpha + \beta + \gamma)\lambda^2 + (\alpha\beta + \beta\gamma + \gamma\alpha)\lambda + \alpha\beta\gamma + c_1 c_2 \phi = 0$$
$$= \lambda^3 + A\lambda^2 + B\lambda + C = 0$$

The Routh-Hurwitz criterion can be used to characterize the roots of this equation, hence determine the stability of the system. The criterion states that letting $G(\lambda_i) = 0$ for $i = 1, 2, 3,$

then $Re(\lambda_i) < 0$ for $i = 1, 2, 3$ (i.e. the system is stable) if and only if (i) $A > 0$, (ii) $C > 0$, and (iii) $AB > C$. Conversely, the Routh-Hurwitz criterion implies that the steady state is unstable if and only if:

$$AB < C$$
$$(\alpha + \beta + \gamma)(\alpha\beta + \beta\gamma + \gamma\alpha) < \alpha\beta\gamma + c_1 c_2 \phi$$

If equality holds, then $G(\lambda) = (\lambda + A)(\lambda^2 + B)$. In this case, the characteristic equation has conjugate roots on the imaginary axis at $\lambda = \pm i\sqrt{\alpha\beta + \beta\gamma + \gamma\alpha}$, which means the eigenvalues are purely imaginative and an Hopf bifurcation takes place [7].

To conduct the stability analysis of the the various proposed models, a Python script was implemented to plot the behavior of the function:

$$f(x) = AB - C$$

where the independent variable x represents a tunable system parameter. The scripts implemented for each model are provided in Appendix A (Figures B.6–B.9). The analysis were conducted varying α_A which denotes the rate at which active Rho promotes the conversion of PI(4,5)P_2 to PIP_3. According to the Routh-Hurwitz criterion, if $f(x) < 0$, then $AB < C$, indicating that a Hopf bifurcation occurs and the system is capable of exhibiting oscillatory dynamics.

Linear model

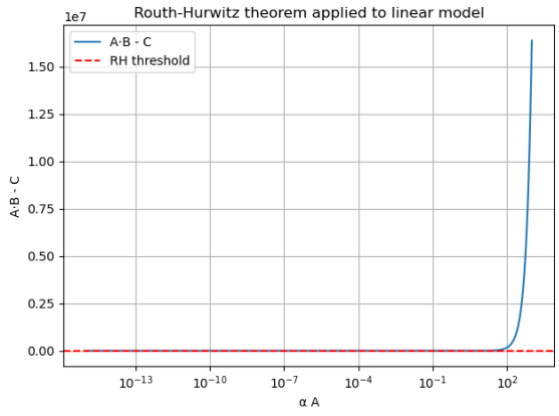
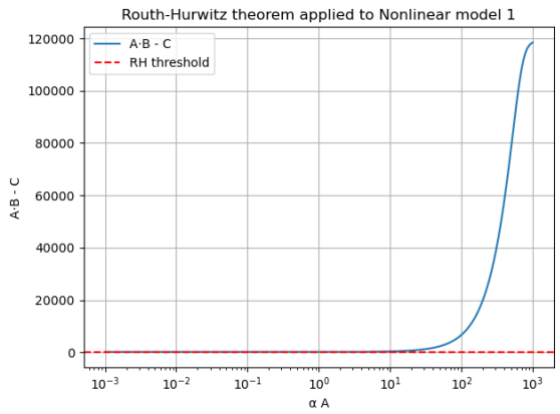


Figure 16: Plot of f(α)=AB-C regarding the linear model.

Analyzing the function $f(\alpha_A) = AB - C$ for the initial linear model, it can be observed that it is asymptotic to zero and it never crosses the Routh-Hurwitz threshold across the range $\alpha_A \in [10^{-15}, 10^{3}]$ (Figure 16). This confirms the prior statement: the linear model cannot sustain oscillations and always converges to a stable equilibrium.

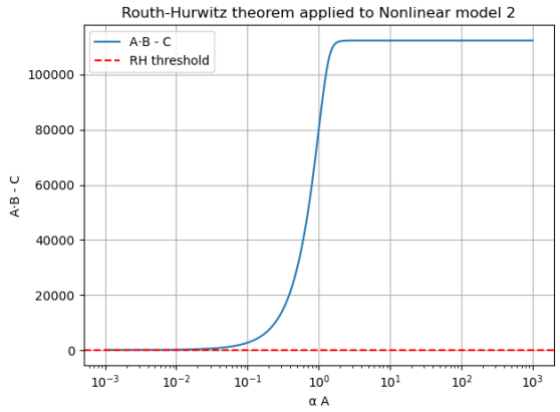
Nonlinear model 1



Applying the same procedure to nonlinear model 1, the function $f(\alpha_A) = AB - C$ again remained strictly positive and asymptotic, never crossing the Routh-Hurwitz threshold (Figure 17). Thus, this version of the model also fails to undergo a Hopf bifurcation and exhibits only steady-state behavior.

Figure 17: Plot of f(α)=AB-C regarding nonlinear model 1.

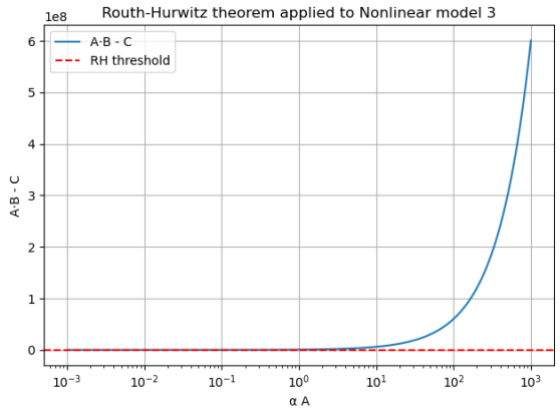
Nonlinear model 2



Similarly, in nonlinear model 2, the function $f(\alpha_A)$ showed asymptotic behavior without ever becoming negative (Figure 18). This indicates that, like the previous formulations, this model is also incapable of producing sustained oscillations within the tested parameter range.

Figure 18: Plot of f(α)=AB-C regarding nonlinear model 2.

Nonlinear model 3



Finally, the same analysis was applied to the third nonlinear formulation. Once again, the function $f(\alpha_A)$ did not cross the Routh-Hurwitz threshold (Figure 19). Therefore, not even this formulation, was able to reproduce the oscillatory behavior observed experimentally in the phospho-inositide–Rho GTPase signaling network.

Figure 19: Plot of f(α)=AB-C regarding nonlinear model 3.

None of the proposed models satisfied the condition AB > C required for a Hopf bifurcation to occur. Therefore, the ordinary differential equations derived from experimental observations fail to adequately capture the oscillatory behavior observed in reality.

4. Conclusions and critical analysis of model failure

In this thesis, the phosphoinositide-Rho GTPase signaling network was analyzed in order to develop a model capable of reproducing the Rho mixed-mode oscillations observed experimentally in nocodazole-treated mitotic rat cells. Starting from the model proposed in *"Periodicity, mixed-mode oscillations, and multiple timescales in a phosphoinositide-Rho GTPase network,"* by Chee San Tong, X.J. Xŭ, and Min Wu, it was shown that the incoherent feedforward loop (IFFL) they proposed could not reproduce sustained fast oscillations. Therefore, additional feedback mechanisms, based on experimental observations, and nonlinear terms were introduced to enable the oscillatory behavior.

Four models were proposed: a linear model and three nonlinear variants (nonlinear model 1, 2 and 3), each one presenting nonlinearities in different parts of the system of ordinary differential equations (ODEs). Despite these modifications, none of the proposed models were able to generate a stable limit cycle, which is an unstable fixed point that corresponds to sustained oscillations. Both phase plane analysis and Routh–Hurwitz stability criterion confirmed that all the proposed models converge to a stable fixed point, failing to produce periodic behavior.

These results suggest that the simplified circuit explored in this study is insufficient to explain the complex oscillatory dynamics observed in cells. One reason could be that the proposed models are too minimal and do not include important biological features. For example, in real cells, some reactions take time to happen and these delays can be essential to generate oscillations. Moreover, the models only describe the average concentrations of molecules over time without considering their distribution in space, yet, molecules like Rho and phosphoinositides often work in specific areas, such as the membrane of the cell. In addition, the models are fully deterministic, meaning they do not include any randomness, while in real biological systems, random fluctuations can influence how molecules behave and sometimes trigger oscillations. None of these aspects were included in the simplified ordinary differential equations used to model the signaling network.

The negative results of this study highlight the limitations of using simplified ordinary differential equations to capture the behavior of a complex intracellular network. Although the proposed models incorporate realistic interactions and nonlinearities, their simplicity may obscure essential mechanisms responsible for the mixed-mode oscillations observed in vivo.

To advance the understanding of this system, future work should explore models that include time delays or introduce additional molecular species beyond the three variables used in this study. It could also be useful to include spatial aspects of the cell or simulate how stochastic

fluctuations affect the system. Finally, more experimental data would help refine the models by confirming which molecular interactions are most important and their kinetic parameters.
Such efforts could guide to the development of more accurate representations of the biochemical circuits regulating cell contractility and help elucidate the nonlinear mechanisms driving Rho oscillations.

5. Bibliography

[1] C. S. Tong, X. J. Xǔ, and M. Wu, "Periodicity, mixed-mode oscillations, and multiple timescales in a phosphoinositide-Rho GTPase network," *Cell Reports*, vol. 42, no. 8, Art. no. 112857, Aug. 2023. doi: 10.1016/j.celrep.2023.112857.

[2] Wikipedia contributors, "Cytokinesis," *Wikipedia, The Free Encyclopedia*, https://en.wikipedia.org/wiki/Cytokinesis (accessed Jun. 29, 2025).

[3] G. M. Odell, "A mathematically modelled cytogel cortex exhibits periodic Ca^{2+}-modulated contraction cycles seen in *Physarum* shuttle streaming," *J. Embryol. Exp. Morphol.,* vol. 83, Suppl., pp. 261–287, 1984. doi: 10.1242/dev.83.Supplement.261.

[4] M. Graessl, J. Koch, A. Calderon, D. Kamps, S. Banerjee, T. Mazel, N. Schulze, J. K. Jungkurth, R. Patwardhan, D. Solouk, N. Hampe, B. Hoffmann, L. Dehmelt, and P. Nalbant, "An excitable Rho GTPase signaling network generates dynamic subcellular contraction patterns," *J. Cell Biol.*, vol. 216, no. 12, pp. 4271–4285, Dec. 2017. doi: 10.1083/jcb.201706052.

[5] G. Di Paolo and P. De Camilli, "Phosphoinositides in cell regulation and membrane dynamics," *Nature*, vol. 443, pp. 651–657, Oct. 2006. doi: 10.1038/nature05185.

[6] S. Y. S. Fung, X. J. Xǔ, and M. Wu, "Nonlinear dynamics in phosphoinositide metabolism," *Curr. Opin. Cell Biol.*, vol. 88, Art. no. 102373, May 2024. doi: 10.1016/j.ceb.2024.102373.

[7] C. J. Paton, J. J. Tyson, and J. C. Neu, "Biochemical oscillations and three-component networks without autocatalysis," in *Computational Cell Biology*, C. Fall, E. Marland, J. Wagner, and J. Tyson, Eds. New York, NY: Springer, 2002, ch. 9, pp. 243–247.

6. Figure sources

Figures 1, 2, 3, 4, 5, 6, and 10 were adapted from: C. S. Tong, X. J. Xǔ, and M. Wu, "Periodicity, mixed-mode oscillations, and multiple timescales in a phosphoinositide-Rho GTPase network," *Cell Reports*, vol. 42, no. 8, Art. no. 112857, Aug. 2023. doi: 10.1016/j.celrep.2023.112857

Figures 7, 12, 13, 14, 15, 16, 17, 18, and 19 were produced by the author using Python scripts as described in the corresponding sections and included in Appendix B.

Figures 8, 9, and 11 were created by the author using BioRender.com.

7. Appendix A – Calculations to implement Routh-Hurwitz criterion

The following calculations aim to determine the coefficients A, B and C required to apply the Routh-Hurwitz stability criterion to each proposed model.

A.1 Linear model

The Jacobian matrix obtained from the set of ODEs describing the linear model is:

$$\boldsymbol{J}_{-} = \begin{bmatrix} -\alpha & 0 & -\phi \\ c_1 & -\beta & 0 \\ 0 & c_2 & -\gamma \end{bmatrix} = \begin{bmatrix} -\beta - \gamma B & 0 & -\gamma R^* \\ \alpha_A & -k_{AB} - \delta_A & 0 \\ 0 & k_{AB} - k_{deg} B & -k_{deg} A - \delta_B \end{bmatrix}$$

From the characteristic equation $G(\lambda)$, the coefficients A, B and C can be derived as:

$$A = \alpha + \beta + \gamma = \beta + \gamma B_0 + k_{AB} + \delta_A + k_{deg} A_0 + \delta_B$$

$$\begin{aligned} B = \alpha\beta + \beta\gamma + \gamma\alpha \\ = (k_{AB} + \delta_A)(\beta + \gamma B_0) + (k_{AB} + \delta_A)\left(k_{deg} A_0 + \delta_B\right) \\ + \left(k_{deg} A_0 + \delta_B\right)(\beta + \gamma B_0) \end{aligned}$$

$$C = \alpha\beta\gamma + c_1 c_2 \phi = (k_{AB} + \delta_A)\left(k_{deg} A_0 + \delta_B\right)(\beta + \gamma B_0) + \alpha_A \gamma {R^*}_0 \left(k_{AB} - k_{deg} B_0\right)$$

The equilibrium concentration A_0, B_0 and ${R^*}_0$ used in the coefficients can be found solving the system of ODEs at steady state:

$$\begin{cases} \dfrac{dA}{dt} = \alpha_A R^* - k_{AB} A_0 - \delta_A A_0 = 0 \\ \dfrac{dB}{dt} = k_{AB} A_0 - k_{deg} A_0 B_0 - \delta_B B_0 = 0 \\ \dfrac{dR^*}{dt} = \beta(1 - {R^*}_0) - \gamma B_0 {R^*}_0 = 0 \end{cases}$$

From the above, the following quadratic equation was derived and implemented in the Python script (Figure B.6) to find B_0 at equilibrium:

$$\gamma\delta_B {B_0}^2 + \beta\left(k_{deg}m + \delta_B\right)B_0 - k_{AB}m\beta = 0$$

Where:

$$a = \gamma\delta_B, \quad b = \beta\left(k_{deg}m + \delta_B\right), \quad c = -k_{AB}m\beta$$

B_0 was then used to find A_0 and ${R^*}_0$, using the following equations:

$${R^*}_0 = \frac{\delta_B B_0}{k_{AB}m - k_{deg}mB_0}$$

$A_0 = m{R^*}_0$, where $m = \frac{\alpha_A}{k_{AB}+\delta_A}$.

A.2 Nonlinear model 1

The Jacobian matrix obtained from the set of ODEs describing nonlinear model 1 is:

$$\boldsymbol{J}_- = \begin{bmatrix} -\alpha & 0 & -\phi \\ c_1 & -\beta & 0 \\ 0 & c_2 & -\gamma \end{bmatrix} = \begin{bmatrix} -\beta - \gamma\frac{B^n}{K^n + B^n} & 0 & -\gamma R^* \frac{nK^nB^{n-1}}{(K^n + B^n)^2} \\ \alpha_A & -k_{AB} - \delta_A & 0 \\ 0 & k_{AB} - k_{deg}B & -k_{deg}A - \delta_B \end{bmatrix}$$

From the characteristic equation $G(\lambda)$, the coefficients A, B and C can be derived as:

$$A = \alpha + \beta + \gamma = \beta + \gamma\frac{{B_0}^n}{K^n + {B_0}^n} + k_{AB} + \delta_A + k_{deg}A_0 + \delta_B$$

$$\begin{aligned} B &= \alpha\beta + \beta\gamma + \gamma\alpha \\ &= (k_{AB} + \delta_A)\left(\beta + \gamma\frac{{B_0}^n}{K^n + {B_0}^n}\right) + (k_{AB} + \delta_A)\left(k_{deg}A_0 + \delta_B\right) \\ &\quad + \left(k_{deg}A_0 + \delta_B\right)\left(\beta + \gamma\frac{{B_0}^n}{K^n + {B_0}^n}\right) \end{aligned}$$

$$C = \alpha\beta\gamma + c_1 c_2 \phi$$

$$= (k_{AB} + \delta_A)(k_{deg} A_0 + \delta_B)\left(\beta + \gamma \frac{{B_0}^n}{K^n + {B_0}^n}\right)$$

$$+ \alpha_A \gamma {R^*}_0 \frac{n K^n {B_0}^{n-1}}{(K^n + {B_0}^n)^2}(k_{AB} - k_{deg} B_0)$$

The equilibrium concentration A_0, B_0 and ${R^*}_0$ used in the coefficients can be found solving the system of ODEs at steady state:

$$\begin{cases} \dfrac{dA}{dt} = \alpha_A R^* - k_{AB} A_0 - \delta_A A_0 = 0 \\ \dfrac{dB}{dt} = k_{AB} A_0 - k_{deg} A_0 B_0 - \delta_B B_0 = 0 \\ \dfrac{dR^*}{dt} = \beta(1 - {R^*}_0) - \gamma \dfrac{{B_0}^n}{K^n + {B_0}^n} {R^*}_0 = 0 \end{cases}$$

From the above, the following equations were derived and implemented in the Python script (Figure B.7) to be solved numerically and find B_0 and ${R^*}_0$ at equilibrium:

$${R^*}_0 = \frac{\beta}{\beta + \gamma \dfrac{{B_0}^n}{K^n + {B_0}^n}}$$

$$B_0 = \frac{m {R^*}_0 k_{AB}}{\delta_B + m {R^*}_0 k_{deg}}$$

A_0 was then found using the following equation:

$A_0 = m{R^*}_0$, where $m = \frac{\alpha_A}{k_{AB} + \delta_A}$.

A.3 Nonlinear model 2

The Jacobian matrix obtained from the set of ODEs describing the nonlinear model 2 is:

$$\boldsymbol{J}_{-} = \begin{bmatrix} -\alpha & 0 & -\phi \\ c_1 & -\beta & 0 \\ 0 & c_2 & -\gamma \end{bmatrix}$$

$$= \begin{bmatrix} -\beta - \gamma \dfrac{B^n}{K^n + B^n} & 0 & -\gamma R^* \dfrac{nK^n B^{n-1}}{(K^n + B^n)^2} \\ \alpha_A & -k_{AB} - \delta_A & 0 \\ 0 & k_{AB} - k_{deg} \dfrac{B^n}{K^n + B^n} & -k_{deg} A \dfrac{nK^n B^{n-1}}{(K^n + B^n)^2} - \delta_B \end{bmatrix}$$

From the characteristic equation $G(\lambda)$, the coefficients A, B and C can be derived as:

$$A = \alpha + \beta + \gamma = \beta + \gamma \frac{{B_0}^n}{K^n + {B_0}^n} + k_{AB} + \delta_A + k_{deg} A_0 \frac{nK^n {B_0}^{n-1}}{(K^n + {B_0}^n)^2} + \delta_B$$

$$B = \alpha\beta + \beta\gamma + \gamma\alpha$$

$$= (k_{AB} + \delta_A)\left(\beta + \gamma \frac{{B_0}^n}{K^n + {B_0}^n}\right) + (k_{AB} + \delta_A)\left(k_{deg} A_0 \frac{nK^n {B_0}^{n-1}}{(K^n + {B_0}^n)^2} + \delta_B\right)$$

$$+ \left(k_{deg} A_0 \frac{nK^n {B_0}^{n-1}}{(K^n + {B_0}^n)^2} + \delta_B\right)\left(\beta + \gamma \frac{{B_0}^n}{K^n + {B_0}^n}\right)$$

$$C = \alpha\beta\gamma + c_1 c_2 \phi$$

$$= (k_{AB} + \delta_A)\left(k_{deg} A_0 \frac{nK^n {B_0}^{n-1}}{(K^n + {B_0}^n)^2} + \delta_B\right)\left(\beta + \gamma \frac{{B_0}^n}{K^n + {B_0}^n}\right)$$

$$+ \alpha_A \gamma {R^*}_0 \frac{nK^n {B_0}^{n-1}}{(K^n + {B_0}^n)^2}\left(k_{AB} - k_{deg} \frac{{B_0}^n}{K^n + {B_0}^n}\right)$$

The equilibrium concentration A_0, B_0 and ${R^*}_0$ used in the coefficients can be found solving the system of ODEs at steady state:

$$\begin{cases} \dfrac{dA}{dt} = \alpha_A R^* - k_{AB} A_0 - \delta_A A_0 = 0 \\ \dfrac{dB}{dt} = k_{AB} A_0 - k_{deg} A_0 \dfrac{{B_0}^n}{K^n + {B_0}^n} - \delta_B B_0 = 0 \\ \dfrac{dR^*}{dt} = \beta(1 - {R^*}_0) - \gamma \dfrac{{B_0}^n}{K^n + {B_0}^n} {R^*}_0 = 0 \end{cases}$$

From the above, the following equations were derived and implemented in the Python script (Figure B.8) to be solved numerically and find B_0 and ${R^*}_0$ at equilibrium:

$$R^*_0 = \frac{\beta}{\beta + \gamma \frac{{B_0}^n}{K^n + {B_0}^n}}$$

$$m{R^*}_0 k_{AB} - m{R^*}_0 k_{deg} \frac{{B_0}^n}{K^n + {B_0}^n} - \delta_B B_0 = 0$$

A_0 was then found using the following equation:

$A_0 = m{R^*}_0$, where $m = \frac{\alpha_A}{k_{AB}+\delta_A}$.

A.4 Nonlinear model 3

The Jacobian matrix obtained from the set of ODEs describing the nonlinear model 3 is:

$$\boldsymbol{J}_- = \begin{bmatrix} -\alpha & 0 & -\phi \\ c_1 & -\beta & 0 \\ 0 & c_2 & -\gamma \end{bmatrix} = \begin{bmatrix} -\beta - \gamma B & 0 & -\gamma R^* \\ \alpha_A & -k_{AB} - \delta_A & 0 \\ 0 & k_{AB} - k_{deg} B \frac{nK^n A^{n-1}}{(K^n + A^n)^2} & -k_{deg} \frac{A^n}{K^n + A^n} - \delta_B \end{bmatrix}$$

From the characteristic equation $G(\lambda)$, the coefficients A, B and C can be derived as:

$$A = \alpha + \beta + \gamma = \beta + \gamma B_0 + k_{AB} + \delta_A + k_{deg} \frac{{A_0}^n}{K^n + {A_0}^n} + \delta_B$$

$$B = \alpha\beta + \beta\gamma + \gamma\alpha$$
$$= (k_{AB} + \delta_A)(\beta + \gamma B_0) + (k_{AB} + \delta_A)\left(k_{deg} \frac{{A_0}^n}{K^n + {A_0}^n} + \delta_B\right)$$
$$+ \left(k_{deg} \frac{{A_0}^n}{K^n + {A_0}^n} + \delta_B\right)(\beta + \gamma B_0)$$

$$C = \alpha\beta\gamma + c_1 c_2 \phi$$
$$= (k_{AB} + \delta_A)\left(k_{deg} \frac{{A_0}^n}{K^n + {A_0}^n} + \delta_B\right)(\beta + \gamma B_0)$$
$$+ \alpha_A \gamma {R^*}_0 \left(k_{AB} - k_{deg} B_0 \frac{nK^n {A_0}^{n-1}}{(K^n + {A_0}^n)^2}\right)$$

The equilibrium concentration A_0, B_0 and ${R^*}_0$ used in the coefficients can be found solving the system of ODEs at steady state:

$$\begin{cases} \dfrac{dA}{dt} = \alpha_A R^* - k_{AB} A_0 - \delta_A A_0 = 0 \\ \dfrac{dB}{dt} = k_{AB} A_0 - k_{deg} \dfrac{{A_0}^n}{K^n + {A_0}^n} B_0 - \delta_B B_0 = 0 \\ \dfrac{dR^*}{dt} = \beta(1 - {R^*}_0) - \gamma B_0 {R^*}_0 = 0 \end{cases}$$

From the above, the following equations were derived and implemented in the Python script (Figure B.9) to be solved numerically and find B_0 and ${R^*}_0$ at equilibrium:

$${R^*}_0 = \frac{\beta}{\beta + \gamma \dfrac{{B_0}^n}{K^n + {B_0}^n}}$$

$$B_0 = \frac{m {R^*}_0 k_{AB}}{k_{deg} \dfrac{(m {R^*}_0)^n}{K^n + (m {R^*}_0)^n} + \delta_B}$$

A_0 was then found using the following equations:

$A_0 = m {R^*}_0$, where $m = \frac{\alpha_A}{k_{AB} + \delta_A}$.

8. Appendix B – Python code

Figure B.1: Python script simulating the behavior of the incoherent feedforward loop proposed by Tong et al. (corresponding to Figure 7).

```python
# Incoherent Feedfoward Loop Simulation

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

# X -> PIP3
# Y -> PI(3)P
# Z -> PI(3,4)P2

# Parameters
alpha_X = 1.0
delta_X = 0.5

beta_Y = 2.0
K_Y = 1.7
n = 10
delta_Y = 0.5

gamma_Z = 2.0
K_Y = 0.5
m = 5
delta_Z = 0.5

# ODE
def iffl(t, y):
    X, Y, Z = y
    dXdt = alpha_X - delta_X * X
    dYdt = (beta_Y * X**n) / (K_Y**n + X**n) - delta_Y * Y
    dZdt = gamma_Z * X * 1 / (1 + (Y / K_Y)**m) - delta_Z * Z
    return [dXdt, dYdt, dZdt]

# Initial conditions
X0, Y0, Z0 = 0.0, 0.0, 0.0

# Simulation
t_eval = np.linspace(0, 20, 1000)
sol = solve_ivp(iffl, [0, 20], [X0, Y0, Z0], t_eval=t_eval)

# Plots
fig, axs = plt.subplots(3, 1, figsize=(4, 5), sharex=True)

axs[0].plot(sol.t, sol.y[0])
axs[0].set_ylabel("PIP3 (X)")

axs[1].plot(sol.t, sol.y[1])
axs[1].set_ylabel("PI(3)P (Y)")

axs[2].plot(sol.t, sol.y[2])
axs[2].set_xlabel("Time")
axs[2].set_ylabel("PI(3,4)P2 (Z)")

plt.suptitle("Incoherent Feedforward Loop", fontsize=14)
plt.tight_layout(rect=[0, 0, 1, 0.97])
plt.show()
```

Figure B.2: Python script simulating the behavior of the the linear model in time (corresponding to Figure 12).

```python
# Linear Model Simulation

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

# Parameters
αA = 1
kAB = 1
δA = 0.1
kdeg = 10
δB = 0.1
β = 1
γ = 50

# ODEs
def iffl(t, y):
    A, B, R = y
    dA_dt = αA * R - kAB * A
    dB_dt = kAB * A - kdeg * A * B
    dR_dt = β * (1 - R) - γ * B * R
    return [dA_dt, dB_dt, dR_dt]

# Initial conditions
A0, B0, R0 = 0.0, 0.0, 0.0

# Simulation
t_eval = np.linspace(0, 20, 1000)
sol = solve_ivp(iffl, [0, 20], [A0, B0, R0], t_eval=t_eval)

# Plot
plt.figure()
plt.plot(sol.t, sol.y[0], label="PIP3 (A)")
plt.plot(sol.t, sol.y[1], label="PI(3,4)P2  (B)")
plt.plot(sol.t, sol.y[2], label="Rho (R*)")
plt.legend(frameon=False)
plt.xlabel("Time")
plt.ylabel("Concentration")
plt.title("Proposed fast oscillation circuit")
plt.tight_layout()
plt.show()
```

Figure B.3: Python script to visualize the phase plane, nullclines, system trajectory and vector field of nonlinear model 1 (corresponding to Figure 13).

```python
# Phase plane and Nullclines of Nonlinear Model 1

import numpy as np
import matplotlib.pyplot as plt
from scipy.integrate import odeint

# Parameters
α_A = 10
kAB = 10
kdeg = 0.05
δA = 0.01
δB = 0.5
β = 1
γ = 100
K = 10
n = 10
m = α_A / (kAB + δA)

# B axis
B = np.arange(0.1, 30, 0.1)

# B nullcline
R1 = (δB * B) / (kAB * m - kdeg * m * B)

# R nullcline
R2 =  β / (β + γ * (B**n / (K**n + B**n)))

# Grid
B_vals = np.linspace(0.01, 30, 40)
R_vals = np.linspace(0.01, 1.0, 40)
b, r = np.meshgrid(B_vals, R_vals)

# ODEs
def f(BR, t):
    B, R = BR
    dBdt = kAB * m * R - kdeg * m * R * B - δB * B
    dRdt = β * (1 - R) - γ * (B**n / (K**n + B**n)) * R
    return [dBdt, dRdt]

# Vector field
U = f([b, r], t = 0)[0]
V = f([b, r], t = 0)[1]

# Time
t = np.arange(0, 1000, 0.1)

# Initial conditions
xy0 = [0.1,0.1]

# Trajectory
xy1 = odeint(f, xy0, t)

# Plot
plt.plot(B, R1, label = "PI(3,4)P2 nullcline")
plt.plot(B, R2, label = "Rho nullcline")
plt.plot(xy1[:,0], xy1[:,1], label = "system trajectory")
plt.title("Nonlinear model 1: PI(3,4)P2 and Rho Nullclines")
plt.legend()
plt.xlabel("PI(3,4)P2")
plt.ylabel("Rho")
plt.ylim(0, 1)
plt.xlim(0, 30)
plt.quiver(b, r, U, V)
plt.show()
```

Figure B.4: Python script to visualize the phase plane, nullclines, system trajectory and vector field of nonlinear model 2 (corresponding to Figure 14).

```
# Phase plane and Nullclines of Nonlinear Model 2

import numpy as np
import matplotlib.pyplot as plt
from scipy.integrate import odeint

# Parameters
αA = 100
kAB = 4.5
kdeg = 3
δA = 0.005
δB = 0.5
β = 1
γ = 15
K = 20
n = 10
m = αA / (kAB + δA)

# B axis
B = np.linspace(0.1, 60, 1000)

# B nullcline
R1 = (δB * B) / (kAB * m - kdeg * m * (B**n / (K**n + B**n)))

# R nullcline
R2 =  β / (β + γ * (B**n / (K**n + B**n)))

# Grid
B_vals = np.linspace(0.01, 60, 30)
R_vals = np.linspace(0.01, 1.0, 30)
b, r = np.meshgrid(B_vals, R_vals)

# ODEs
def f(BR, t):
    B, R = BR
    dBdt = kAB * m * R - kdeg * m * R * (B**n / (K**n + B**n)) - δB * B
    dRdt = β * (1 - R) - γ * (B**n / (K**n + B**n)) * R
    return [dBdt, dRdt]

# Vector field
U = f([b, r], t = 0)[0]
V = f([b, r], t = 0)[1]

# Time
t = np.arange(0, 1000, 0.1)

# Initial condition
xy0 = [0.1,0.01]

# Trajectory
xy1 = odeint(f, xy0, t)

# Plot
plt.plot(B, R1, label = "PI(3,4)P2 nullcline")
plt.plot(B, R2, label = "Rho nullcline")
plt.plot(xy1[:,0], xy1[:,1], label = "system trajectory")
plt.title("Nonlinear model 2: PI(3,4)P2 and Rho Nullclines")
plt.legend()
plt.xlabel("PI(3,4)P2")
plt.ylabel("Rho")
plt.ylim(0, 1)
plt.xlim(0, 30)
plt.quiver(b, r, U, V)
plt.show()
```

Figure B.5: Python script to visualize the phase plane, nullclines, system trajectory and vector field of nonlinear model 3 (corresponding to Figure 15).

```
# Phase Plane and Nullclines of Nonlinear Model 3

import numpy as np
import matplotlib.pyplot as plt
from scipy.integrate import odeint

# Parameters
αA = 100
kAB = 4.5
kdeg = 3
δA = 0.5
δB = 0.5
β = 100
γ = 0.1
K = 20
n = 10
m = αA / (kAB + δA)

# A axis
A = np.linspace(0.1, 60, 1000)

# B nullcline
B1 = (kAB * A) / (kdeg * (A**n / (K**n + A**n)) + δB)

# A nullcline
B2 =  β / γ * (m/A - 1)

# Grid
B_vals = np.linspace(0, 150, 30)
A_vals = np.linspace(0, 75, 30)
b, a = np.meshgrid(B_vals, A_vals)

# ODEs
def f(AB, t):
    A, B = AB
    dBdt = kAB * A - kdeg * B * (A**n / (K**n + A**n)) - δB * B
    dAdt = αA * β / (β + γ * B) - kAB * A - δA * A
    return [dAdt, dBdt]

# Vector field
U = f([a, b], t = 0)[0]
V = f([a, b], t = 0)[1]

# Time
t = np.arange(0, 1000, 0.1)

# Initial conditions
xy0 = [0, 0]

# Trajectory
xy1 = odeint(f, xy0, t)

# Plot
plt.plot(A, B1, label = "PI(3,4)P2 nullcline")
plt.plot(A, B2, label = "PIP3 nullcline")
plt.plot(xy1[:,0], xy1[:,1], label = "system trajectory")
plt.title("Nonlinear model 3: PI(3,4)P2 and PIP3 Nullclines")
plt.legend()
plt.xlabel("PIP3")
plt.ylabel("PI(3,4)P2")
plt.ylim(0,150)
plt.xlim(0,60)
plt.quiver(a, b, U, V)
plt.show()
```

Figure B.6: Python script to apply Routh-Hurwitz criterion to the linear model by plotting function f(x)=AB-C (corresponding to Figure 16).

```
# Routh-Hurwitz theorem applied to linear model

import matplotlib.pyplot as plt
import numpy as np

# Parameters
kdeg = 10
kAB = 1
δA = 0.1
β = 1
γ = 50
δB = 0.1

# Scan α A over a log-range
αA_vals = np.logspace(-15, 3, 600)
results = []
inequality_diff = []

# Concentrations of A, B and C at equilibrium: A0, B0, R0
for αA in αA_vals:
    m = αA/(kAB+δA) # A0 = m * R0

    # Quadratic equation of B0
    a = δB*γ
    b = β*(kdeg*m + δB)
    c = -kAB*m*β
    disc = b*b - 4*a*c # discriminant
    B0 = (-b + np.sqrt(disc)) / (2*a) # positive root, as B0 > 0

    # A0 and R0
    R0 = β/(β + γ*B0)
    A0 = m*R0

    # Coefficients to use Routh-Hurwitz theorem
    A = kAB + δA + kdeg*A0 + δB + β + γ*B0
    B = (kAB + δA) * (kdeg*A0 + δB) + (kAB+δA) * (β + γ*B0) + (kdeg*A0 + δB) * (β + γ*B0)
    C = (kAB + δA) * (kdeg*A0 + δB) * (β + γ*B0) + αA * γ*R0 * (kAB - kdeg*B0)

    if A*B < C:
        results.append(kdeg)

    inequality_diff.append(A * B - C)

if len(results) != 0:
    print("A·B ≤ C: ", results)
else:
    print("No α_A destabilizes equilibrium.")

# Plot
inequality_diff = np.array(inequality_diff)
plt.figure()
plt.plot(αA_vals, inequality_diff, label='A·B - C')
plt.axhline(0, color='red', linestyle='--', label='RH threshold')
plt.title("Routh-Hurwitz theorem applied to linear model")
plt.xscale('log')
plt.xlabel('α A')
plt.ylabel('A·B - C')
plt.legend()
plt.grid(True)
plt.tight_layout()
plt.show()
```

Figure B.7: Python script to apply Routh-Hurwitz criterion to nonlinear model 1 by plotting function f(x)=AB-C (corresponding to Figure 17).

```
# Routh-Hurwitz theorem applied to Nonlinear Model 1

import numpy as np
from scipy.optimize import fsolve
import matplotlib.pyplot as plt
import warnings

# Parameters
kAB = 10
kdeg = 0.05
δA = 0.01
δB = 0.5
β = 1
γ = 100
K = 10
n = 10

# scan αA over a log-range
αA_vals = np.logspace(-3, 3, 500)
results = []
inequality_diff = []

# Solve equilibrium equations numerically
guess = [0.5, 0.5]

for αA in α_A_vals:
    m = αA / (kAB + δA)

    # Define nullcline equations
    def equations(vars):
        R, B = vars
        eq1 = B - (m * R * kAB) / (δB + m * R * kdeg)
        hill = B**n / (K**n + B**n)
        eq2 = R - β / (β + γ * hill)
        return [eq1, eq2]

    # Check that equations are solved correctly: if a warning is given the loop ignores the
results given by fsolve
    with warnings.catch_warnings():
        warnings.simplefilter("ignore")
        sol, info, ier, msg = fsolve(equations, guess, full_output=True)

    # Skip αA value if solver failed
    if ier != 1:
        continue

    R0, B0 = sol

    # Skip solutions if invalid
    if R0 <= 0 or B0 <= 0:
        continue

    # Update guess for next step
    guess = [R0, B0]
    A0 = m * R0

    # Routh-Hurwitz terms
    hill = B0**n / (K**n + B0**n)
    der_hill = n * K**n * B0**(n - 1) / (K**n + B0**n)**2

    A = kAB + δA + kdeg * A0 + δB + β + γ * hill
    B = (kAB + δA) * (kdeg * A0 + δB) + (kAB + δA) * (β + γ * hill) + (kdeg * A0 + δB) * (β + γ
* hill)
    C = (kAB + δA) * (kdeg * A0 + δB) * (β + γ * hill) + αA * γ * R0 * der_hill * (kAB - kdeg *
B0)

    RH = A * B - C
    inequality_diff.append(RH)
```

```
    if RH < 0:
        results.append(αA)

if results:
    print("Routh-Hurwitz fails for α_A values:", results)
else:
    print("No α_A destabilizes equilibrium.")

# Plot
plt.figure()
plt.plot(αA_vals, inequality_diff, label='A·B - C')
plt.axhline(0, color='red', linestyle='--', label='RH threshold')
plt.title("Routh-Hurwitz theorem applied to Nonlinear model 1")
plt.xscale('log')
plt.xlabel('α A')
plt.ylabel('A·B - C')
plt.legend()
plt.grid(True)
plt.tight_layout()
plt.show()
```

Figure B.8: Python script to apply Routh-Hurwitz criterion to nonlinear model 2 by plotting function f(x)=AB-C (corresponding to Figure 18).

```
# Routh-Hurwitz theorem applied to Nonlinear Model 2

import numpy as np
from scipy.optimize import fsolve
import matplotlib.pyplot as plt
import warnings

# Parameters
kAB = 10
kdeg = 0.05
δA = 0.01
δB = 0.01
β = 1
γ = 100
K = 1
n = 10

# Scan αA over a log-range
αA_vals = np.logspace(-3, 3, 400)

results = []
inequality_diff = []

# Solve equilibrium equations numerically
guess = [0.5, 0.5]

for αA in αA_vals:
    m = αA / (kAB + δA)

    # Define nullcline equations
    def equations(vars):
        R, B = vars
        hill = B**n / (K**n + B**n)
        eq1 = m * R * kAB - m * R * kdeg * hill - δB * B
        eq2 = R - β / (β + γ * hill)
        return [eq1, eq2]

    # Check that equations are solved correctly: if a warning is given the loop ignores the
results given by fsolve
    with warnings.catch_warnings():
        warnings.simplefilter("ignore")
        sol, info, ier, msg = fsolve(equations, guess, full_output=True)

    # Skip this α_A if solver failed
    if ier != 1:
        continue

    # Skip solutions if invalid
    R0, B0 = sol
    if R0 <= 0 or B0 <= 0:
        continue

    # Update guess for next step
    guess = [R0, B0]
    A0 = m * R0

    # Routh-Hurwitz terms
    hill = B0**n / (K**n + B0**n)
    der_hill = n * K**n * B0**(n - 1) / (K**n + B0**n)**2

    A = kAB + δA + kdeg * A0 * der_hill + δB + β + γ * hill
    B = (kAB + δA) * (kdeg * A0 * der_hill + δB) + (kAB + δA) * (β + γ * hill) + (kdeg * A0 *
der_hill + δB) * (β + γ * hill)
    C = (kAB + δA) * (kdeg * A0 * der_hill + δB) * (β + γ * hill) + αA * γ * R0 * der_hill *
(kAB - kdeg * hill)
```

```
    RH = A * B - C
    inequality_diff.append(RH)

    if RH < 0:
        results.append(αA)

if results:
    print('Routh-Hurwitz fails for α_A values:')
    print(results)
else:
    print('No α_A destabilizes equilibrium.')

# Plot
plt.figure()
plt.plot(αA_vals, inequality_diff, label='A·B - C')
plt.axhline(0, color='red', linestyle='--', label='RH threshold')
plt.title('Routh-Hurwitz theorem applied to Nonlinear model 2')
plt.xscale('log')
plt.xlabel('α A')
plt.ylabel('A·B - C')
plt.legend()
plt.grid(True)
plt.tight_layout()
plt.show()
```

Figure B.9: Python script to apply Routh-Hurwitz criterion to nonlinear model 3 by plotting function f(x)=AB-C (corresponding to Figure 19).

```
# Routh-Hurwitz theorem applied to Nonlinear Model 3

import numpy as np
from scipy.optimize import fsolve
import matplotlib.pyplot as plt
import warnings

# Parameters
kAB   = 10
kdeg  = 0.02
δA    = 0.001
δB    = 0.001
β     = 0.1
γ     = 600
K     = 8
n     = 15

# Scan αA over a log-range
αA_vals = np.logspace(-3, 3, 1000)
results = []
inequality_diff = []

# Solve equilibrium equations numerically
guess = [0.5, 0.5]

for αA in αA_vals:
    m = αA / (kAB + δA)

    # Define nullcline equations
    def equations(vars):
        R, B = vars
        hill = (m * R)**n / (K**n + (m * R)**n)
        eq1 = m * R * kAB - hill * kdeg * B - δB * B
        eq2 = R - β / (β + γ * B)
        return [eq1, eq2]

    # Check that equations are solved correctly: if a warning is given the loop ignores the
results given by fsolve
    with warnings.catch_warnings():
        warnings.simplefilter("ignore")
        sol, info, ier, msg = fsolve(equations, guess, full_output=True)

    R0, B0 = sol

    # Skip  solution if solver failed or they are invalid (negative)
    if ier != 1 or R0 <= 0 or B0 <= 0:
        inequality_diff.append(np.nan)
        continue

    # Update guess for next step
    guess = [R0, B0]

    # Routh-hurwitz terms
    A0 = m * R0
    hill = A0**n / (K**n + A0**n)
    der_hill = n * K**n * A0**(n - 1) / (K**n + A0**n)**2

    A = kAB + δA + kdeg * hill + δB + β + γ * B0
    B = (kAB + δA) * (kdeg * hill + δB) + (kAB + δA) * (β + γ * B0) + (kdeg * hill + δB) * (β +
γ * B0)
    C = (kAB + δA) * (kdeg * hill + δB) * (β + γ * B0) + αA * γ * R0 * (kAB - kdeg * B0 *
der_hill)

    RH = A * B - C
    inequality_diff.append(RH)
```

```
    if RH < 0:
        results.append(αA)

if results:
    print("Routh-Hurwitz fails for α_A values:")
    print(results)
else:
    print("No α A destabilizes equilibrium.")

# Plot
plt.figure()
plt.plot(αA_vals, inequality_diff, label='A·B - C')
plt.axhline(0, color='red', linestyle='--', label='RH threshold')
plt.title('Routh-Hurwitz theorem applied to Nonlinear model 3')
plt.xscale('log')
plt.xlabel('α A')
plt.ylabel('A·B - C')
plt.legend()
plt.grid(True)
plt.tight_layout()
plt.show()
```