Mathematical Modelling of a Friendship Using a Third Dummy Friend

Let people be modeled as hypothetical springs with n-dimensional stiffness matrices. Such a model should be feasible because all physical dynamical systems can be mathematically formulated as a spring-mass-damper configuration. The goal is to figure out whether 2 people can be friends.

The algorithm introduces a user-chosen questionnaire comprising of n questions. This questionnaire should be answered by both participants in the form of numbers. These numbers would be the diagonals of the corresponding stiffness matrices of each participant, thus the numbers should be chosen such that they realistically model the person. For instance, a high value can represent a relatively "stubborn" response, which means that the corresponding element of the stiffness matrix tries to make the spring more rigid. A low number can represent a more "accepting" or "flexible" response to the question, thus the corresponding matrix element contributes in making the spring more flexible. Intuitively, this should describe parts of the brain being taut or relaxed, which determine stress levels[1].

The third person is a dummy model, which is used as an analogy to what situation the two participants are in. The stiffness parameters for this dummy needs to be chosen carefully using trends. The dummy acts as an analogous "person" that mathematically replaces "situations and events", or anything else that might affect the friendship. The selection of the dummy elements are quite possibly the hardest and most tedious part of this problem, and requires careful study of psychology; each situation or event would require to be uniquely modeled into the dummy.

The solver analyzes an elastic 3-body problem[2], where ideally 3 bodies are connected via 3 springs in a star configuration.
The current solver solves:

$$\mathbf{M}\ddot{\mathbf{x}} + \mathbf{C}\dot{\mathbf{x}} + (\mathbf{K_1} + \mathbf{K_2} + \mathbf{K_3})\mathbf{x} = 0$$

The symbols have their usual meanings. It is to be noted that the matrices have been compressed using block-reduction[3] such that the state vector is 3-D.

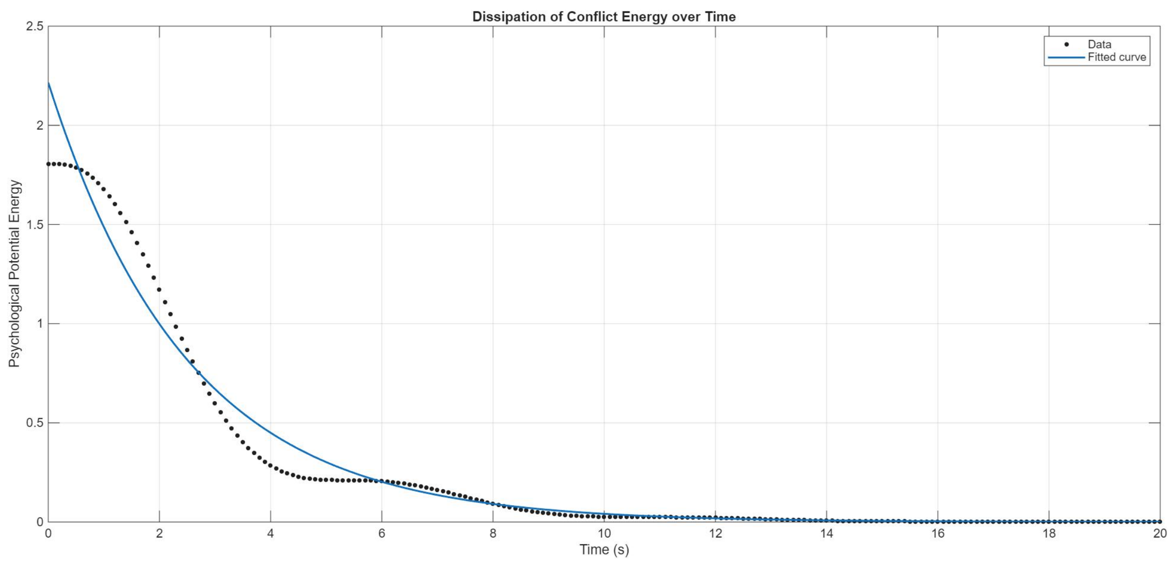
For testing purposes, the inputs have been automated and randomized. The following shows the MATLAB code, and the output so far:

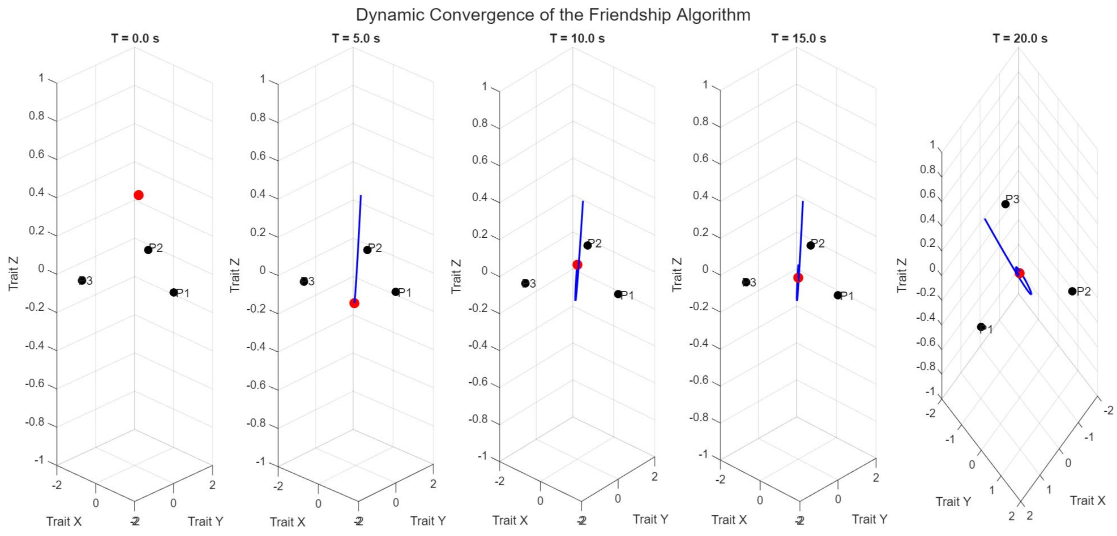
```
function friendship_dynamics()
clc; clear; close all;

%% 1. The Questionnaire & Stiffness Generation
% We simulate inputs for 3 people (P1, P2, P3).
% In a real scenario, you would use 'input()' to get these values.
% Scale: [-1, 1]. Magnitude = Stiffness (Stubbornness).
disp('Initializing Psychological Questionnaire...');
num_questions = 20;
% Generating synthetic responses for demonstration (Simulating User Input)
% A value of 0.9 means "Very Stubborn/Stiff" on that topic.
% A value of 0.1 means "Very Flexible".
responses_P1 = 2 * rand(num_questions, 1) - 1;
responses_P2 = 2 * rand(num_questions, 1) - 1;
responses_P3 = 2 * rand(num_questions, 1) - 1;

% Create High-Dimensional Stiffness Matrices (20x20)
% K is diagonal because we assume topics are initially independent
% We take absolute value because Stiffness must be positive.
```

```
% We add 0.1 so no spring is perfectly broken (0 stiffness).
K1_full = diag(abs(responses_P1) + 0.1);
K2_full = diag(abs(responses_P2) + 0.1);
K3_full = diag(abs(responses_P3) + 0.1);

fprintf('Stiffness Matrices created for 3 subjects based on %d questions.\n', num_questions);

%% 2. Dimensionality Reduction (Compression to 3D)
% We compress the 20 dimensions into 3 fundamental dimensions (X, Y, Z)
% to visualize the "Star Configuration" in 3D space.
% Mapping: Q1-Q7 -> X-axis, Q8-Q14 -> Y-axis, Q15-Q20 -> Z-axis
function K_3D = compress_stiffness(K_full)
diag_vals = diag(K_full);
k_x = mean(diag_vals(1:7)); % Average stiffness in "Trait X"
k_y = mean(diag_vals(8:14)); % Average stiffness in "Trait Y"
k_z = mean(diag_vals(15:20)); % Average stiffness in "Trait Z"
K_3D = diag([k_x, k_y, k_z]);
end

K1 = compress_stiffness(K1_full);
K2 = compress_stiffness(K2_full);
K3 = compress_stiffness(K3_full);

%% 3. Defining the Dynamic System (Star Configuration)
% Mass M is connected to 3 walls by springs K1, K2, K3.
% Equation of Motion: M*x'' + C*x' + K_eq*x = 0
% where K_eq = K1 + K2 + K3 (Parallel springs holding the center)
M = 5 * eye(3); % Mass matrix (Inertia of the group)
C = 2 * eye(3); % Damping matrix (Emotional regulation/friction)
K_eq = K1 + K2 + K3; % Equivalent Stiffness of the star center
% Initial Condition: A "Disagreement" pushes the group away from center
x0 = [1.0; -0.8; 0.5]; % Initial displacement
v0 = [0; 0; 0]; % Initial velocity

% Solve ODE using State Space representation
% State vector z = [position; velocity]
tspan = 0:0.1:20;
[t, z] = ode45(@(t,z) system_dynamics(t, z, M, C, K_eq), tspan, [x0; v0]);
positions = z(:, 1:3); % Extract X, Y, Z coordinates over time

%% 4. Visualization: 5 Time-Step Snapshots
figure('Name', 'Friendship Dynamics: The Search for Equilibrium', 'Color', 'w');
indices = floor(linspace(1, length(t), 5)); % Select 5 evenly spaced time steps
for i = 1:5
idx = indices(i);
subplot(1, 5, i);
% Plot the Center (Friendship State)
plot3(positions(1:idx,1), positions(1:idx,2), positions(1:idx,3), 'b-', 'LineWidth', 1.5);
hold on;
plot3(positions(idx,1), positions(idx,2), positions(idx,3), 'ro', 'MarkerFaceColor', 'r',
'MarkerSize', 8);
% Plot "Walls" (The 3 Individuals anchoring the friendship)
% We arbitrarily place them in 3D space to visualize the pull
scatter3(2, 0, 0, 50, 'k', 'filled'); text(2.1,0,0, 'P1');
scatter3(-1, 1.7, 0, 50, 'k', 'filled'); text(-1.1,1.8,0, 'P2');
scatter3(-1, -1.7, 0, 50, 'k', 'filled'); text(-1.1,-1.8,0, 'P3');
% Formatting
grid on; axis([-2 2 -2 2 -1 1]);
title(sprintf('T = %.1f s', t(idx)));
xlabel('Trait X'); ylabel('Trait Y'); zlabel('Trait Z');
view(45, 30);
end
sgtitle('Dynamic Convergence of the Friendship Algorithm');
```

```
%% 5. Trend Analysis (Mathematical Formulation)
% Calculate System Energy (Lyapunov Function) over time
% V(t) = 0.5 * x' * K_eq * x + 0.5 * v' * M * v
energy = zeros(length(t), 1);
for k = 1:length(t)
pos = z(k, 1:3)';
vel = z(k, 4:6)';
energy(k) = 0.5 * pos' * K_eq * pos + 0.5 * vel' * M * vel;
end
% Fit an exponential decay curve: E(t) = E0 * exp(-lambda * t)
f = fit(t, energy, 'exp1');
% Display Formulation
fprintf('\n--------------------------------------------------\n');
fprintf('TREND ANALYSIS:\n');
fprintf('The stability of this friendship follows a Lyapunov Decay.\n');
fprintf('Mathematical Trend: E(t) = %.2f * e^(%.2f * t)\n', f.a, f.b);
if f.b < -0.5
fprintf('Conclusion: Highly Stable. Conflicts resolve quickly.\n');
elseif f.b < 0
fprintf('Conclusion: Stable, but grudges linger (slow decay).\n');
else
fprintf('Conclusion: Unstable. The group cannot reach consensus.\n');
end
fprintf('------------------------------------------------\n');
% Plot Energy Trend
figure('Name', 'Energy Trend', 'Color', 'w');
plot(f, t, energy);
title('Dissipation of Conflict Energy over Time');
xlabel('Time (s)'); ylabel('Psychological Potential Energy');
grid on;

end

%% Auxiliary Function: System Dynamics ODE
function dzdt = system_dynamics(~, z, M, C, K)
% State vector z contains [x; y; z; vx; vy; vz]
pos = z(1:3);
vel = z(4:6);
% Acceleration = M^-1 * (-C*vel - K*pos)
acc = M \ (-C * vel - K * pos);
dzdt = [vel; acc];
end
```

```
Initializing Psychological Questionnaire...
Stiffness Matrices created for 3 subjects based on 20 questions.

------------------------------------------------
TREND ANALYSIS:
The stability of this friendship follows a Lyapunov Decay.
Mathematical Trend: E(t) = 2.22 * e^(-0.40 * t)
Conclusion: Stable, but grudges linger (slow decay).
------------------------------------------------
```





This particular group of people show a slow decay.

In reality, however, relationships are not predictable[4], they are almost always extremely chaotic systems. It thus makes sense to convert the elastic 3 body problem into a modified version of the classic gravitational 3 body problem where non-linearity is introduced[2]. This transforms the algorithm into a much more realistic model which tracks the complexity and non-predictability[5] of human interactions. This seems feasible because the human brain is usually modeled as a non-linear system[6]. The solver now solves the following equation:

$$m\frac{d^2\mathbf{r}_i}{dt^2} = \underbrace{-c\frac{d\mathbf{r}_i}{dt}}_{\text{Damping}} + \sum_{\substack{j=1 \\ j\neq i}}^{3} \left[\underbrace{\frac{Gm^2}{\|\mathbf{r}_{ij}\|^2 + \delta}}_{\text{Gravitational}} + \underbrace{k_{ij}\left(\|\mathbf{r}_{ij}\| - L_0\right)}_{\text{Elastic}} \right] \frac{\mathbf{r}_j - \mathbf{r}_i}{\|\mathbf{r}_{ij}\|}$$

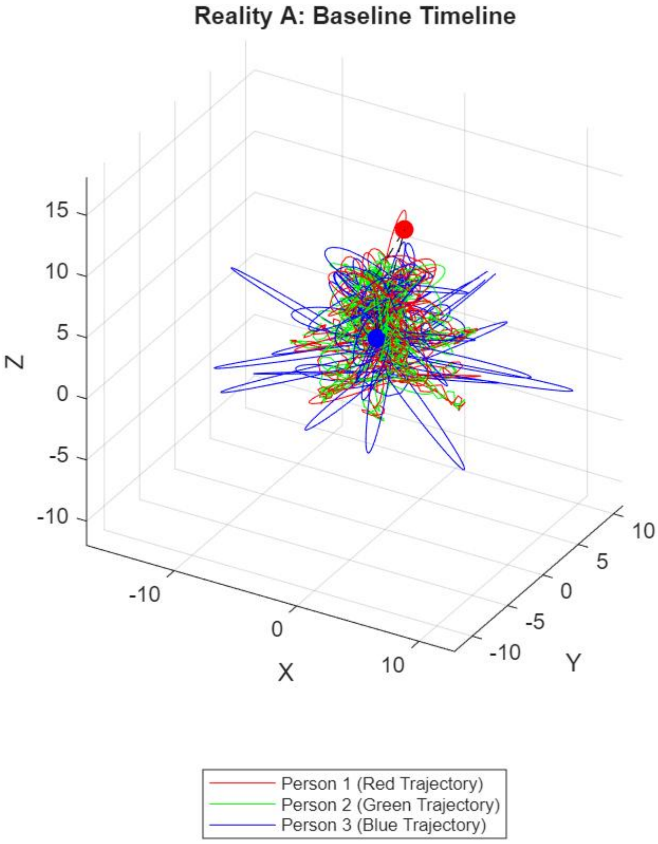
Each symbol has their usual meaning. The padding has been added to the gravitational term to eliminate singularities. The new algorithm uses 2 different initial conditions, and plots the trajectories of the masses, along with the state difference with respect to time on a logarithmic scale. The code and the output are as follows:

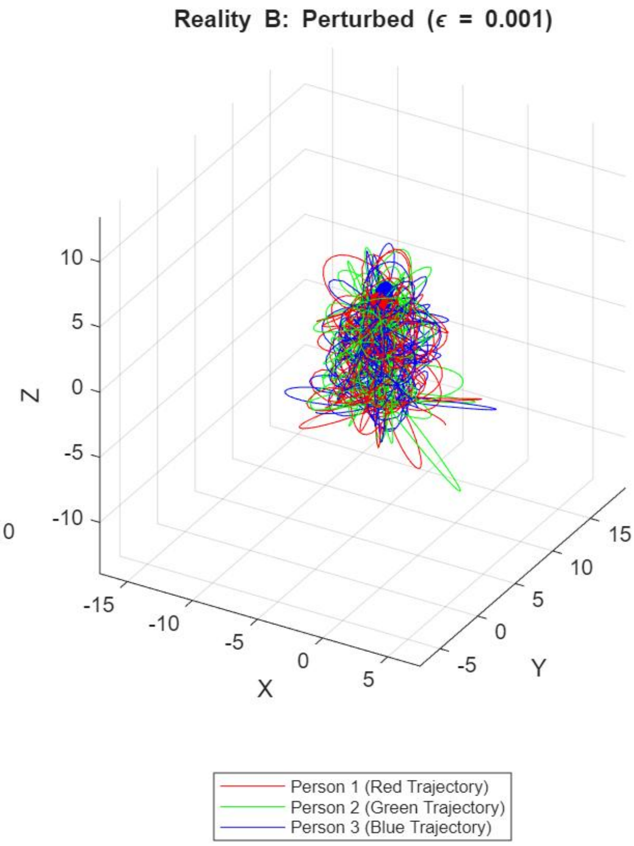
```
function chaotic_friendship_detailed_viz()
clc; clear; close all;

%% 1. Setup: Randomized Psychological Stiffness
disp('Initializing Dual-Reality Simulation with Detailed Legends...');
% --- Step 1: Generate Personalities ---
num_questions = 20;
R1 = 2 * rand(num_questions, 1) - 1;
R2 = 2 * rand(num_questions, 1) - 1;
R3 = 2 * rand(num_questions, 1) - 1;

% --- Step 2: Calculate Elastic "Bond Strength" ---
% Multiplier 35 ensures springs compete with the strong gravity
K12 = mean(abs(R1 - R2)) * 35;
K23 = mean(abs(R2 - R3)) * 35;
K31 = mean(abs(R3 - R1)) * 35;
K_params = [K12, K23, K31];
fprintf('Bond Strengths: K12=%.1f, K23=%.1f, K31=%.1f\n', K12, K23, K31);
```

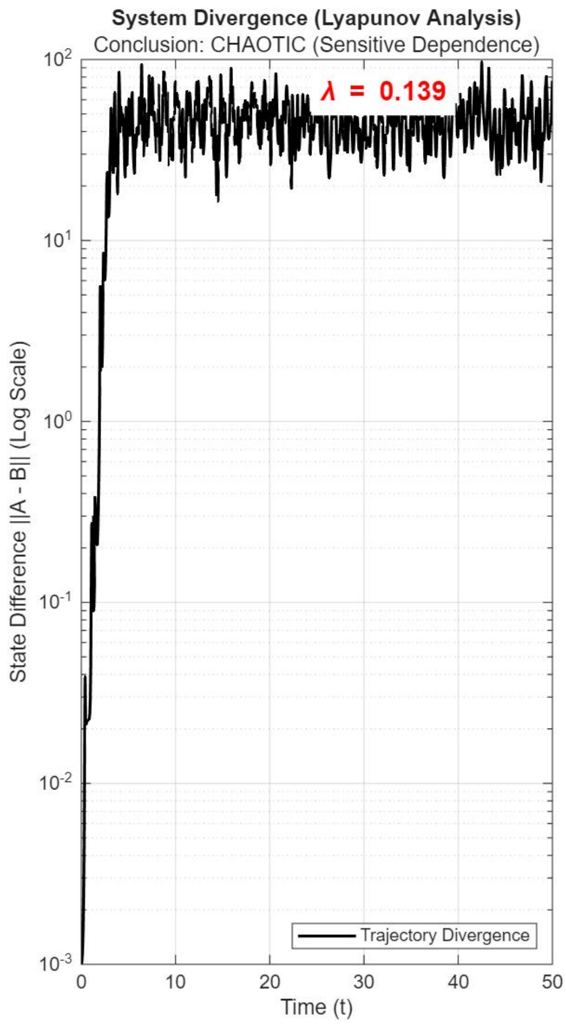
```matlab
%% 2. System Physics
G = 150; % Social Gravity (Chaotic driver)
m = 10; % Mass (Inertia)
c = 0.02; % Damping (Low friction)
L0 = 4; % Ideal distance

%% 3. The Butterfly Effect Setup (Epsilon)
pos1 = [5, 0, 0];
pos2 = [-2.5, 4.3, 0];
pos3 = [-2.5, -4.3, 1];
vel1 = [0, 2, 1];
vel2 = [-1.7, -1, 0];
vel3 = [1.7, -1, -0.5];

% Reality A
y0_A = [pos1, pos2, pos3, vel1, vel2, vel3]';
% Reality B (The Perturbed Timeline)
epsilon = 1e-3;
y0_B = y0_A;
y0_B(1) = y0_B(1) + epsilon; % Tiny shift in Person 1's X position

%% 4. Run Simulations
tspan = 0:0.05:50;
options = odeset('RelTol', 1e-6, 'AbsTol', 1e-6);
disp('Simulating Reality A...');
[t, yA] = ode45(@(t,y) dynamics(t, y, m, G, c, K_params, L0), tspan, y0_A, options);
disp('Simulating Reality B...');
[~, yB] = ode45(@(t,y) dynamics(t, y, m, G, c, K_params, L0), tspan, y0_B, options);

%% 5. Visualization
figure('Name', 'Chaotic Friendship: Butterfly Effect Analysis', 'Color', 'w', 'Position',
[50, 50, 1600, 600]);
% --- SUBPLOT 1: REALITY A ---
subplot(1, 3, 1);
plot_trajectory(yA, 'Reality A: Baseline Timeline');
% --- SUBPLOT 2: REALITY B ---
subplot(1, 3, 2);
plot_trajectory(yB, sprintf('Reality B: Perturbed (\\epsilon = %.3f)', epsilon));
% --- SUBPLOT 3: THE DIVERGENCE ---
subplot(1, 3, 3);
% Calculate Euclidean Distance
delta = sqrt(sum((yA - yB).^2, 2));
semilogy(t, delta, 'k-', 'LineWidth', 1.5);
title('System Divergence (Lyapunov Analysis)');
xlabel('Time (t)');
ylabel('State Difference ||A - B|| (Log Scale)');
grid on;
% Add Legend for Divergence
legend({'Trajectory Divergence'}, 'Location', 'southeast');
% Fit Lyapunov Trend
half_idx = round(length(t)/2);
p = polyfit(t(1:half_idx), log(delta(1:half_idx)), 1);
lambda = p(1);
% Annotate Graph
text(t(half_idx), delta(half_idx), sprintf('\\lambda = %.3f', lambda), ...
'FontSize', 14, 'FontWeight', 'bold', 'Color', 'r', 'BackgroundColor', 'w');
if lambda > 0
subtitle('Conclusion: CHAOTIC (Sensitive Dependence)');
else
subtitle('Conclusion: STABLE (Predictable)');
end

end
```

```matlab
%% Helper: Trajectory Plotter with Detailed Legends
function plot_trajectory(y, plot_title)
% Extract Coordinates
P1 = y(:, 1:3);
P2 = y(:, 4:6);
P3 = y(:, 7:9);
hold on;
% Plot Paths (Capture handles for legend)
h1 = plot3(P1(:,1), P1(:,2), P1(:,3), 'r-', 'LineWidth', 0.5);
h2 = plot3(P2(:,1), P2(:,2), P2(:,3), 'g-', 'LineWidth', 0.5);
h3 = plot3(P3(:,1), P3(:,2), P3(:,3), 'b-', 'LineWidth', 0.5);
% Plot End Positions (Markers)
plot3(P1(end,1), P1(end,2), P1(end,3), 'ro', 'MarkerFaceColor', 'r', 'MarkerSize', 8);
plot3(P2(end,1), P2(end,2), P2(end,3), 'go', 'MarkerFaceColor', 'g', 'MarkerSize', 8);
plot3(P3(end,1), P3(end,2), P3(end,3), 'bo', 'MarkerFaceColor', 'b', 'MarkerSize', 8);
% Connect the group at the final moment (The triangle)
line([P1(end,1) P2(end,1)], [P1(end,2) P2(end,2)], [P1(end,3) P2(end,3)], 'Color', 'k',
'LineStyle', '--');
line([P2(end,1) P3(end,1)], [P2(end,2) P3(end,2)], [P2(end,3) P3(end,3)], 'Color', 'k',
'LineStyle', '--');
line([P3(end,1) P1(end,1)], [P3(end,2) P1(end,2)], [P3(end,3) P1(end,3)], 'Color', 'k',
'LineStyle', '--');

grid on; axis equal;
% Detailed Legend
legend([h1, h2, h3], ...
{'Person 1 (Red Trajectory)', ...
'Person 2 (Green Trajectory)', ...
'Person 3 (Blue Trajectory)'}, ...
'Location', 'best', 'FontSize', 8);
% Simplified Axes Labels
xlabel('X'); ylabel('Y'); zlabel('Z');
title(plot_title);
view(30, 30);
% Set fixed limits to ensure easy visual comparison between Plot A and B
xlim([-20 20]); ylim([-20 20]); zlim([-20 20]);
end

%% Physics Engine
function dydt = dynamics(~, y, m, G, c, K, L0)
% Unpack State
r1 = y(1:3); r2 = y(4:6); r3 = y(7:9);
v1 = y(10:12); v2 = y(13:15); v3 = y(16:18);
k12 = K(1); k23 = K(2); k31 = K(3);
% Force Calculation
% Pair 1-2
d12 = r2 - r1; dist12 = norm(d12); dir12 = d12 / dist12;
F_grav_12 = (G * m^2 / (dist12^2 + 2)) * dir12;
F_spr_12 = k12 * (dist12 - L0) * dir12;
% Pair 2-3
d23 = r3 - r2; dist23 = norm(d23); dir23 = d23 / dist23;
F_grav_23 = (G * m^2 / (dist23^2 + 2)) * dir23;
F_spr_23 = k23 * (dist23 - L0) * dir23;
% Pair 3-1
d31 = r1 - r3; dist31 = norm(d31); dir31 = d31 / dist31;
F_grav_31 = (G * m^2 / (dist31^2 + 2)) * dir31;
F_spr_31 = k31 * (dist31 - L0) * dir31;
% Net Forces
F1 = F_grav_12 - F_grav_31 + F_spr_12 - F_spr_31 - c*v1;
F2 = -F_grav_12 + F_grav_23 - F_spr_12 + F_spr_23 - c*v2;
F3 = -F_grav_23 + F_grav_31 - F_spr_23 + F_spr_31 - c*v3;
dydt = [v1; v2; v3; F1/m; F2/m; F3/m];
```

```
end
```







It can be observed that with a tiny difference in initial conditions, the outcomes are largely different, and the state difference between A and B shoots up. This is a demonstration of the Butterfly Effect[7], which is often used in psychology to explain patterns[8]. This mathematical model can be used as an approximate predictor of friendship patterns. Further analysis would include modelling the dummy person according to a variety of situations. Methodology for such can be proposed, such as creating a separate questionnaire for the dummy with questions about the situation or event, and using an unbiased Artificial Intelligence model to generate the answers for those questions. Validation can be done using multiple AI models.

A viable method might be to choose 6 or 7 different situations and prepare a default questionnaire based on “threat level”, “comfort”, etc, for all the situations. This questionnaire can be used on different AI models with a properly structured prompt for each situation. Their answers can be modeled into the dummy, and the different answers from different AI models can be used as a means of validation for the outcomes. Further analysis is required on the development of the questionnaire.

References:

1. McEwen, B. S. (2007). Physiology and Neurobiology of Stress and Adaptation: Central Role of the Brain. *Physiological Reviews*, *87*(3), 873–904. https://doi.org/10.1152/physrev.00041.2006
2. Kao, Z. C. (n.d.). *CLASSICAL MECHANICS: THE THREE-BODY PROBLEM*.
3. Dongarra, J. J., Sorensen, D. C., & Hammarling, S. J. (1989). Block reduction of matrices to condensed forms for eigenvalue computations. *Journal of Computational and Applied Mathematics*, *27*(1–2), 215–227. https://doi.org/10.1016/0377-0427(89)90367-1
4. Tartakovsky E. (2023). The psychology of romantic relationships: motivations and mate preferences. *Frontiers in psychology*, *14*, 1273607. https://doi.org/10.3389/fpsyg.2023.1273607
5. Ybarra, O., Keller, M. C., Chan, E., Garcia, S. M., Sanchez-Burks, J., Morrison, K. R., & Baron, A. S. (2010). Being Unpredictable: Friend or Foe Matters: Friend or Foe Matters. *Social Psychological and Personality Science*, *1*(3), 259-267. https://doi.org/10.1177/1948550610370214 (Original work published 2010)
6. Vignesh, D., He, S., & Banerjee, S. (2025). A review on the complexities of brain activity: insights from nonlinear dynamics in neuroscience. In *Nonlinear Dynamics* (Vol. 113, Issue 5, pp. 4531–4552). Springer Science and Business Media B.V. https://doi.org/10.1007/s11071-024-10558-2
7. Lorenz, E. (2000). The butterfly effect. In *The chaos avant-garde: Memories of the early days of chaos theory* (pp. 91-94).
8. Ozdemir, O., Ozdemir, P. G., & Yilmaz, E. (2014). The butterfly effect in psychiatry: A case example. *Psychiatry and Behavioral Sciences*, *4*(1), 34.