**Оглавление**

[Математическая модель химического реактора с водяной рубашкой: 2](#_Toc181107107)

[По уставке: 3](#_Toc181107108)

[1. Код моделирования системы (без регуляторов) 3](#_Toc181107109)

[2. Код моделирования системы с ПИ регулятором. Коэффициенты вручную. С подсчётом времени переходного процесса через производную. 4](#_Toc181107110)

[3. Готовый код моделирования с подбором коэффициентов через цикл for. 5](#_Toc181107111)

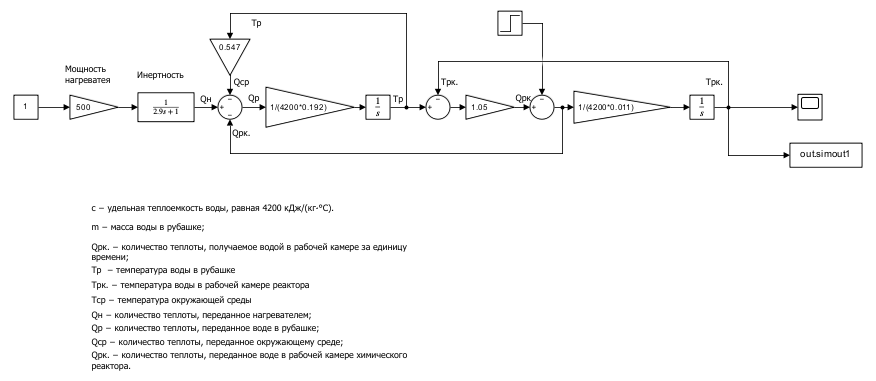
[По возмущению 8](#_Toc181107112)

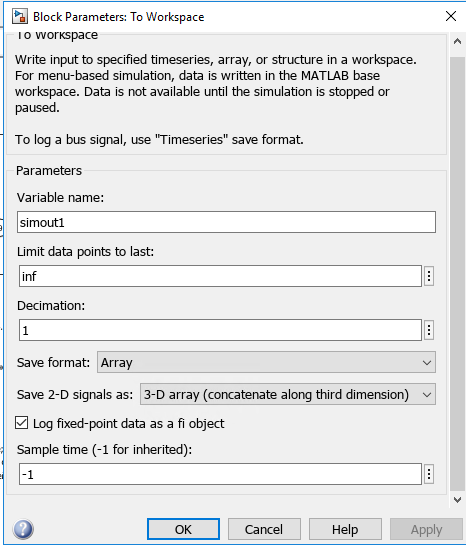
[1. Код моделирования системы с возмущением для сравнения со значениями simulink 8](#_Toc181107113)

[2. Код моделирования системы с возмущением, вручную и подсчётом перерегулирования и времени переходного процесса 9](#_Toc181107114)

[3. Готовый код моделирования системы с возмущением и перебором коэффициентов 11](#_Toc181107115)

### Математическая модель химического реактора с водяной рубашкой:





### По уставке:

### Код моделирования системы (без регуляторов)

|  |
| --- |
| function system  % 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.  % Количество элементов в векторе t. Поскольку вектор t содержит все временные точки симуляции, его длина соответствует количеству временных шагов 𝑁    % 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  % 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    % 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];    % Управляющее воздействие (постоянное для примера)  u = ones(1, N); % Мощность нагревателя  % Функция ones в MATLAB создаёт матрицу или вектор, заполненный единицами.  % В данном случае ones(1, N) создаёт вектор длины 𝑁 (то есть количество шагов симуляции), заполненный единицами.    % Дискретное моделирование методом Эйлера  for i = 1:N-1  x(:,i+1) = (A \* x(:,i) + B \* u(i)) \* h + x(:,i);  end  % Построение графиков  figure;  subplot(1,1,1);  plot(t, x(1,:), 'r', 'LineWidth', 2);  title('Температура рабочей камеры T\_{р.к.}(t) (X1)');  xlabel('Время (с)');  ylabel('T\_{р.к.} (°C)');  grid on;    % subplot(3,1,2);  % plot(t, x(2,:), 'b', 'LineWidth', 2);  % title('Температура рубашки T\_p(t) (X2)');  % xlabel('Время (с)');  % ylabel('T\_p (°C)');  % grid on;  %  % subplot(3,1,3);  % plot(t, x(3,:), 'g', 'LineWidth', 2);  % title('Количество теплоты Q\_H(t) (X3)');  % xlabel('Время (с)');  % ylabel('Q\_H (Дж)');  % grid on;  end |

Взято из ВКР Бутузова: Коэффициенты теплопередачи 𝑘1 = 0,547 и 𝑘2 = 1,05. Мощность k = 1973. Было взято 30% коэффициент мощности k = 592.2. В данной системе моделирование при начальных нулевых условиях.

### Код моделирования системы с ПИ регулятором. Коэффициенты вручную. С подсчётом времени переходного процесса через производную.

|  |
| --- |
| 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 = 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];  % Параметры для стабилизации  transition\_time = NaN; % Время переходного процесса  derivative\_threshold = 0.01; % Пороговое значение 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 для поиска  Kp = 0.0006; % Значение параметар Kp  Ti = 1000; % Значение параметар Ti  error\_integral = 0; % Интеграл ошибки  u = ones(1, N); % Управляющее воздействие  transition\_time = NaN; % Время переходного процесса  % Моделирование системы  for i = 1:N-1  % Вычисляем ошибку (разница между уставкой и текущей температурой)  error = setpoint - x(1,i);  % Интегрируем ошибку  error\_integral = error\_integral + error \* h;  % ПИ-регулятор  u(i) = Kp \* (error + (1/Ti) \* error\_integral);  % Дискретное моделирование  x(:,i+1) = (A\*x(:,i) + B\*u(i)) \* h + x(:,i);  % Проверка установившегося состояния  if 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) - 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  % Вывод коэффициентов, времени переходного процесса и перерегулирования  fprintf('Лучшее время переходного процесса: %.2f секунд\n', transition\_time);  fprintf('Kp: %.4f\n', Kp);  fprintf('Ti: %.2f\n', Ti);  fprintf('Tmax: %.4f\n', Tmax);  fprintf('A2: %.4f\n', A2);  fprintf('Перерегулирование: %.4f%%\n', overshoot\_ratio);    % % Получение информации о переходном процессе через stepinfo  % info = stepinfo(x(1,:), t); % Здесь x(1,:) - это выходная величина, а t - время  % % Вывод информации о переходном процессе  % fprintf('Время переходного процесса (stepinfo): %.2f секунд\n', info.SettlingTime);  % Построение графика  figure;  plot(t, x(1,:), 'b', 'LineWidth', 2);  title('Температура рабочей камеры с наилучшими коэффициентами');  xlabel('Время (с)');  ylabel('T\_{р.к.} (°C)');  grid on;  end  end |

В данном коде реализован ПИ регулятор без подбора коэффициентов. Время переходного процесса считает со счётчиком, если следующие 1000 шагов производная равна меньше 1% (0.01) и значение температуры в пределах 2% (0.02). Так же есть проверка на уставку, если она 0, то берём абсолютное отклонение. Если перерегулирование равно меньше 4%, значит система стабилизировалась. Коэффициент k = 592,2.

### Готовый код моделирования с подбором коэффициентов через цикл for.

|  |
| --- |
| function PI\_regulator\_ustavka()  % 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];  % Параметры для стабилизации  transition\_time = NaN; % Время переходного процесса  derivative\_threshold = 0.01; % Пороговое значение 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 = 0;  best\_Ti = 0;  best\_overshoot = Inf; % Лучшее перерегулирование  best\_tmax = Inf;  % Параметры для поиска  Kp\_values = 0.001:0.001:0.1; % Диапазон значений Kp  Ti\_values = 1000:1:1500; % Диапазон значений Ti  % Начало цикла перебора коэффициентов  for Kp = Kp\_values  for Ti = Ti\_values  error\_integral = 0; % Интеграл ошибки  u = ones(1, N); % Управляющее воздействие  transition\_time = NaN; % Время переходного процесса  % Моделирование системы  for i = 1:N-1  % Вычисляем ошибку (разница между уставкой и текущей температурой)  error = setpoint - x(1,i);  % Интегрируем ошибку  error\_integral = error\_integral + error \* h;  % ПИ-регулятор  u(i) = Kp \* (error + (1/Ti) \* error\_integral);  % Дискретное моделирование  x(:,i+1) = (A\*x(:,i) + B\*u(i)) \* h + x(:,i);  % Проверка установившегося состояния  if 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) - 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 = Kp;  best\_Ti = Ti;  best\_A2 = A2;  best\_tmax = Tmax;  best\_overshoot = overshoot\_ratio;  best\_response = x(1,:); % Сохраняем ответ для построения графика  end  end  end  if best\_overshoot >= 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: %.4f\n', best\_Kp);  fprintf('Лучший Ti: %.2f\n', best\_Ti);  fprintf('Tmax: %.4f\n', best\_tmax);  fprintf('A2: %.4f\n', best\_A2);  fprintf('Перерегулирование: %.4f%%\n', best\_overshoot);  % % Получение информации о переходном процессе через stepinfo  % info = stepinfo(x(1,:), t); % Здесь x(1,:) - это выходная величина, а t - время  % % Вывод информации о переходном процессе  % fprintf('Время переходного процесса (stepinfo): %.2f секунд\n', info.SettlingTime);  % Построение графика с наилучшими коэффициентами  figure;  plot(t, best\_response, 'r', 'LineWidth', 2);  title('Температура рабочей камеры с наилучшими коэффициентами');  xlabel('Время (с)');  ylabel('T\_{р.к.} (°C)');  grid on;  end  end |

В данном коде реализован подбор параметров, который осуществляется через цикл for, выполняющий полный перебор значений параметров 𝐾𝑝 и 𝑇𝑖

% Параметры для поиска

Kp\_values = 0.001:0.001:0.1; % Диапазон значений Kp

Ti\_values = 1000:1:1500; % Диапазон значений Ti

Найденный параметры автоматическим способом при уставке 80:

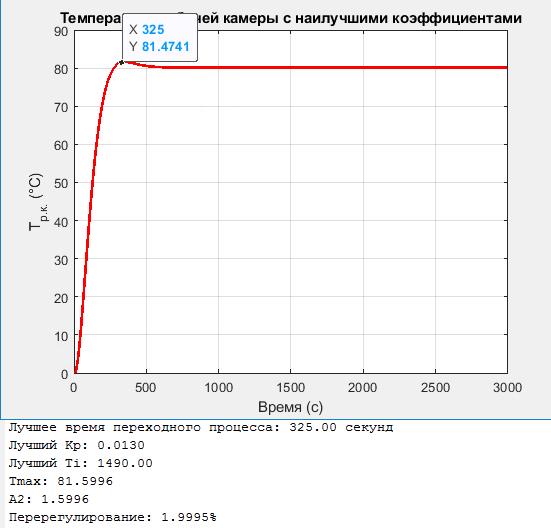
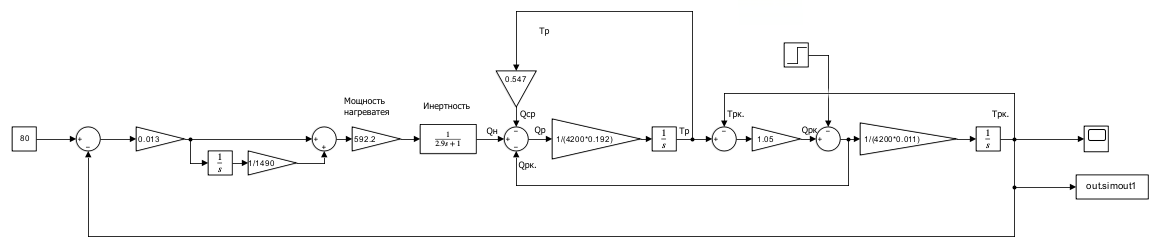
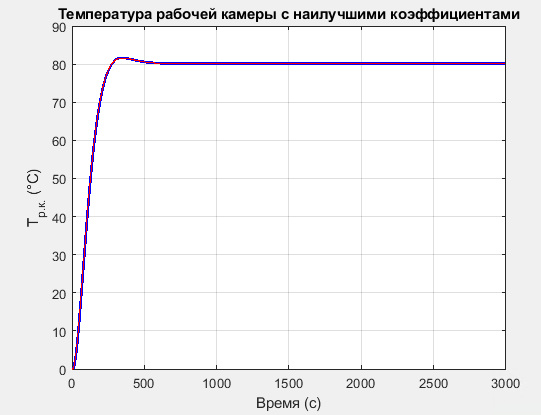


Схема с ПИ регулятором:





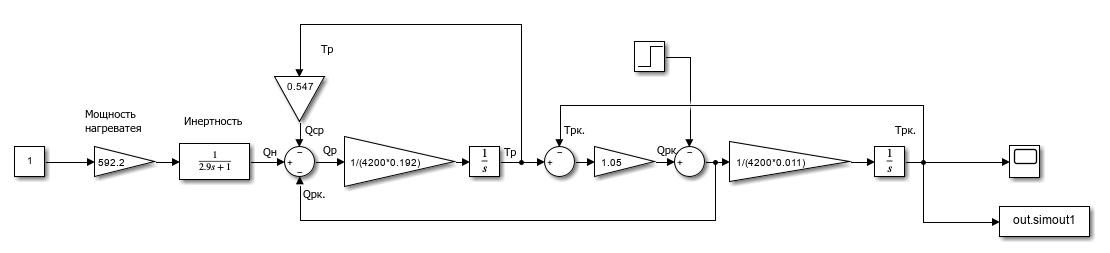
Сравнение графиков с Simulink и со скрипта. Графики сходятся, следовательно модель и код написаны правильно.

### По возмущению

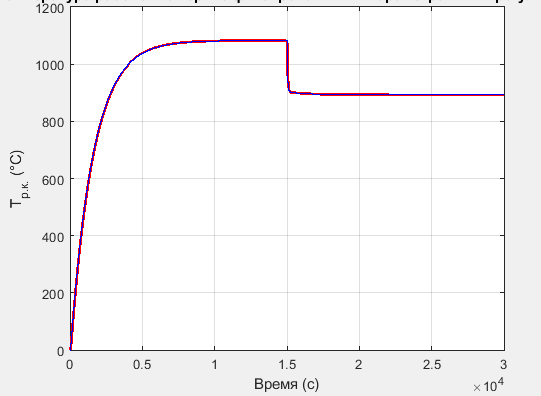
### Код моделирования системы с возмущением для сравнения со значениями simulink

|  |
| --- |
| function system() % Время моделирования  t\_total = 30000; % 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    % Начальные условия [T\_р.к., T\_p, Q\_H]  x = zeros(3, N); % Массив для хранения результатов  x(:,1) = [0; 0; 0];    % 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    % 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];  u = ones(1, N);  % Возмущение: изменение температуры окружающей среды  f = zeros(1, N);  f(15000:end) = 200;    for i = 1:N-1  % Дискретное моделирование с учётом возмущения  x(:,i+1) = (A\*x(:,i) + B\*u(i) + F\*f(i)) \* h + x(:,i);  end    % Построение графика с коэффициентами  %figure;  plot(t, x(1,:), 'r', 'LineWidth', 2);  title('Температура рабочей камеры с фиксированными параметрами ПИ-регулятора');  xlabel('Время (с)');  ylabel('T\_{р.к.} (°C)');  grid on; end |

Проверим с системой в Simulink:





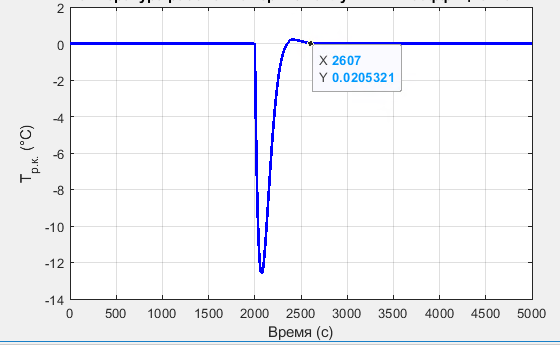


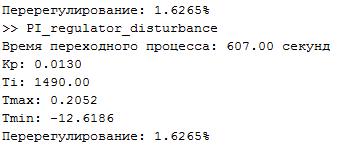
Графики совпадают, следовательно, система построена правильно.

Взято из ВКР Бутузова: Коэффициенты теплопередачи 𝑘1 = 0,547 и 𝑘2 = 1,05. Мощность k = 1973. Было взято 30% коэффициент мощности k = 592.2. В данной системе моделирование при начальных нулевых условиях.

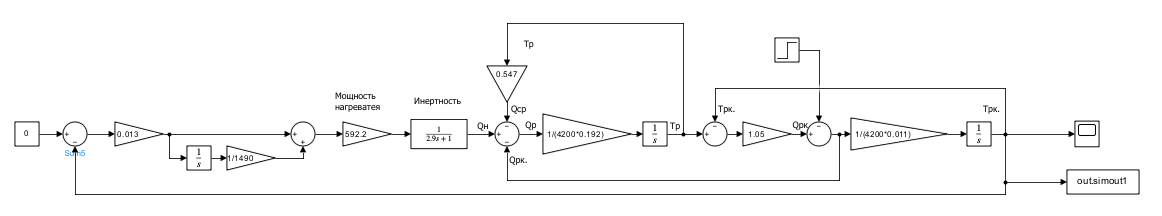
### Код моделирования системы с возмущением, вручную и подсчётом перерегулирования и времени переходного процесса

|  |
| --- |
| function PI\_regulator\_disturbance()  % 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 = 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];  % Возмущение: изменение температуры окружающей среды  f = zeros(1, N);  start\_disturbance = 2000;  f(start\_disturbance:end) = 20;  % Параметры для стабилизации  transition\_time = NaN; % Время переходного процесса  derivative\_threshold = 0.01; % Пороговое значение 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  Kp = 0.013;  Ti = 1490;  error\_integral = 0; % Интеграл ошибки  u = ones(1, N); % Управляющее воздействие  % Моделирование системы  for i = 1:N-1  % Вычисляем ошибку (разница между уставкой и текущей температурой)  error = setpoint - x(1,i);  % Интегрируем ошибку  error\_integral = error\_integral + error \* h;  % ПИ-регулятор  u(i) = Kp \* (error + (1/Ti) \* error\_integral);  % Дискретное моделирование  x(:,i+1) = (A\*x(:,i) + B\*u(i) + F\*f(i)) \* h + x(:,i);  % Проверка установившегося состояния после возмущения  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) - 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, 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('Время переходного процесса: %.2f секунд\n', transition\_time);  fprintf('Kp: %.4f\n', Kp);  fprintf('Ti: %.2f\n', Ti);  fprintf('Tmax: %.4f\n', Tmax);  fprintf('Tmin: %.4f\n', Tmin);  fprintf('Перерегулирование: %.4f%%\n', overshoot\_ratio);    % Построение графика с наилучшими коэффициентами  figure;  plot(t, x(1,:), 'b', 'LineWidth', 2);  title('Температура рабочей камеры с наилучшими коэффициентами');  xlabel('Время (с)');  ylabel('T\_{р.к.} (°C)');  grid on;  end  end |

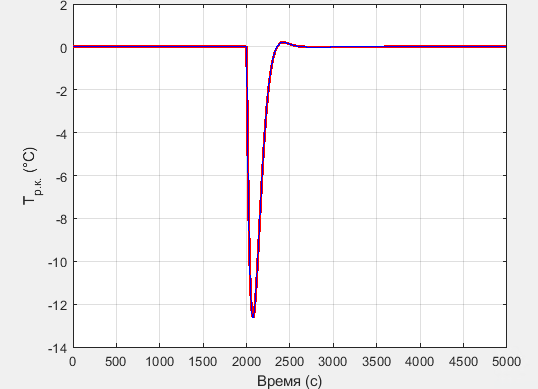


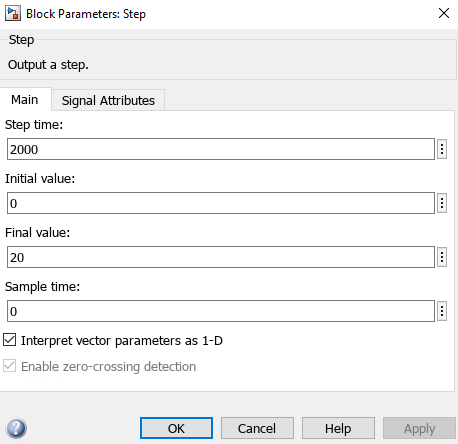


Код написан с возмущением и ПИ регулятором без перебора коэффициентов. Возмущение с 2000с на 20 градусов. Время переходного процесса считает со счётчиком, если следующие 1000 шагов производная равна меньше 1% (0.01) и значение температуры в пределах 2% (0.02). Так же есть проверка на уставку, если она 0, то берём абсолютное отклонение. Если перерегулирование равно меньше 4%, значит система стабилизировалась. Коэффициент k = 592,2.



Сравнение графиков с моделью:

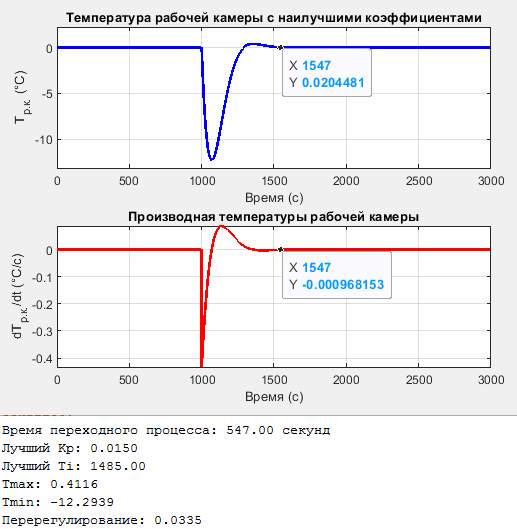




### Готовый код моделирования системы с возмущением и перебором коэффициентов

|  |
| --- |
| function PI\_regulator\_disturbance\_auto()  % 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];  % Возмущение: изменение температуры окружающей среды  f = zeros(1, N);  start\_disturbance = 1000;  f(start\_disturbance:end) = 20;  % Параметры для стабилизации  derivative\_threshold = 0.01; % Пороговое значение 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 перебора коэффициентов ПИ-регулятора  Kp\_values = 0.001:0.001:0.12; % Диапазон значений Kp  Ti\_values = 500:5:1500; % Диапазон значений Ti  % Переменные для хранения наилучших результатов  best\_time = Inf;  best\_Kp = 0;  best\_Ti = 0;  best\_overshoot = Inf;  % Цикл перебора значений Kp и Ti  for Kp = Kp\_values  for Ti = Ti\_values  error\_integral = 0; % Интеграл ошибки  u = ones(1, N); % Управляющее воздействие  transition\_time = NaN; % Время переходного процесса  % Моделирование системы  for i = 1:N-1  % Вычисляем ошибку (разница между уставкой и текущей температурой)  error = setpoint - x(1,i);  % Интегрируем ошибку  error\_integral = error\_integral + error \* h;  % ПИ-регулятор  u(i) = Kp \* (error + (1/Ti) \* error\_integral);  % Дискретное моделирование  x(:,i+1) = (A\*x(:,i) + B\*u(i) + F\*f(i)) \* h + x(:,i);  % Проверка установившегося состояния после возмущения  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) - 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, start\_disturbance:end));  Tmin = min(x(1, start\_disturbance:end));  A2 = Tmax - setpoint;  A1 = Tmin - setpoint;  overshoot\_ratio = abs(A2 / A1);  % Обновление наилучших параметров, если найдено меньшее время переходного процесса  if ~isnan(transition\_time) && transition\_time < best\_time && overshoot\_ratio <= 0.04  best\_time = transition\_time;  best\_Kp = Kp;  best\_Ti = Ti;  best\_overshoot = overshoot\_ratio;  best\_Tmax = Tmax;  best\_Tmin = Tmin;  best\_response = x(1,:); % Сохраняем ответ для построения графика  end  end  end  if best\_overshoot >= 4  fprintf('Система не сбалансировалась\n');  else  % Вывод наилучших коэффициентов, времени переходного процесса и перерегулирования  fprintf('Время переходного процесса: %.2f секунд\n', best\_time);  fprintf('Лучший Kp: %.4f\n', best\_Kp);  fprintf('Лучший Ti: %.2f\n', best\_Ti);  fprintf('Tmax: %.4f\n', best\_Tmax);  fprintf('Tmin: %.4f\n', best\_Tmin);  fprintf('Перерегулирование: %.4f\n', best\_overshoot);    % Построение графика температуры  figure;  subplot(2, 1, 1); % Первый график в позиции 1  plot(t, best\_response, 'b', 'LineWidth', 2);  title('Температура рабочей камеры с наилучшими коэффициентами');  xlabel('Время (с)');  ylabel('T\_{р.к.} (°C)');  grid on;  % Расчёт производной температуры  temperature\_derivative = diff(best\_response) / h;  % Создание временной оси для производной (короче на 1 шаг)  t\_derivative = t(1:end-1);  % Построение графика производной температуры  subplot(2, 1, 2); % Второй график в позиции 2  plot(t\_derivative, temperature\_derivative, 'r', 'LineWidth', 2);  title('Производная температуры рабочей камеры');  xlabel('Время (с)');  ylabel('dT\_{р.к.}/dt (°C/с)');  grid on;  end  end |

При уставке 0, возмущении 20:



Сравнение графиков с Simulink:

