VORTEX 13.0

CODATHON ROUND 2

Date: 10-01-26
Duration: 4 hr

I : Cooperation, Conflict, and Conservation

Q1. Iterated Prisoner's Dilemma (IPD): Cooperation, Conflict, and Strategy in a Noisy World - 8 marks

Human societies, animal groups, and even nations constantly face a fundamental dilemma: **should I cooperate, or should I act purely in my own interest?** Cooperation can create long-term benefits, but it also exposes one to exploitation. This tension lies at the heart of economics, politics, biology, and everyday human interaction.

A few intuitive examples:

- **Nature (Biology):** Vampire bats share blood meals with others who failed to feed. Helping today reduces starvation risk tomorrow - but only if others return the favor. Cheaters threaten the system.
- **Geopolitics:** Arms treaties between rival nations promise mutual safety (cooperation), but any single defection - secret weapons, hidden violations - can give one side an advantage. The best example is the Nuclear Arms Deal after the Cold War.
- **Human interactions:** Group projects, friendships, traffic rules, or online communities all rely on cooperation. One person exploiting the system (free-riding, cheating) can collapse trust.

The central question is:

How can cooperation emerge and survive among self-interested agents, especially when mistakes and misunderstandings are unavoidable?

Game theory provides a clean, mathematical way to study this question, and **the Prisoner's Dilemma** is its most famous thought experiment.

The Prisoner's Dilemma is a two-player game where each player independently chooses one of two actions:

C: Cooperate **D**: Defect

The payoffs are defined as follows:

	Player 2: Cooperate	Player 2: Defect
Player 1: Coopera	R,R	S,T
Player 1: Defect	T,S	P,P

Where:

- **R (Reward)**: payoff for mutual cooperation
- **T (Temptation)**: payoff for defecting while the other cooperates
- **P (Punishment)**: payoff for mutual defection

- **S (Sucker’s payoff)**: payoff for cooperating while the other defects

These values must satisfy the inequality:
T > R > P > S
This ordering captures the dilemma:

- Defection is tempting in the short term.
- Mutual cooperation is better than mutual defection.
- Being exploited is the worst outcome.

In a *single-round* Prisoner’s Dilemma, rational self-interest always leads to defection - even though both players would be better off cooperating.
Real life is not a one-shot encounter.

- You meet the same people again.
- Countries interact over decades.
- Organisms evolve across generations.

The **Iterated Prisoner’s Dilemma (IPD)** repeats the same game many times between the same players. Each player can remember past actions and adapt their behavior accordingly.
Iteration introduces:

- **Reputation** (what did you do last time? - did you cheat someone)
- **Retaliation** (punishing defection)
- **Forgiveness** (recovering cooperation or just that you are non chalant)
- **Trust building** over time

In the 1970s, political scientist **Robert Axelrod** famously organized tournaments where different strategies competed in the IPD.
In theory, strategies execute perfectly. In reality, mistakes happen:

- A message is misunderstood.
- A signal is corrupted.
- A button is pressed incorrectly.

In our model, **action noise** represents such errors:
Even if a strategy chooses C, a random external disturbance may flip it to D (or vice versa).
This is not malicious - it is accidental.
Noise is crucial because:

- It tests robustness of cooperation
- It distinguishes **forgiving** strategies from **vindictive** ones
- It prevents unrealistic “perfect play” outcomes

In this simulation, action noise occurs with probability: **p ~ (5%)**
Strategies Under Consideration
Each strategy decides whether to Cooperate (C) or Defect (D) based on rules and memory.

1. **Neutral Strategies**
   Tit-for-Tat (TFT):
   - Start with C
   - Copy opponent’s last move

   Win–Stay, Lose–Shift (WSLS):
   - Repeat last move if payoff was R or T
   - Switch otherwise
2. **Altruistic Strategies**
   TFT with Forgiveness:
   - Like TFT, but sometimes forgives defection

   Always Cooperate (ALLC):
   - Always plays C
3. **Selfish Strategies**

Always Defect (ALLD):

- Always plays D

Periodic Defector:

- Cooperates, but defects at fixed intervals

Grim Trigger:

- Cooperates until opponent defects once
- Then defects forever

Tournament Rules:

- Every strategy plays against **every other strategy**, including itself
- Each match consists of **200 rounds** of IPD
- Action noise applies independently at each move

For each match, store:

- Full action history
- Total points after each round
- Average payoff per round
- Final match payoff

Output and Analysis Requirements

(a) Points Table

Strategy	Average score per Match

(b) Heat Map

Define a matrix:

H[i, j] = average payoff per round of strategy i against strategy j

Rows represent the focal strategy; columns represent opponents.

This question is intentionally designed as a *simpler cutout* to the original problem by AxelRod and you can visit to know more about different strategies here: https://axelrod.readthedocs.io/en/dev/reference/strategy_index.html.You are a game theorist attempting to reproduce and extend Axelrod's experiment under noisy conditions.

Your Tasks:

1. Choose numerical values for **T, R, P, S** such that **T > R > P > S** and justify your choice.
2. Implement the IPD tournament with the given strategies and noise.
3. Compare strategy performance using tables and heat maps.
4. Propose **three of your own strategies**, clearly describing:
   i. Their rules
   ii. Which class they belong to (neutral, altruistic, selfish, hybrid)
   iii. Their expected strengths and weaknesses
5. Evaluate whether your strategies behave as intended in a noisy environment.

Short Note (Reflection)

Write a concise discussion addressing:

- Which strategies performed best and why
- How noise affected retaliation, escalation, and forgiveness
- Whether cooperation survived despite errors
- Whether your custom strategies succeeded or failed, and why

The Iterated Prisoner's Dilemma is not just a game.

It is a model of:

- Trust under uncertainty
- Justice versus revenge
- The balance between kindness and caution

In a noisy world, **the most successful strategies are rarely the most aggressive or the most naive - but those that are firm, fair, and forgiving**.
Your task is to discover which behaviors thrive when perfection is impossible. Go ahead and take that in, it is not just about this problem - it is about your life and the social decisions you make.
If you want to know more and learn ahead, we'll recommend you to visit https://www.youtube.com/watch?v=mScpHTIi-kM for a very insightful take on the same topic.

Q2. Big Brother & The Finite Curve - 10 marks

The year is **1984**.
Big Brother does not sleep.
That is the first myth outsiders believe that control comes from omniscience, from watching everything, from screens stacked on screens. In reality, control comes from **structure**. From reducing the world until it fits inside a system small enough to hold, count, and exhaust.
Big Brother learned this the hard way.
The old networks failed because they trusted continuity. Floating numbers. Infinite precision. Signals that blurred and drifted and accumulated error. Too many assumptions layered on top of one another. The system collapsed under its own smoothness.So Big Brother rebuilt the world differently.He chose finiteness.
Lesser mortals believe the world is continuous. They believe numbers stretch forever. They believe division always exists and space can be navigated smoothly.
Big Brother knows better.He knows that every real machine snaps reality to a grid. Every sensor rounds. Every processor truncates. Every decision is discrete, whether the human admits it or not.
If the system is finite, it can be known. If it can be known, it can be controlled.
This one happened on a Tuesday during Chemical Process Control. A gaggle of engineers restless, sleepy, and a little too fond of curves were doodling. They loved loglog plots; the way noise smoothed into lines made their hearts do a weird flip. Someone sketched a cute cubic-looking curve on graph paper and joked about how pretty it would look on a wall.

The students noticed a thing: if you draw a line through two of those points, the line hits the paper again at a third spot that's also on the curve. Flip that spot vertically and, damn, there's another grid-point. Repeatable. Deterministic. No laplace transform needed, just straightedge and patience.
Big Brother loved the idea. Map every server, every relay, every chatterbox in your stack to one of those points. Give each device a coordinate. Make messages flows of point-combinations. Catalogue everything. Precompute the whole damn universe. You can stamp out whispers before they start.

In a regime where every signal is catalogued and every computation logged, even affection must be engineered. Winston Smith and Julia are not just lovers - they are anomalies moving through Big Brother's arithmetic world.

Winston Smith keeps the machines running. Julia manages the comms. They laugh at the feeds and pass each other jokes in the margins of logs. At first it is silly: they stamp "69" into messages as a private wink. But Big Brother is patient. The suits don't read memes - they read arithmetic. They notice that $69 = 3 \times 23$. That tidy arithmetic is a breadcrumb that leads straight into their mapping rules.

So Winston and Julia change tactics. They stop throwing obvious integers into the log. They switch to a different language: coordinates drawn not in continuous ink but mapped onto the tiny constellation of points that Big Brother itself has defined, an elliptic-curve lattice over the prime field $\mathbb{F}_{23}$. They encode presence, whispers, rendezvous plans, all as *points* on that curve. The rule is simple: take your local reading, map it to a valid curve point, and combine your point with your partner's by the curve's addition law. The sum tells you where to meet.
Big Brother smiles. He already precomputed the entire little universe. If he knows the mapping rule, he can follow the breadcrumbs and predict meetups before Winston and Julia arrive.

They work over the finite prime field F_{23} , meaning all arithmetic is done modulo 23.

Using the elliptic curve: **E: $y^2 \equiv (x^3 - 19x + 84) \pmod{23}$**
This curve is well-defined and non-singular.
We will work with a very small grid of numbers:

- Allowed numbers: 0 through 22
- Arithmetic rule: whenever a number goes past 22, wrap it back around (like a clock)

What matters is this:

- You can add numbers
- You can subtract numbers
- You can multiply numbers
- You can divide numbers (except by zero)

And the result is **always** another number between 0 and 22.Nothing escapes.Now place a curve on this grid:

- Pick any x from 0 to 22
- Compute the value: **$y^2 = (x^3 - 19x + 84) \pmod{23}$**
- Keep only the remainder after dividing by 23
- If that value happens to be a perfect square (on this grid), then x has one or two matching y-values

Every matching (x, y) pair is a **valid point**. Collect them all. That entire collection plus one special extra point we'll meet shortly is the whole universe. Finite. Countable. Exhaustible.

The missing piece: how do points "add"?

At first glance, adding two points sounds meaningless. You can't just add coordinates like vectors - that breaks the curve.Instead, addition is defined as a **process**, not a formula. Think like an engineer with graph paper, a straightedge, and patience. You are supposed to use the curve in R^2 (affine/euclidean geometry-as depicted in the image below) while computing the "Addition" operation but remember the points of significance at the end of complete operation will be in the space F_{23} i.e. **$y^2 = (x^3 - 19x + 84) \pmod{23}$** and not in R^2: **$y^2 = x^3 - 19x + 84$**

Step 1: draw a line

Take two points on the curve:

- Winston's point
- Julia's point

Draw the straight line that passes through both of them.
If the two points are the same, draw the tangent line the line that just kisses the curve at that point.

Step 2: find the third intersection

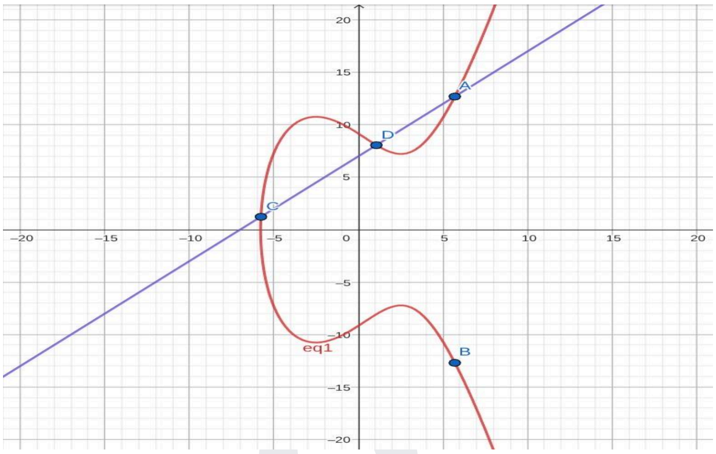
A cubic curve has a stubborn property. Any straight line that hits it twice will hit it a third time. That third intersection point is guaranteed - even on the grid.

Step 3: flip the point

Take that third point and reflect it vertically (flip it across the horizontal axis). The flipped point is defined as the **sum** of the original two points. That's it.

No magic. No mystery. Just geometry turned into a rule.

Below You can see how we "add" C and D to get B.



- There is a **neutral point**: a special "point at infinity" that changes nothing when added. Think of it as zero.
- Every point has an **opposite**: add a point and its mirror image and you get the neutral point.
- Order doesn't matter: adding A to B gives the same result as B to A.
- You never leave the system: adding two valid points always gives another valid point.

In plain language: The system is closed, reversible, and finite.

Big Brother can precompute everything.

Some lines don't come back down.

A vertical line - straight up and down - never intersects the curve a third time on the grid.

Instead of breaking the rules, the system invents a placeholder:

- The **point at infinity**

It lives off the page, but it behaves exactly like zero:

- Add it to any point, nothing changes
- Add a point to its mirror image, you get it

This keeps the world consistent and closed.

Winston & Julia's encoding rule (how numbers become locations)

Winston and Julia don't send raw coordinates. They send ordinary integers hidden in logs. The Oracle sitting in the system computes the Addition and gives them the final location of the next meet. T

The meet rule

Winston's current location becomes a point P_W.

Julia's current location becomes point P_J.

The meeting location is defined as: $P_M = P_W$ "added to" P_J (using the curve's addition process)

No messages about places. No maps. Just arithmetic.

Part 1 Identifying locations

1. Suppose the oracle outputs the point **(16, 9)**.
   What are the individual points they are at?
2. If Winston's observed point is **(3,10)** and Julia's observed point is **(18, 13)**, what if the oracle doesn't work and they have to reveal their location and flee to the computed point immediately can you (BigBrother) find it out?
   Closure! - come here when you are done with the problem; either solving it or giving up on it.

What do you think of this idea of sharing coordinates on a finite complete/closed space, such that any operations done in that space give the points in the space itself - where have you seen this before? Have you?
Did you manage to observe that given the point of meet and let’s say Julia knowing her coordinates, she can figure out Winston’s current location - given the Elliptic curve and prime number base. But an outsider only sees the *combined* point.
Now pause and think.
You have just built a system where:

- information from two people is merged into one object,
- the object lives in a complete, self-contained space,
- every operation stays inside that space,
- and yet separating the object back into its original parts is hard unless you already know one of them.

You’ve seen this idea before.
Not in maps.
Not in geometry class.
Not in continuous space.
You’ve seen it in **locks**, **handshakes**, **shared secrets**, and **agreements made in public that only two parties understand**.
The twist here is that this world is **finite**. There are only so many points. That makes it easy to break when the space is small (like p = 23). Big Brother can list everything. He always wins.
But what if the space isn’t small?
What if the grid has:

- a four-digit prime - F_{abcd}
- a thirty-two-digit prime?
- more points than any machine could list?

Suddenly, the same idea stops being a toy and starts becoming a shield.
You are now standing at the edge of a much deeper idea:

- sharing information by **combining**, not revealing,
- operating in a space where forward moves are easy
- but reversing them without a secret is painfully slow.
  This is the intuition behind **modern secure communication**.
  Not magic. Not secrecy by hiding.
  But secrecy by structure.

Why is it important to have a **Bigggg but finite** world.
You don’t need to know the name yet.
Just remember the feeling:
A closed world, a simple rule, and a problem that becomes impossible only because the space is large.
That’s where Winston and Julia would hide if the grid were big enough.
If this idea stays with you and you want to see how far it goes, a gentle but serious next step is a book by Hoffstein, Pipher, Silverman, 2014.

II : Mazes, Machines, and Moving Anyway

Q3. Navigating the Unknown - 7 marks

Autonomous systems rarely operate with perfect information. In practice, they are deployed into environments that are only partially known, structured by physical constraints rather than ideal geometry, and hostile to brute-force computation.

Consider an **autonomous underwater vehicle (AUV)** deployed inside a submerged cave system during a marine survey mission. Communication with the surface is intermittent, GPS is unavailable, and onboard computation is limited. The vehicle must rely on **local sensing and a general navigation strategy**, not on precomputed global routes.
The cave system has been surveyed and discretized into a grid representation, but the AUV's navigation logic must be **independent of the specific cave layout**.
The cave is represented as a **25 × 25 grid**, where:

- 1 represents **rock wall** (impassable)
- `0` represents **open water** (navigable)

The following structural guarantees hold:

1. The cave is completely enclosed by rock walls along its outer boundary
2. All internal walls are connected to the boundary walls
   (there are no isolated rock "islands" - am I not doing you a favour with this information)
3. The AUV starts at a known **entry point** and must reach a known **exit point**

Task:
Write a program that performs the following:

1. Accepts a cave layout represented as a 25 × 25 matrix of `0`s and `1`s
2. Computes a **valid path** from the given start position to the exit position
3. Uses a **general maze-navigation algorithm**
   - The algorithm must work for *any* cave satisfying the conditions above
   - It must not rely on hardcoded paths or maze-specific tuning
4. Visualizes the result using matplotlib

The visualization must clearly show:

- Rock walls
- The AUV's traversed path
- The start location
- The exit location

In real autonomous systems, success often comes not from clever optimization, but from **choosing the right abstraction of the world**.
For your convenience we have given you the data in digital format in the Gform.

Closure !
The algorithm didn't need intelligence.
It needed faith in constraints.
This is how many real systems work: not by knowing everything, but by exploiting how the world is built. When the environment is shaped carefully, even simple rules can always find a way out.
When the world has guarantees, exploration doesn't need intelligence, only persistence.
The AUV never sees the maze the way you do.
It doesn't know there is an exit.
It doesn't know the walls are connected.
It doesn't know the structure was designed so that persistence is enough.
It moves anyway.
This is closer to how real science works than we like to admit. We do not get the top view. We don't know whether the path we're following leads somewhere or folds back into itself. Most of the time, we don't even know what "somewhere" would look like.
And yet we keep moving, probing, testing, turning, correcting,etc not because we are certain of the outcome, but because stopping guarantees nothing will ever be found.
The maze only looks obvious *after* you've escaped it.

“You can't connect the dots looking forward; you can only connect them looking backwards.” - Steve Jobs.
If you want to see more realistic and general maze solving, watch: https://www.youtube.com/watch?v=ZMQbHMgK2rw

Q4. Elastic Collision at the Bay - 7 marks

A small boat is resting close to a rigid shoreline.
Far offshore, a massive container ship - 1000 times heavier, is pushed toward the shore by a storm. The ship is moving slowly but relentlessly. The boat is initially at rest, floating quietly between the ship and the shore.

The shore behaves like a perfectly rigid wall.
As the ship drifts inward, it collides with the boat. The boat ricochets off the shore, collides again with the ship, then the shore again, a rapid back-and-forth exchange of momentum. This cycle repeats many times.
Eventually, something counterintuitive happens:

- the massive ship reverses direction,
- and the ship’s speed exceeds that of the small boat.

Nothing exploded. No external force intervened. All that happened was **a sequence of elastic collisions**.
Hint: Assume that both the objects are perfectly cubical and the motion is captured such that it occurs in 1 dimension only.
Am I not doing you a favour?Questions:
Record:

- the total number of collisions
- the final velocities of both blocks

Plot velocities of both blocks as a function of collision count.

Bonus:

Generalize your code to N ships and the wall, with decreasing masses. N = 3
Progress is often invisible until the moment it suddenly isn’t !

For a deeper intuition behind repeated elastic collisions and why unexpected outcomes appear, see: https://www.youtube.com/watch?v=6dTyOI1fmDo&vl=en

III : Processes, Power, and Pretending

Q.5 If it were obvious, it wouldn’t be engineering - 10 marks

Ammonia oxidation lies at the heart of one of the most economically important chemical processes ever developed, the Ostwald process, used worldwide for nitric acid production. At industrial scale, this reaction network moves millions of tonnes of material every year, releases enormous heat, destroys expensive catalysts, and quietly prints money.

Your commerce/MBA friend looks at the overall reaction, sees ammonia go in and nitric acid come out, and smiles.
“This is trivial,” they say.
“Raw material in, product out, margins look great. Scale it. IPO it.”
You, unfortunately, know better.

Because before a single reactor is sized, before heat removal is designed, before safety margins are set you don't even know what's happening inside the box.And yet, the plant must be built.

Ammonia is converted to nitric acid in two major stages. The first stage involves high-temperature catalytic oxidation, where several side reactions compete with the desired pathway. The catalyst degrades. The gas phase is violent. Residence times are short. Many intermediates are unobservable. Assume the reactions to follow rate laws and might be series + parallel gas phase reactions only.

You are not asked to model the full Ostwald process.

Instead, you are given a reduced kinetic model, inferred from experimental data, which captures only selected nitrogen-containing species.
Data is mentioned in the Gform for your convenience.

Using the provided dataset, the unknown kinetic parameters are to be estimated numerically. There is no requirement that the fitted parameters correspond to physically measurable quantities, only that the resulting model reproduces the observed behaviour in a stable and consistent manner. Since the data are noisy and partially aggregated, exact agreement is neither expected nor required.

There is no unique correct solution. The model will be evaluated solely on its ability to explain the data, as quantified through goodness-of-fit metrics such as the coefficient of determination and mean squared error, together with clear concentration–time plots comparing model predictions against experimental measurements. Marks will be awarded even if the proposed mechanism and rate laws differ from our solution, provided the modelling logic is sound and the fit quality is satisfactory.

Q6- Reaction Equilibrium for Multiple Gas-Phase Reactions - 5 marks

The following reactions are taking place in a constant-volume, gas-phase batch reactor.

$$A + B \leftrightarrow C + D$$
$$B + C \leftrightarrow X + Y$$
$$A + X \leftrightarrow Z$$

A system of algebraic equations describes the equilibrium of the above reactions. The nonlinear equilibrium relationships used the thermodynamic equilibrium expressions, and the linear relationships have been obtained from the stoichiometry of the reactions.

$$K_{C1} = \frac{C_C C_D}{C_A C_B} \qquad K_{C2} = \frac{C_X C_Y}{C_C C_B} \qquad K_{C3} = \frac{C_Z}{C_A C_X}$$

$$C_A = C_{A0} - C_D - C_Z \qquad C_B = C_{B0} - C_D - C_Y$$
$$C_C = C_D - C_Y \qquad C_Y = C_X + C_Z$$

In this equation set, C A , CB , CC , CD , CX , CY , and C Z are concentrations of the various species at equilibrium resulting from initial concentrations of only C A 0 and C B 0 . The equilibrium constants KC 1 , KC 2, and KC 3 have known values.
Solve this system of equations when $C_{A0} = C_{B0} = 1.5,\ K_{C1} = 1.06,\ K_{C2} = 2.63,\ and\ K_{C3} = 5,$ starting from three sets of initial estimates.

$$1.\ C_D = C_X = C_Z = 0$$

2. $C_D = C_X = C_Z = 1$

3. $C_D = C_X = C_Z = 10$

Question 7- Dynamics of a Heated Tank with Proportional/Integral (PI) Temperature Control - 8 marks

A continuous process system consisting of a well-stirred tank, heater and PI temperature controller is depicted in Figure (A.4). The feed stream of liquid with density of ϱ (Kg/m^3) and heat capacity of C $(KJ\ Kg^{-1} \circ C^{-1})$ flows into the heated tank at a constant rate of W (Kg/min) and temperature T_i $(\circ C)$. The volume of the tank is V (m^3). It is desired to heat this stream to a higher set point temperature T_r (∘C). The outlet temperature is measured by a thermocouple as T_m (∘C), and the required heater input q (kJ/min) is adjusted by a PI temperature controller. The control objective is to maintain $T_0 = T_r$ in the presence of a change in inlet temperature T_i, which differs from the steady state design temperature of T_{is}. An energy balance on the stirred tank yields with initial condition $T = T_r$ at t=0, which corresponds to steady-state operation at the set point temperature T_r.

$$\frac{dT}{dt} = \frac{WC_p(T_i - T) + q}{\rho V C_p}$$

The thermocouple for temperature sensing in the outlet stream is described by a first-order system plus the dead time td , which is the time for the output flow to reach the measurement point. The dead time expression is given by

$T_0(t) = T(t - \tau_d)$

The effect of dead time may be calculated for this situation by the Pade ' approximation, which is a first-order differential equation for the measured temperature.

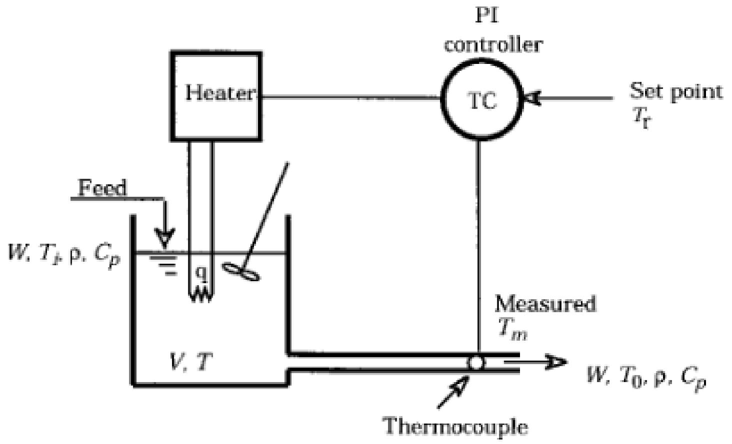


Figure A.4 Well-mixed tank with heater and temperature controller.

$$\frac{dT}{dt} = \left[T - T_0 - \left(\frac{\tau_d}{2}\right)\left(\frac{dT}{dt}\right)\right]\frac{2}{\tau_d} \quad \text{I.C}$$

$T_0 = T_r$ at t=0 (steady state)

The above equation is used to generate the temperature input to the thermocouple, T_0.

The thermocouple shielding and electronics are modeled by a first-order system for the input temperature T_0 given by

$$\frac{dT}{dt} = \left[\frac{T_0 - T_m}{T_m}\right]$$

I.C. $T_m = T_r$ at t=0 (steady state)

where the thermocouple time constant tm is known. The energy input to the tank, q, as manipulated by the PI controller can be described by

$$q = q_s + K_c(T_r - T_m) + \frac{K_c}{\tau_d \int_0^t (T_r - T_m)dt} \qquad \text{(A.44)}$$

where K_c is the proportional gain of the controller, and τ_d is the integral time constant or reset time. The q_s in the above equation is the energy input required at steady state for the design conditions as calculated by

$$q_s = WC_p(T_r - T_{is}) \qquad \text{(A.45)}$$

The integral in Equation (A.44) can be conveniently calculated by defining a new variable as

$$\frac{d}{dt}(errsum) = T_r - T_m$$

I.C. erratum = 0 at t=0 (steady state) (A.46)

Thus, Equation (A.44) becomes

$$q = q_s + K_c(T_r - T_m) + K_c/\tau_I \text{ (errsum)} \qquad \text{(A.47)}$$

Let us consider some of the interesting aspects of this system as it responds to a variety of parameter and operational changes. The numerical values of the system and control parameters in Table A.3 will be considered as leading to baseline steady-state operation.

$\rho VC_p = 4000$ kJ/°C	$WC_p = 500$ kJ min^{-1} °C^{-1}
$T_{is} = 60$°C	$T_r = 80$°C
$\tau_d = 1$ min	$\tau_m = 5$ min
$K_c = 50$ kJ min^{-1} °C^{-1}	$\tau_I = 2$ min

1. Demonstrate the open-loop performance (set $K_c = 0$) of this system when the system is
initially operating at design steady state at a temperature of 80∘C, and inlet temperature T_i is suddenly changed to 40∘C at time t Å 10 min. Plot the temperatures T,T_0, and T_m to steady state, and verify that Pade^ approximation for 1 min of dead time given in Equation (A.42) is working properly.
2. Demonstrate the closed-loop performance of the system for the conditions of part 1 and the baseline parameters from Table A.3. Plot Temperatures T, T_0 , and T_m to a steady state.
3. Repeat part 2 with K_c = 500 kJ min^{-1} $\circ C^{-1}$.
4. Repeat part 3 for proportional-only control action by setting the term K_c/τ_I= 0.
5. Implement limits on q [as per Eq. (A.47)] so that the maximum is 2.6 times the baseline steady-state value and the minimum is zero. Demonstrate the system response from baseline steady state for a proportional only controller when the set point is changed from 80∘C to 90∘C at t=10 min. K_c = 5000 kJ min^{-1} $\circ C^{-1}$. Plot q and q_{lim} versus time to steady state to demonstrate the limits. Also, plot the temperatures T, T_0 , and T_m to steady state to indicate controller performance.

Q8 Hmm… is this operations research? Hmmmmmmm - 8 marks

Imagine a toddler who loves crushing insects and doing other unusual activities(not necessarily bad...). This toddler has in mind 5 different creations to complete. He has in possession 1 pressure cooker, 2 solar dryers, 1 blender, 1 batch distillation column.

- “Cockroach pellets” need a pressure cooker, a blender and a dryer. It takes 1 hour in the pressure cooker, 0.2 hours in the blender and 1 hour in the dryer
- “Fish crystals” need a blender, a batch distillation column and a dryer. It takes 2 hours in each
- “Intercontinental HVDC Underwater Cable crystals” needs a blender and a dryer. It takes 1.5 hours in the blender and 1 hour in the dryer
- “Dell Laptop Liquid” needs a pressure cooker and a batch distillation column. It takes 0.4 hours in the pressure cooker and 1 hour in the batch distillation column
- “Egg Yolk pellets” need a blender and a dryer. It takes 0.5 hours in the blender and 1 hour in the dryer

Find an optimal schedule to produce all 5 of his creations. Assume the pressure cooker takes 0.5 hours to be cleaned if two separate products are being manufactured in it one after the other. Assume storage between tasks is not an issue.
Once you’re done with the above task - what would have happened if I increased the number of products being manufactured in this disturbed child’s room 4 fold? If he had 20 products and the production schedule had to be decided. How would you find the optimal schedule? Would you face any issues? (no need to write the code for this)

Q8 **Beautiful particles! So pretty to look at…** - 10 marks

`Cute cat staring at you as you make this simulation`

Make yourself a 2D particle simulation. The particles must whoosh about and kick each other around, occasionally sit by each other for a chat, a chai. So, gang, there are N particles in the mix. They each have a position (x,y) and a velocity (vx,vy). Acceleration of the particles depends on the sum of pairwise interaction forces between the particles and others near it, friction (to simulate a medium, you'll see why later). The interaction forces will have their attraction and repulsion components. The equations are given below. You are to build a set of differential equations or invent a new mathematical technique(joke) to see how the particles in this system behave over time.

$$\frac{dr_i}{dt} = v_i \quad m\frac{dv_i}{dt} = \Sigma_{j\neq i}F_{ij} \quad \gamma v_i$$

F_{ij} is a bit of an emo function. So we must define 2 radii for particles: $r_{0,}$ r_1, where the firat is the radius below whic repulsion is the contrioling force and r1 is the radius after which no force is applicable between that particle and the rest.

- When the distance between two particles, rij<r0.

- R1
- If its more than r1, you can assume the force is 0.

Very importantly, pay close attention to what Gkl is. Its an interaction parameter betwen classes of particles. Yiu can have as many classes of particles in your mixtures (keep more than two to see exciting stuff). K and L were just filter letters for this.
Keep this in mind that forces must be in the vector direction of rij.

$$\frac{dr_i}{dt} = v_i$$

$$m\frac{dv_i}{dt} = \Sigma_{j\neq i}F_{ij} - \gamma v_i$$

F_{ij} is a bit of an emo function. So we must define 2 radii for particles: r_0, r_1, where the first is the radius below which repulsion is the controlling force and r_1 is the radius after which no force is applicable between that particle and the rest

- When the distance between two particles, $r_{ij} < r_0$:

$$F_{AB} = G_{KL}A(1 - \frac{r}{r_0})$$

- $r_1 > r_{ij} > r_0$:

$$F_{ij} = -G_{KL}B(1 - \frac{r - r_0}{r_1 - r_0})$$

- if its more than r1, you can assume the force is 0

Very importantly, pay close attention to what G_{KL} is. It's an interaction parameter between classes of particles. You can have as many classes of particles in your mixtures (keep more than two to see exciting stuff). K and L were just filler letters for this.

Keep in mind that the forces must be in the vector direction of r_{ij}.

To talk a little more on the attraction para meter. They show how much different classes o f particles attractepel each other - basically how much they interact with each other. It's easy to visualise this as a matrix of interaction parameters

It must, rhen be symmetric matrix since the pairwise interactions between any two particles must be the same (otherwise you break an array of the physics laws).

Your task is to make this simulation of the particles, and make a video or GIF animation of it using python. If I were you, and I wasn't using something like P5JS and didn't know python drawing libraries like the back of my hand, I had just plot it and animate the plot.

Enjoy.

ALSO MAKE WALLS!!!

	H2O	Nitrogen	CO2	H2S	Methane	Ethane
H2O	---	-0.48000	0.10000	0.03000	0.45000	0.45000
Nitrogen	-0.48000	---	-0.03150	0.16960	0.02780	0.04070
CO2	0.10000	-0.03150	---	0.09890	0.12000	0.12000
H2S	0.03000	0.16960	0.09890	---	0.08000	0.08520
Methane	0.45000	0.02780	0.12000	0.08000	---	0.00000
Ethane	0.45000	0.04070	0.12000	0.08520	0.00000	---

It must, then be a symmetric matrix since the pairwise interactions between any two particles must be the same (otherwise you break an array of physics laws)

Your task is to make this simulation of particles, and make a video or GIF animation of it using python. If I were you, and I wasn't using something like P5JS and didnt know python drawing libraries like the back of my hand, I'd just plot it and animate the plot.

Enjoy.

ALSO MAKE WALLS!!!

Now interpret creation and how it relates to real life:

Once you actually run this simulation, it's beautiful. Simply breathtaking. You'll notice that sometimes the particles form complex looking clumps and move about in very pretty, very ordered manners. If this is a simulation of particles that seemingly follows the laws of physics, then WHY? WHY isn't it absolutely random, completely chaotic? We all know that the fundamental law of the universe is that entropy increases. Why doesn't it become more chaotic? (no need to answer this (you can if you want though...), just ponder upon it and maybe do a google search after the contest)

From a thermodynamic point of view, the chemical potential of a component measures the free energy cost of removing one particle from its current environment and placing it elsewhere, such as into a vapour phase. In the simulation, this is reflected by how deep the effective potential well of a particle is due to its neighbours and how constrained it is by packing. Strong cross interactions that stabilise a particle lower its chemical potential relative to an ideal mixture, while weaker cross interactions or poor packing raise it. Even though all particles may attract each other, what matters is whether the mixed environment stabilises a particle more or less than expected from ideal mixing.

With this in mind, explore how changing the off diagonal elements of the interaction matrix alters the structures you see and the ease with which particles escape from dense regions into low density regions. When cross interactions are strong, particles become more tightly bound within mixed clusters and escape events are rare. When cross interactions are weak, particles feel less stabilised by unlike neighbours and are more likely to leave the liquid like region. Based on what you observe, ask yourself: how would the tendency of each component to vaporize change as the cross interaction strength is varied, and how does this connect to the idea of activity coefficients being greater than or less than one in real liquid mixtures?

Notes:
While I really want to include funny and cool images in this question, I won't. The video you'll generate yourself is more than enough stimulation
Why didn’t I use proper functions for the forces like that gravity one? Infinity

If you reached this point looking for a moral, there isn’t one. These problems do not reward virtue, effort, or even correctness, only structure, patience, and your willingness to operate inside rules you did not choose. Cooperation fails, control wins, exploration survives without understanding, and meaning appears only after the computation is over. If that makes you uncomfortable, good you’ve understood the paper. Given the same constraints, the same behaviors would resurface indefinitely, indifferent to interpretation. What feels like an ending is only where observation stops. The system would prefer you not mistake that for closure.