Table of Contents

==========

==========

СЦЕНАРИЙ 2: «ОПЛАТА ПО СЧЕТАМ» - ОПТИМИЗИРОВАННАЯ ВЕРСИЯ
==========

```
clear all; close all; clc;
```

ПАРАМЕТРЫ СИМУЛЯЦИИ

```
R = 10;                     % Скорость команды
k = 0.3;                    % Генерация долга
alpha = 0.02;               % Рост долга
beta = 0.1;                 % Ресурсы на рефакторинг

T = 100;                    % Время симуляции
dt = 0.05;                  % Шаг времени
t = 0:dt:T;                 % Вектор времени
N = length(t);              % Количество точек
```

РАСЧЕТ ДИНАМИКИ

```
V = zeros(N, 1);            % Ценность
D = zeros(N, 1);            % Долг
P = zeros(N, 1);            % Производительность

% Начальные условия
V(1) = 0.1;
D(1) = 0;
P(1) = R;

% Расчет траектории
for i = 2:N
    % Уравнение долга с колебаниями
    dDdt = k*R - beta*R + alpha*D(i-1) + 0.1*sin(0.2*t(i));
```

```
    % Уравнение ценности
    eff_factor = 1 - 0.2*tanh(0.05*D(i-1));
    dVdt = R*(1 - beta) * eff_factor;

    % Производительность
    dPdt = 0.1*(R*(1 - beta)*(1 - 0.15*tanh(0.1*D(i-1))) - P(i-1));

    % Интеграция
    D(i) = D(i-1) + dt * dDdt;
    V(i) = V(i-1) + dt * dVdt;
    P(i) = P(i-1) + dt * dPdt;
end
```

ПОДГОТОВКА 3D ВИЗУАЛИЗАЦИИ

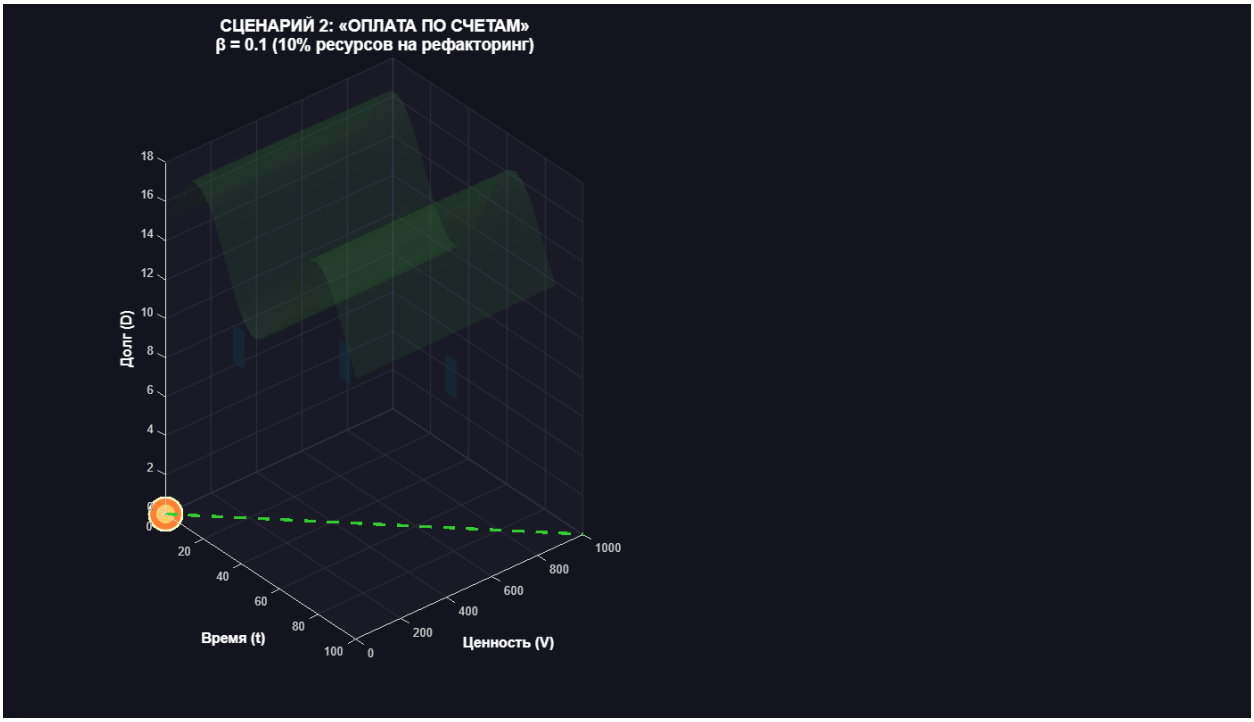
Создание сетки для фоновых эффектов

```
[X, Y] = meshgrid(linspace(0, T, 30), linspace(0, max(V)*1.2, 30));
Z_balance = 15 + 3*sin(0.1*X) .* cos(0.2*Y/max(V));

% Идеальная траектория
V_ideal = R * t;
```

СОЗДАНИЕ АНИМАЦИИ

```
fig = figure('Position', [100, 100, 1400, 800], 'Color', [0.08 0.08 0.12]);

% Основная 3D ось
ax_main = subplot(2,2,[1 3]);
set(ax_main, 'Color', [0.1 0.1 0.15], ...
             'GridColor', [0.3 0.3 0.4], ...
             'GridAlpha', 0.3);

hold on;

% 1. Фоновая поверхность баланса
surf_handle = surf(X, Y, Z_balance, ...
                   'FaceColor', [0.3 0.7 0.3], ...
                   'EdgeColor', 'none', ...
                   'FaceAlpha', 0.15);

% 2. Идеальная траектория
ideal_handle = plot3(t, V_ideal, zeros(size(t)), ...
                     'Color', [0.2 0.8 0.2], ...
                     'LineWidth', 2.5, ...
                     'LineStyle', '--');

% 3. Основная траектория
traj_handle = plot3(t(1), V(1), D(1), ...
                    'Color', [1 0.9 0.6], ...
                    'LineWidth', 3.5);

% 4. Текущая точка
```

```
point_outer = plot3(t(1), V(1), D(1), 'o', ...
                    'MarkerSize', 28, ...
                    'MarkerFaceColor', [1 0.5 0.2], ...
                    'MarkerEdgeColor', [1 1 0.8], ...
                    'LineWidth', 2);

point_inner = plot3(t(1), V(1), D(1), 'o', ...
                    'MarkerSize', 16, ...
                    'MarkerFaceColor', [1 0.8 0.4], ...
                    'MarkerEdgeColor', 'none');

% 5. Области рефакторинга
refactor_times = [20, 50, 80];
for rt = refactor_times
    [Xc, Yc, Zc] = cylinder(1, 20);
    Xc = Xc * 3 + rt;
    Yc = Yc * 2 + interp1(t, V, rt);
    Zc = Zc * 2 + 8;
    surf(Xc, Yc, Zc, ...
         'FaceColor', [0.2 0.7 0.9], ...
         'EdgeColor', 'none', ...
         'FaceAlpha', 0.2);
end

% Настройка вида
view(50, 25);
grid on;
xlabel('Время (t)', 'FontSize', 12, 'Color', 'white', 'FontWeight', 'bold');
ylabel('Ценность (V)', 'FontSize', 12, 'Color', 'white', 'FontWeight',
'bold');
zlabel('Долг (D)', 'FontSize', 12, 'Color', 'white', 'FontWeight', 'bold');
title({'СЦЕНАРИЙ 2: «ОПЛАТА ПО СЧЕТАМ»', ...
       'β = 0.1 (10% ресурсов на рефакторинг)'}, ...
       'FontSize', 14, 'Color', 'white', 'FontWeight', 'bold');

light('Position', [T, max(V), max(D)], 'Style', 'infinite');
light('Position', [0, 0, max(D)], 'Style', 'infinite');
lighting gouraud;
```

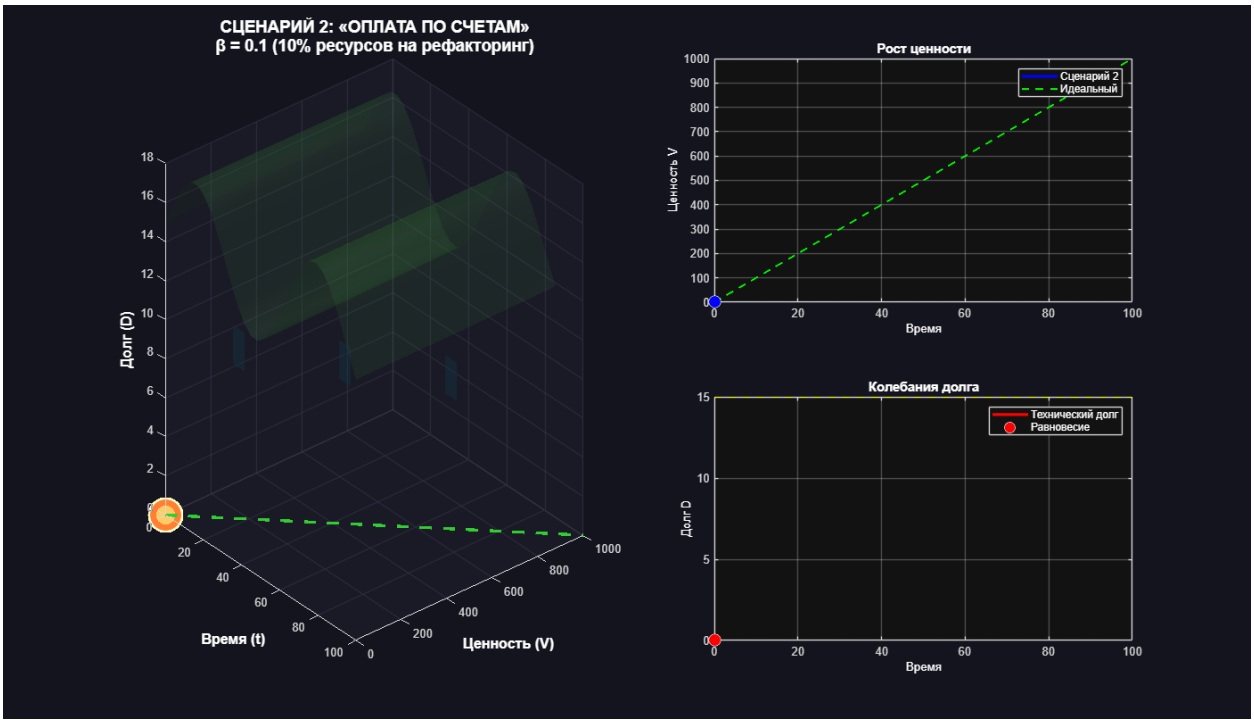


ДОПОЛНИТЕЛЬНЫЕ ГРАФИКИ

График ценности

```
ax_value = subplot(2,2,2);
set(ax_value, 'Color', [0.1 0.1 0.15], ...
              'XColor', 'white', 'YColor', 'white');

value_handle = plot(t(1), V(1), 'b-', 'LineWidth', 2.5);
hold on;
ideal_value_handle = plot(t, V_ideal, 'g--', 'LineWidth', 1.5);
value_point = plot(t(1), V(1), 'o', ...
                   'MarkerSize', 10, ...
                   'MarkerFaceColor', 'b', ...
                   'MarkerEdgeColor', 'white');

grid on;
xlabel('Время', 'FontSize', 10, 'Color', 'white');
ylabel('Ценность V', 'FontSize', 10, 'Color', 'white');
title('Рост ценности', 'FontSize', 11, 'Color', 'white', 'FontWeight',
'bold');
xlim([0 T]);
legend({'Сценарий 2', 'Идеальный'}, 'TextColor', 'white');

% График долга
ax_debt = subplot(2,2,4);
set(ax_debt, 'Color', [0.1 0.1 0.15], ...
             'XColor', 'white', 'YColor', 'white');

debt_handle = plot(t(1), D(1), 'r-', 'LineWidth', 2.5);
```

```
hold on;
debt_point = plot(t(1), D(1), 'o', ...
                  'MarkerSize', 10, ...
                  'MarkerFaceColor', 'r', ...
                  'MarkerEdgeColor', 'white');

% Линия равновесия
balance_line = plot([0 T], [15 15], 'y--', 'LineWidth', 1.5);

grid on;
xlabel('Время', 'FontSize', 10, 'Color', 'white');
ylabel('Долг D', 'FontSize', 10, 'Color', 'white');
title('Колебания долга', 'FontSize', 11, 'Color', 'white', 'FontWeight',
'bold');
xlim([0 T]);
legend({'Технический долг', 'Равновесие'}, 'TextColor', 'white');
```



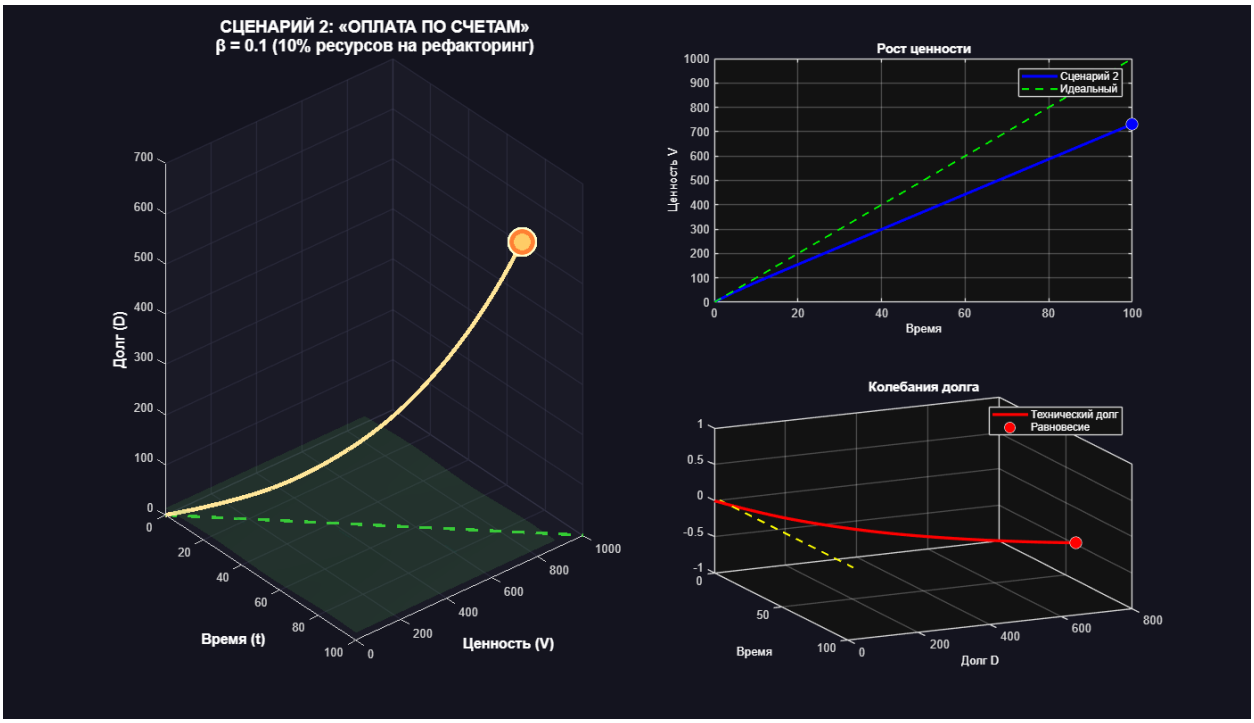
АНИМАЦИЯ

```
fprintf('Запуск анимации...\n');
pause(2);

% Параметры анимации
skip = 5;  % Пропуск кадров для скорости

for i = 1:skip:N
    % Обновление основной траектории
    set(traj_handle, 'XData', t(1:i), ...
                     'YData', V(1:i), ...
                     'ZData', D(1:i));
```

```
    % Пульсация точки
    pulse = 28 + 10*sin(i/20);
    set(point_outer, 'XData', t(i), ...
                     'YData', V(i), ...
                     'ZData', D(i), ...
                     'MarkerSize', pulse);

    set(point_inner, 'XData', t(i), ...
                     'YData', V(i), ...
                     'ZData', D(i), ...
                     'MarkerSize', pulse*0.6);

    % Волны на поверхности баланса
    wave_phase = 0.1*i/skip;
    Z_wave = 15 + 3*sin(0.1*X + wave_phase) .* cos(0.2*Y/max(V) +
wave_phase);
    set(surf_handle, 'ZData', Z_wave);

    % Динамический вид камеры
    view_angle = 50 + 15*sin(i/(skip*50));
    view_elev = 25 + 8*cos(i/(skip*80));
    view(view_angle, view_elev);

    % Обновление дополнительных графиков
    set(value_handle, 'XData', t(1:i), 'YData', V(1:i));
    set(value_point, 'XData', t(i), 'YData', V(i));

    set(debt_handle, 'XData', t(1:i), 'YData', D(1:i));
    set(debt_point, 'XData', t(i), 'YData', D(i));

    % Эффект при достижении равновесия
    if abs(D(i) - 15) < 1
        text(T*0.7, max(V)*0.8, max(D)*0.8, '⚖️ БАЛАНС ⚖️', ...
             'FontSize', 14, 'Color', [0.2 1 0.5], ...
             'FontWeight', 'bold');
    end

    % Эффект при рефакторинге
    for rt = refactor_times
        if abs(t(i) - rt) < 0.5
            flash = scatter3(ax_main, t(i), V(i), D(i), 300, ...
                             'o', 'MarkerFaceColor', [0.2 0.7 0.9], ...
                             'MarkerEdgeColor', 'none', 'MarkerFaceAlpha',
0.3);
            drawnow;
            delete(flash);
        end
    end

    drawnow;

    % Адаптивная скорость
    if i < N/4
```

```
        pause(0.03);
    elseif i < 3*N/4
        pause(0.015);
    else
        pause(0.008);
    end
end
```

Запуск анимации...

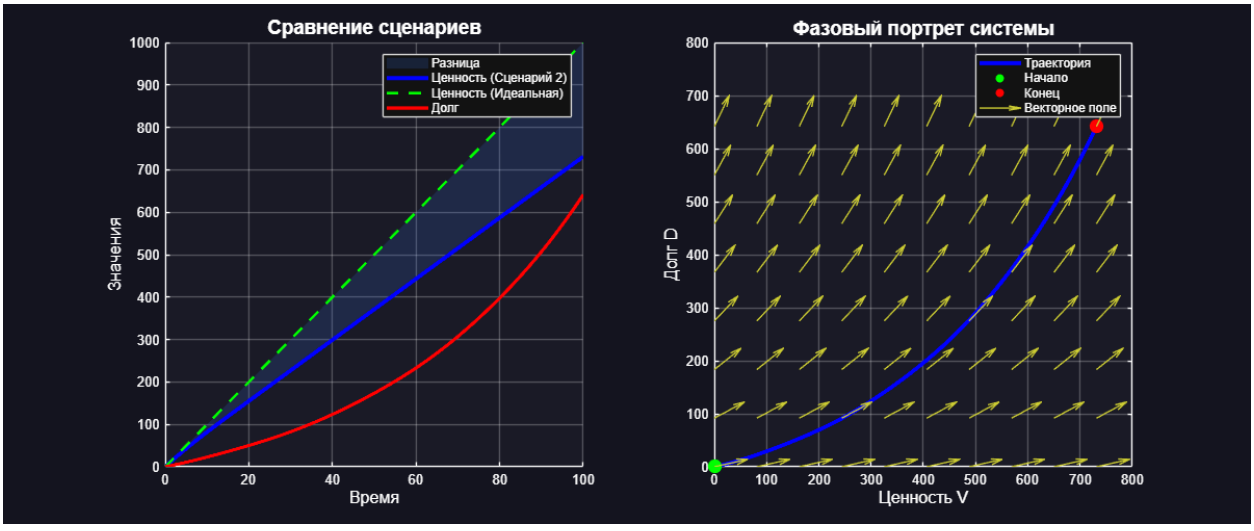


ФИНАЛЬНАЯ ВИЗУАЛИЗАЦИЯ

```
fprintf('Анимация завершена!\n\n');

% Создание итогового графика
final_fig = figure('Position', [200, 200, 1200, 500], 'Color', [0.08 0.08
0.12]);

% Сравнение траекторий
subplot(1,2,1);
hold on;

% Заполнение между кривыми
fill([t, fliplr(t)], [V', fliplr(V_ideal)], [0.2 0.4 0.8], 'FaceAlpha', 0.2,
'EdgeColor', 'none');
plot(t, V, 'b-', 'LineWidth', 3);
plot(t, V_ideal, 'g--', 'LineWidth', 2);
plot(t, D, 'r-', 'LineWidth', 2.5);

grid on;
```

```
xlabel('Время', 'FontSize', 12, 'Color', 'white');
ylabel('Значения', 'FontSize', 12, 'Color', 'white');
title('Сравнение сценариев', 'FontSize', 14, 'Color', 'white', 'FontWeight',
'bold');
legend({'Разница', 'Ценность (Сценарий 2)', 'Ценность (Идеальная)', 'Долг'},
...
       'TextColor', 'white');
set(gca, 'Color', [0.1 0.1 0.15], 'XColor', 'white', 'YColor', 'white');

% Фазовый портрет
subplot(1,2,2);
plot(V, D, 'b-', 'LineWidth', 3);
hold on;
scatter(V(1), D(1), 100, 'go', 'filled', 'LineWidth', 2);
scatter(V(end), D(end), 100, 'ro', 'filled', 'LineWidth', 2);

% Векторное поле
[Vg, Dg] = meshgrid(linspace(0, max(V), 10), linspace(0, max(D), 8));
dV = R*(1-beta)*(1 - 0.2*tanh(0.05*Dg));
dD = k*R - beta*R + alpha*Dg;
L = sqrt(dV.^2 + dD.^2);

quiver(Vg, Dg, dV./L, dD./L, 0.6, 'Color', [0.8 0.8 0.2], 'LineWidth', 1);

grid on;
xlabel('Ценность V', 'FontSize', 12, 'Color', 'white');
ylabel('Долг D', 'FontSize', 12, 'Color', 'white');
title('Фазовый портрет системы', 'FontSize', 14, 'Color', 'white',
'FontWeight', 'bold');
legend({'Траектория', 'Начало', 'Конец', 'Векторное поле'}, 'TextColor',
'white');
set(gca, 'Color', [0.1 0.1 0.15], 'XColor', 'white', 'YColor', 'white');
```

Анимация завершена!



ИТОГОВЫЕ ПОКАЗАТЕЛИ

```
fprintf('================================================\n');
fprintf('ИТОГОВЫЕ ПОКАЗАТЕЛИ СЦЕНАРИЯ 2:\n');
fprintf('================================================\n');
fprintf('Финальная ценность: %.1f (%.1f%% от идеальной)\n', V(end), V(end)/
V_ideal(end)*100);
fprintf('Финальный долг: %.1f\n', D(end));
fprintf('Средний долг: %.1f\n', mean(D));
fprintf('Амплитуда колебаний долга: %.1f\n', max(D)-min(D));
fprintf('Отношение D/V: %.3f\n', D(end)/V(end));
fprintf('Производительность: %.1f (%.1f%% от максимальной)\n', P(end),
P(end)/R*100);
fprintf('\nВЫВОД: Система поддерживает равновесие долга\n');
fprintf('при почти линейном росте ценности.\n');
```

```
================================================
ИТОГОВЫЕ ПОКАЗАТЕЛИ СЦЕНАРИЯ 2:
================================================
Финальная ценность: 731.1 (73.1% от идеальной)
Финальный долг: 641.6
Средний долг: 220.8
Амплитуда колебаний долга: 641.6
Отношение D/V: 0.878
Производительность: 7.7 (76.5% от максимальной)

ВЫВОД: Система поддерживает равновесие долга
при почти линейном росте ценности.
```

Published with MATLAB® R2025b