### Каскадный ПИ-ПИ регулятор

Каскадный ПИ-ПИ регулятор состоит из двух уровней регуляторов, где первый регулятор (внутренний контур) управляет промежуточной переменной (например, температурой рубашки), а второй (внешний контур) управляет основной переменной системы (например, температурой рабочей камеры). В каскадной системе внешний регулятор рассчитывает уставку для внутреннего регулятора, а внутренний регулятор непосредственно управляет системой.

Для этой системы я добавлю каскадную структуру регуляторов:

* Внешний контур будет регулировать температуру рабочей камеры (переменная ) с помощью ПИ-регулятора.
* Внутренний контур будет регулировать температуру рубашки (переменная ) с помощью ПИ-регулятора, который будет управлять подачей тепла.

**Общая схема каскадного ПИ-ПИ-регулятора:**

1. **Внешний контур (регулятор температуры рабочей камеры)**:

* Заданная температура рабочей камеры ​ — это уставка для внешнего контура.
* Внешний ПИ-регулятор рассчитывает уставку для температуры рубашки

1. **Внутренний контур (регулятор температуры рубашки)**:

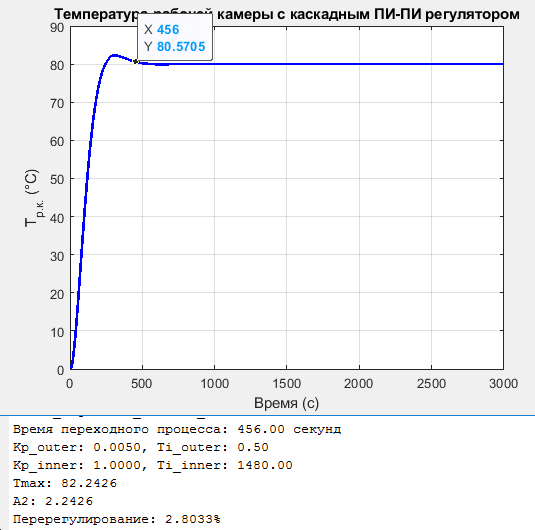
* Уставка температуры рубашки поступает от внешнего регулятора.
* Внутренний ПИ-регулятор управляет подачей тепла на рубашку, чтобы достичь заданной температуры рубашки.

### По уставке:

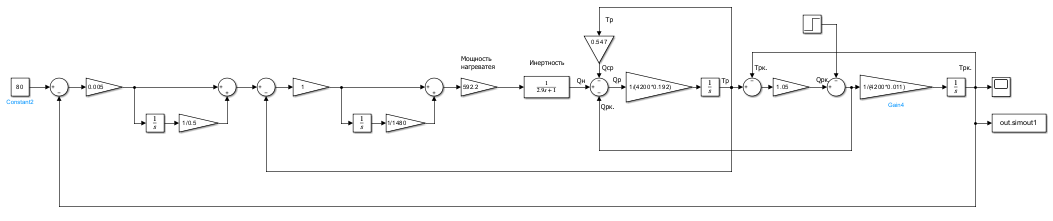
### Код каскадного ПИ-ПИ регулятора, коэффициенты вручную

|  |
| --- |
| function PI\_regulator\_ustavka\_nonauto()  % Время моделирования  t\_total = 3000; % Общее время моделирования (в секундах)  h = 1; % Шаг по времени (в секундах)  t = 0:h:t\_total; % Вектор времени  N = length(t); % Количество шагов на основе времени моделирования и шага  % Параметры системы  c = 4200; % Удельная теплоёмкость воды, Дж/(кг·°C)  mp = 0.192; % Масса воды в рубашке, кг  mp\_k = 0.011; % Масса воды в рабочей камере, кг  k1 = 0.547; % Коэффициент теплопередачи в среду  k2 = 1.05; % Коэффициент теплопередачи между рубашкой и камерой  T = 2.9; % Постоянная времени системы  k = 592.2; % Коэффициент управляющего воздействия  setpoint = 80; % Уставка — желаемая температура  % Начальные условия [T\_р.к., T\_p, Q\_H]  x = zeros(3, N); % Массив для хранения результатов  x(:,1) = [0; 0; 0]; % Начальные температуры и теплота  % Матрицы системы  A = [-k2/(c\*mp\_k), k2/(c\*mp\_k), 0;  k2/(c\*mp), -(k2/(c\*mp) + k1/(c\*mp)), 1/(c\*mp);  0, 0, -1/T];  B = [0; 0; k/T];  % Переменные для вычисления переходного времени и перерегулирования  transition\_time = NaN; % Время переходного процесса  derivative\_threshold = 0.01; % Порог производной для установившегося состояния  steady\_state\_criteria = 0.02; % Критерий ±1% от уставки  % Счетчик для проверки условия производной  derivative\_count = 0;  duration\_threshold = 1000; % Продолжительность для проверки  % Параметры ПИ-регуляторов для каскадного управления  Kp\_outer = 0.005; % Пропорциональный коэффициент внешнего контура  Ti\_outer = 0.5; % Интегральный коэффициент внешнего контура  Kp\_inner = 1; % Пропорциональный коэффициент внутреннего контура  Ti\_inner = 1480; % Интегральный коэффициент внутреннего контура  error\_integral\_outer = 0; % Интеграл ошибки внешнего контура  error\_integral\_inner = 0; % Интеграл ошибки внутреннего контура  u\_inner = ones(1, N); % Управляющее воздействие внутреннего контура  % Моделирование системы  for i = 1:N-1  % Внешний контур: расчёт ошибки и интеграла ошибки для температуры рабочей камеры  error\_outer = setpoint - x(1,i);  error\_integral\_outer = error\_integral\_outer + error\_outer \* h;  u\_outer = Kp\_outer \* (error\_outer + (1/Ti\_outer) \* error\_integral\_outer);  % Внутренний контур: расчёт ошибки для внутреннего ПИ-регулятора  error\_inner = u\_outer - x(2,i);  error\_integral\_inner = error\_integral\_inner + error\_inner \* h;  u\_inner(i) = Kp\_inner \* (error\_inner + (1/Ti\_inner) \* error\_integral\_inner);  % Дискретное моделирование  x(:,i+1) = (A\*x(:,i) + B\*u\_inner(i)) \* h + x(:,i);  % Проверка установившегося состояния  if isnan(transition\_time)  temperature\_derivative = abs(x(1,i+1) - x(1,i)) / h;    % Проверка на отклонение в зависимости от значения уставки  if setpoint == 0  % Если уставка 0, используем абсолютное отклонение  deviation\_met = abs(x(1,i+1)) < steady\_state\_criteria;  else  % Если уставка не 0, используем относительное отклонение  deviation\_met = abs(x(1,i+1) - setpoint) < steady\_state\_criteria \* abs(setpoint);  end  % Проверка на установившееся состояние  if deviation\_met && temperature\_derivative < derivative\_threshold  derivative\_count = derivative\_count + 1;  if derivative\_count >= duration\_threshold  transition\_time = t(i) - duration\_threshold; % Время переходного процесса с момента возмущения  end  else  derivative\_count = 0; % Сброс счётчика, если условие не выполняется  end  end  end  % Расчет перерегулирования  Tmax = max(x(1,:));  A2 = Tmax - setpoint; % Максимальное отклонение от уставки  overshoot\_ratio = (A2 / setpoint) \* 100; % Перерегулирование в процентах  if overshoot\_ratio >= 4  fprintf('Система не сбалансировалась\n');  else  fprintf('Время переходного процесса: %.2f секунд\n', transition\_time);  fprintf('Kp\_outer: %.4f, Ti\_outer: %.2f\n', Kp\_outer, Ti\_outer);  fprintf('Kp\_inner: %.4f, Ti\_inner: %.2f\n', Kp\_inner, Ti\_inner);  fprintf('Tmax: %.4f\n', Tmax);  fprintf('A2: %.4f\n', A2);  fprintf('Перерегулирование: %.4f%%\n', overshoot\_ratio);  % Построение графика  plot(t, x(1,:), 'b', 'LineWidth', 2);  title('Температура рабочей камеры с каскадным ПИ-ПИ регулятором');  xlabel('Время (с)');  ylabel('T\_{р.к.} (°C)');  grid on;  end  end |

Код каскадного ПИ-ПИ регулятора без перебора коэффициентов, с подсчётом времени переходного процесса и проверкой перерегулирования.



Модель системы с каскадным ПИ-ПИ регулятором



Сравнение графиков с Simulink и скрипта:



### Код с подсчётом параметров автоматически, каскадного ПИ-ПИ регулятора

|  |
| --- |
| function PI\_regulator\_ustavka\_nonauto()  % Время моделирования  t\_total = 5000; % Общее время моделирования (в секундах)  h = 1; % Шаг по времени (в секундах)  t = 0:h:t\_total; % Вектор времени  N = length(t); % Количество шагов на основе времени моделирования и шага  % Параметры системы  c = 4200; % Удельная теплоёмкость воды, Дж/(кг·°C)  mp = 0.192; % Масса воды в рубашке, кг  mp\_k = 0.011; % Масса воды в рабочей камере, кг  k1 = 0.547; % Коэффициент теплопередачи в среду  k2 = 1.05; % Коэффициент теплопередачи между рубашкой и камерой  T = 2.9; % Постоянная времени системы  k = 1; % Коэффициент управляющего воздействия  setpoint = 80; % Уставка — желаемая температура  % Начальные условия [T\_р.к., T\_p, Q\_H]  x = zeros(3, N); % Массив для хранения результатов  x(:,1) = [0; 0; 0]; % Начальные температуры и теплота  % Матрицы системы  A = [-k2/(c\*mp\_k), k2/(c\*mp\_k), 0;  k2/(c\*mp), -(k2/(c\*mp) + k1/(c\*mp)), 1/(c\*mp);  0, 0, -1/T];  B = [0; 0; k/T];  % Переменные для вычисления переходного времени и перерегулирования  derivative\_threshold = 0.01; % Порог производной для установившегося состояния  steady\_state\_criteria = 0.02; % Критерий ±1% от уставки  % Счетчик для проверки условия производной  derivative\_count = 0;  duration\_threshold = 1000; % Продолжительность для проверки  % Параметры поиска лучших коэффициентов  best\_time = Inf; % Лучшее время переходного процесса  best\_Kp\_outer = 0;  best\_Ti\_outer = 0;  best\_Kp\_inner = 0;  best\_Ti\_inner = 0;  best\_overshoot = Inf; % Лучшее перерегулирование  best\_tmax = Inf;  % Параметры для поиска  Kp\_values\_outer = 0.0001:0.0001:0.05; % Диапазон значений Kp внешнего контура  Ti\_values\_outer = 0.01:0.01:1; % Диапазон значений Ti внешнего контура  Kp\_values\_inner = 0:0.05:2; % Диапазон значений Kp внутреннего контура  Ti\_values\_inner = 700:5:1500; % Диапазон значений Ti внутреннего контура  % Начало цикла перебора коэффициентов  for Kp\_outer = Kp\_values\_outer  for Ti\_outer = Ti\_values\_outer  for Kp\_inner = Kp\_values\_inner  for Ti\_inner = Ti\_values\_inner  error\_integral\_outer = 0; % Интеграл ошибки внешнего контура  error\_integral\_inner = 0; % Интеграл ошибки внутреннего контура  u\_inner = ones(1, N); % Управляющее воздействие внутреннего контура  transition\_time = NaN; % Время переходного процесса  % Моделирование системы  for i = 1:N-1  % Внешний контур: расчёт ошибки и интеграла ошибки для температуры рабочей камеры  error\_outer = setpoint - x(1,i);  error\_integral\_outer = error\_integral\_outer + error\_outer \* h;  u\_outer = Kp\_outer \* (error\_outer + (1/Ti\_outer) \* error\_integral\_outer);  % Внутренний контур: расчёт ошибки для внутреннего ПИ-регулятора  error\_inner = u\_outer - x(2,i);  error\_integral\_inner = error\_integral\_inner + error\_inner \* h;  u\_inner(i) = Kp\_inner \* (error\_inner + (1/Ti\_inner) \* error\_integral\_inner);  % Дискретное моделирование  x(:,i+1) = (A\*x(:,i) + B\*u\_inner(i)) \* h + x(:,i);    % Проверка установившегося состояния  if isnan(transition\_time)  temperature\_derivative = abs(x(1,i+1) - x(1,i)) / h;    % Проверка на отклонение в зависимости от значения уставки  if setpoint == 0  % Если уставка 0, используем абсолютное отклонение  deviation\_met = abs(x(1,i+1)) < steady\_state\_criteria;  else  % Если уставка не 0, используем относительное отклонение  deviation\_met = abs(x(1,i+1) - setpoint) < steady\_state\_criteria \* abs(setpoint);  end    % Проверка на установившееся состояние  if deviation\_met && temperature\_derivative < derivative\_threshold  derivative\_count = derivative\_count + 1;  if derivative\_count >= duration\_threshold  transition\_time = t(i) - duration\_threshold; % Время переходного процесса с момента возмущения  end  else  derivative\_count = 0; % Сброс счётчика, если условие не выполняется  end  end  end  % Расчет перерегулирования  Tmax = max(x(1,:));  A2 = Tmax - setpoint; % Максимальное отклонение от уставки  overshoot\_ratio = (A2 / setpoint) \* 100; % Перерегулирование в процентах  % Обновление лучших параметров при соблюдении критериев  if ~isnan(transition\_time) && transition\_time < best\_time && overshoot\_ratio <= 4  best\_time = transition\_time;  best\_Kp\_outer = Kp\_outer;  best\_Ti\_outer = Ti\_outer;  best\_Kp\_inner = Kp\_inner;  best\_Ti\_inner = Ti\_inner;  best\_A2 = A2;  best\_tmax = Tmax;  best\_overshoot = overshoot\_ratio;  best\_response = x(1,:); % Сохраняем ответ для построения графика  end  end  end  end  end  if best\_overshoot >= 4  fprintf('Система не сбалансировалась\n');  else  fprintf('Лучшее время переходного процесса: %.2f секунд\n', best\_time);  fprintf('Лучший Kp\_outer: %.4f\n', best\_Kp\_outer);  fprintf('Лучший Ti\_outer: %.2f\n', best\_Ti\_outer);  fprintf('Лучший Kp\_inner: %.4f\n', best\_Kp\_inner);  fprintf('Лучший Ti\_inner: %.2f\n', best\_Ti\_inner);  fprintf('Tmax: %.4f\n', best\_tmax);  fprintf('A2: %.4f\n', best\_A2);  fprintf('Перерегулирование: %.4f%%\n', best\_overshoot);  % Построение графика с наилучшими коэффициентами  figure;  plot(t, best\_response, 'r', 'LineWidth', 2);  title('Температура рабочей камеры с каскадным ПИ-ПИ регулятором');  xlabel('Время (с)');  ylabel('T\_{р.к.} (°C)');  grid on;  end  end |

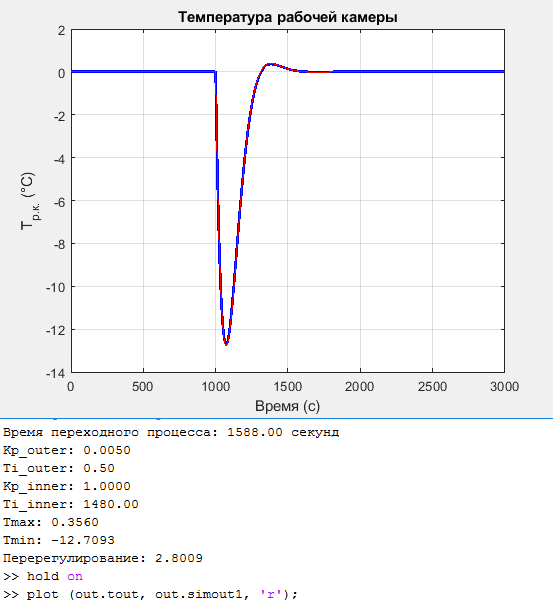
Нужно посчитать коэффициенты, за 9 часов 32 000 000 итераций не посчиталось

(

### По возмущению:

* 1. **Код каскадного ПИ-ПИ регулятора с возмущением, коэффициенты вручную**

|  |
| --- |
| function PI\_regulator\_disturbance2()  % Время моделирования  t\_total = 3000; % Общее время моделирования (в секундах)  h = 1; % Шаг по времени (в секундах)  t = 0:h:t\_total; % Вектор времени  N = length(t); % Количество шагов на основе времени моделирования и шага  % Параметры системы  c = 4200; % Удельная теплоёмкость воды, Дж/(кг·°C)  mp = 0.192; % Масса воды в рубашке, кг  mp\_k = 0.011; % Масса воды в рабочей камере, кг  k1 = 0.547; % Коэффициент теплопередачи в среду  k2 = 1.05; % Коэффициент теплопередачи между рубашкой и камерой  T = 2.9; % Постоянная времени системы  k = 592.2; % Коэффициент управляющего воздействия  setpoint = 0; % Уставка — желаемая температура  % Начальные условия [T\_р.к., T\_p, Q\_H]  x = zeros(3, N); % Массив для хранения результатов  x(:,1) = [0; 0; 0]; % Начальные температуры и теплота  % Матрицы системы  A = [-k2/(c\*mp\_k), k2/(c\*mp\_k), 0;  k2/(c\*mp), -(k2/(c\*mp) + k1/(c\*mp)), 1/(c\*mp);  0, 0, -1/T];  B = [0; 0; k/T];  F = [-1/(c\*mp\_k); 1/(c\*mp); 0];  % Переменные для вычисления переходного времени и перерегулирования  transition\_time = NaN; % Время переходного процесса  derivative\_threshold = 0.01; % Порог производной для установившегося состояния  steady\_state\_criteria = 0.02; % Критерий ±1% от уставки  % Счетчик для проверки условия производной  derivative\_count = 0;  duration\_threshold = 1000; % Продолжительность для проверки  % Возмущение: изменение температуры окружающей среды  f = zeros(1, N);  start\_disturbance = 1000;  f(start\_disturbance:end) = 20;  % Параметры ПИ-регуляторов для каскадного управления  Kp\_outer = 0.005; % Пропорциональный коэффициент внешнего контура  Ti\_outer = 0.5; % Интегральный коэффициент внешнего контура  Kp\_inner = 1; % Пропорциональный коэффициент внутреннего контура  Ti\_inner = 1480; % Интегральный коэффициент внутреннего контура    % Параметры ПИ-регуляторов для каскадного управления  error\_integral\_outer = 0; % Интеграл ошибки внешнего контура  error\_integral\_inner = 0; % Интеграл ошибки внутреннего контура  u\_inner = ones(1, N); % Управляющее воздействие внутреннего контура  % Моделирование системы  for i = 1:N-1  % Внешний контур: расчёт ошибки и интеграла ошибки для температуры рабочей камеры  error\_outer = setpoint - x(1,i);  error\_integral\_outer = error\_integral\_outer + error\_outer \* h;  u\_outer = Kp\_outer \* (error\_outer + (1/Ti\_outer) \* error\_integral\_outer);  % Внутренний контур: расчёт ошибки для внутреннего ПИ-регулятора  error\_inner = u\_outer - x(2,i);  error\_integral\_inner = error\_integral\_inner + error\_inner \* h;  u\_inner(i) = Kp\_inner \* (error\_inner + (1/Ti\_inner) \* error\_integral\_inner);  % Дискретное моделирование  x(:,i+1) = (A\*x(:,i) + B\*u\_inner(i) + F\*f(i)) \* h + x(:,i);  % Проверка установившегося состояния  if i >= start\_disturbance && isnan(transition\_time)  % Рассчитываем производную температуры как разность текущей и предыдущей  temperature\_derivative = abs(x(1,i+1) - x(1,i)) / h;    % Проверка на отклонение в зависимости от значения уставки  if setpoint == 0  % Если уставка 0, используем абсолютное отклонение  deviation\_met = abs(x(1,i+1)) < steady\_state\_criteria;  else  % Если уставка не 0, используем относительное отклонение  deviation\_met = abs(x(1,i+1) - setpoint) < steady\_state\_criteria \* abs(setpoint);  end    % Проверка на установившееся состояние  if deviation\_met && temperature\_derivative < derivative\_threshold  derivative\_count = derivative\_count + 1;  if derivative\_count >= duration\_threshold  transition\_time = t(i) - t(start\_disturbance); % Время переходного процесса с момента возмущения  end  else  derivative\_count = 0; % Сброс счётчика, если условие не выполняется  end  end  end  % Расчет перерегулирования  Tmax = max(x(1, start\_disturbance:end));  Tmin = min(x(1, start\_disturbance:end));  A2 = Tmax - setpoint;  A1 = Tmin - setpoint;  overshoot\_ratio = abs(A2 / A1) \* 100;    if overshoot\_ratio >= 4  fprintf('Система не сбалансировалась\n');  else  % Вывод наилучших коэффициентов, времени переходного процесса и перерегулирования  fprintf('Лучшее время переходного процесса: %.2f секунд\n', transition\_time);  fprintf('Kp\_outer: %.4f\n', Kp\_outer);  fprintf('Ti\_outer: %.2f\n', Ti\_outer);  fprintf('Kp\_inner: %.4f\n', Kp\_inner);  fprintf('Ti\_inner: %.2f\n', Ti\_inner);  fprintf('Tmax: %.4f\n', Tmax);  fprintf('Tmin: %.4f\n', Tmin);  fprintf('deviation\_met: %.4f\n', deviation\_met);  fprintf('Перерегулирование: %.4f\n', overshoot\_ratio);  % Построение графика с наилучшими коэффициентами  figure;  plot(t, x(1,:), 'b', 'LineWidth', 2);  title('Температура рабочей камеры');  xlabel('Время (с)');  ylabel('T\_{р.к.} (°C)');  grid on;  end  end |



С Simulink совпадает

* 1. **Код каскадного ПИ-ПИ регулятора с возмущением, коэффициенты автомтически**

|  |
| --- |
| function PI\_regulator\_disturbance\_auto1()  tic  % Время моделирования  t\_total = 3000; % Общее время моделирования (в секундах)  h = 1; % Шаг по времени (в секундах)  t = 0:h:t\_total; % Вектор времени  N = length(t); % Количество шагов на основе времени моделирования и шага  % Параметры системы  c = 4200; % Удельная теплоёмкость воды, Дж/(кг·°C)  mp = 0.192; % Масса воды в рубашке, кгww  mp\_k = 0.011; % Масса воды в рабочей камере, кг  k1 = 0.547; % Коэффициент теплопередачи в среду  k2 = 1.05; % Коэффициент теплопередачи между рубашкой и камерой  T = 2.9; % Постоянная времени системы  k = 592.2; % Коэффициент управляющего воздействия  setpoint = 0; % Уставка — желаемая температура  % Начальные условия [T\_р.к., T\_p, Q\_H]  x = zeros(3, N); % Массив для хранения результатов  x(:,1) = [0; 0; 0]; % Начальные температуры и теплота  % Матрицы системы  A = [-k2/(c\*mp\_k), k2/(c\*mp\_k), 0;  k2/(c\*mp), -(k2/(c\*mp) + k1/(c\*mp)), 1/(c\*mp);  0, 0, -1/T];  B = [0; 0; k/T];  F = [-1/(c\*mp\_k); 1/(c\*mp); 0];  % Переменные для вычисления переходного времени и перерегулирования  derivative\_threshold = 0.01; % Порог производной для установившегося состояния  steady\_state\_criteria = 0.02; % Критерий ±1% от уставки  % Счетчик для проверки условия производной  derivative\_count = 0;  duration\_threshold = 1000; % Продолжительность для проверки  % Возмущение: изменение температуры окружающей среды  f = zeros(1, N);  start\_disturbance = 1000;  f(start\_disturbance:end) = 20;  % Параметры поиска лучших коэффициентов  best\_time = Inf; % Лучшее время переходного процесса  best\_Kp\_outer = 0;  best\_Ti\_outer = 0;  best\_Kp\_inner = 0;  best\_Ti\_inner = 0;  best\_overshoot = Inf; % Лучшее перерегулирование  best\_tmax = Inf;  % Параметры для поиска  Kp\_values\_outer = 0.0001:0.0001:0.05; % Диапазон значений Kp внешнего контура  Ti\_values\_outer = 0.01:0.01:1; % Диапазон значений Ti внешнего контура  Kp\_values\_inner = 0:0.05:2; % Диапазон значений Kp внутреннего контура  Ti\_values\_inner = 700:5:1500; % Диапазон значений Ti внутреннего контура  % Начало цикла перебора коэффициентов  for Kp\_outer = Kp\_values\_outer  for Ti\_outer = Ti\_values\_outer  for Kp\_inner = Kp\_values\_inner  for Ti\_inner = Ti\_values\_inner  error\_integral\_outer = 0; % Интеграл ошибки внешнего контура  error\_integral\_inner = 0; % Интеграл ошибки внутреннего контура  u\_inner = ones(1, N); % Управляющее воздействие внутреннего контура  transition\_time = NaN; % Время переходного процесса  % Моделирование системы  for i = 1:N-1  % Внешний контур: расчёт ошибки и интеграла ошибки для температуры рабочей камеры  error\_outer = setpoint - x(1,i);  error\_integral\_outer = error\_integral\_outer + error\_outer \* h;  u\_outer = Kp\_outer \* (error\_outer + (1/Ti\_outer) \* error\_integral\_outer);    % Внутренний контур: расчёт ошибки для внутреннего ПИ-регулятора  error\_inner = u\_outer - x(2,i);  error\_integral\_inner = error\_integral\_inner + error\_inner \* h;  u\_inner(i) = Kp\_inner \* (error\_inner + (1/Ti\_inner) \* error\_integral\_inner);    % Дискретное моделирование  x(:,i+1) = (A\*x(:,i) + B\*u\_inner(i) + F\*f(i)) \* h + x(:,i);  % Проверка установившегося состояния    if i >= start\_disturbance && isnan(transition\_time)  % Рассчитываем производную температуры как разность текущей и предыдущей  temperature\_derivative = abs(x(1,i+1) - x(1,i)) / h;    % Проверка на отклонение в зависимости от значения уставки  if setpoint == 0  % Если уставка 0, используем абсолютное отклонение  deviation\_met = abs(x(1,i+1)) < steady\_state\_criteria;  else  % Если уставка не 0, используем относительное отклонение  deviation\_met = abs(x(1,i+1) - setpoint) < steady\_state\_criteria \* abs(setpoint);  end    % Проверка на установившееся состояние  if deviation\_met && temperature\_derivative < derivative\_threshold  derivative\_count = derivative\_count + 1;  if derivative\_count >= duration\_threshold  transition\_time = t(i) - t(start\_disturbance); % Время переходного процесса  end  else  derivative\_count = 0; % Сброс счётчика, если условие не выполняется  end  end  end  % Расчет перерегулирования  Tmax = max(x(1, start\_disturbance:end));  Tmin = min(x(1, start\_disturbance:end));  A2 = Tmax - setpoint;  A1 = Tmin - setpoint;  overshoot\_ratio = abs(A2 / A1) \* 100;  % Обновление лучших параметров при соблюдении критериев  if ~isnan(transition\_time) && transition\_time < best\_time && overshoot\_ratio <= 4  best\_time = transition\_time;  best\_Kp\_outer = Kp\_outer;  best\_Ti\_outer = Ti\_outer;  best\_Kp\_inner = Kp\_inner;  best\_Ti\_inner = Ti\_inner;  best\_A2 = A2;  best\_A1 = A1;  best\_tmax = Tmax;  best\_tmin = Tmax;  best\_overshoot = overshoot\_ratio;  best\_response = x(1,:); % Сохраняем ответ для построения графика  end  end  end  end  end    if overshoot\_ratio >= 4  fprintf('Система не сбалансировалась\n');  else  % Вывод наилучших коэффициентов, времени переходного процесса и перерегулирования    fprintf('Лучшее время переходного процесса: %.2f секунд\n', best\_time);  fprintf('Лучший Kp\_outer: %.4f\n', best\_Kp\_outer);  fprintf('Лучший Ti\_outer: %.2f\n', best\_Ti\_outer);  fprintf('Лучший Kp\_inner: %.4f\n', best\_Kp\_inner);  fprintf('Лучший Ti\_inner: %.2f\n', best\_Ti\_inner);  fprintf('Tmax: %.4f\n', best\_tmax);  fprintf('Tmin: %.4f\n', best\_tmin);  fprintf('A2: %.4f\n', best\_A2);  fprintf('A1: %.4f\n', best\_A1);  fprintf('Перерегулирование: %.4f%%\n', best\_overshoot);  % Построение графика с наилучшими коэффициентами  figure;  plot(t, best\_response, 'b', 'LineWidth', 2);  title('Температура рабочей камеры');  xlabel('Время (с)');  ylabel('T\_{р.к.} (°C)');  grid on;  end  toc  end |