

# **3 Специальная часть**

### 3.1 ПИД РЕГУЛЯТОР, ЕГО ЭЛЕМЕНТЫ И ИХ НАЗНАЧЕНИЕ

Пропорционально-интегрально-дифференцирующий (ПИД) регулятор – устройство в управляющем контуре с обратной связью. Используется в системах автоматического управления для формирования управляющего сигнала с целью получения необходимых точности и качества переходного процесса. ПИД-регулятор формирует управляющий сигнал, являющийся суммой трёх слагаемых, первое из которых пропорционально разности входного сигнала и сигнала обратной связи (сигнал рассогласования), второе – интеграл сигнала рассогласования, третье – производная сигнала рассогласования.

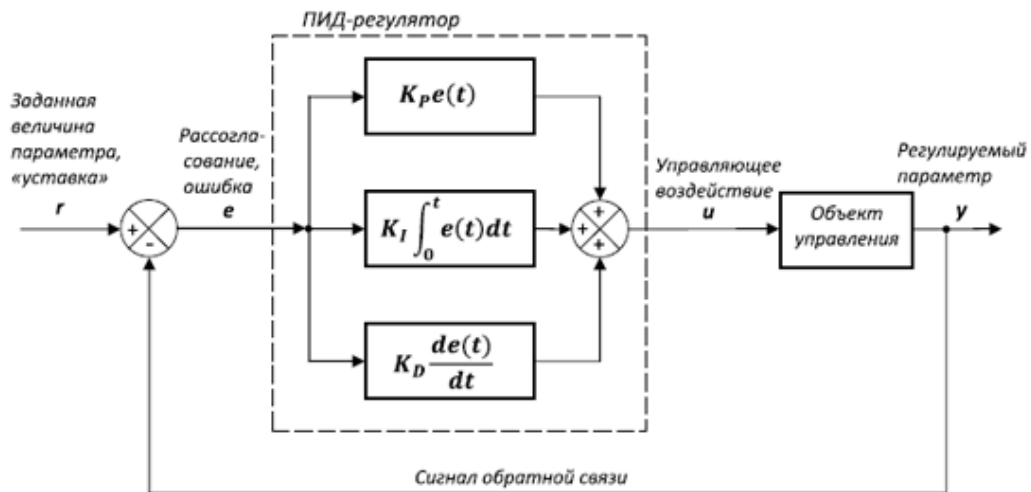


Рисунок 4.1 — Принципиальная схема ПИД регулятора

#### 3.1.1 Пропорциональная составляющая

Пропорциональная составляющая вырабатывает выходной сигнал, противодействующий отклонению регулируемой величины от заданного значения, наблюдаемому в данный момент времени. Он тем больше, чем больше это отклонение. Если входной сигнал равен заданному значению, то выходной равен нулю.

$$u(t) = K_p e(t),$$

где  $K_p$  - коэффициент пропорциональной составляющей  
 $e(t)$  - величина рассогласования, ошибка регулирования

Однако при использовании только пропорционального регулятора значение регулируемой величины никогда не стабилизируется на заданном значении. Существует так называемая статическая ошибка, которая равна такому отклонению регулируемой величины, которое обеспечивает выходной сигнал, стабилизирующий выходную величину именно на этом значении. Например, в регуляторе температуры выходной сигнал (мощность нагревателя) постепенно уменьшается при приближении температуры к заданной, и система стабилизируется при мощности, равной тепловым потерям. Температура не может достичь заданного значения, так как в этом случае мощность нагревателя станет равна нулю, и он начнёт остывать.

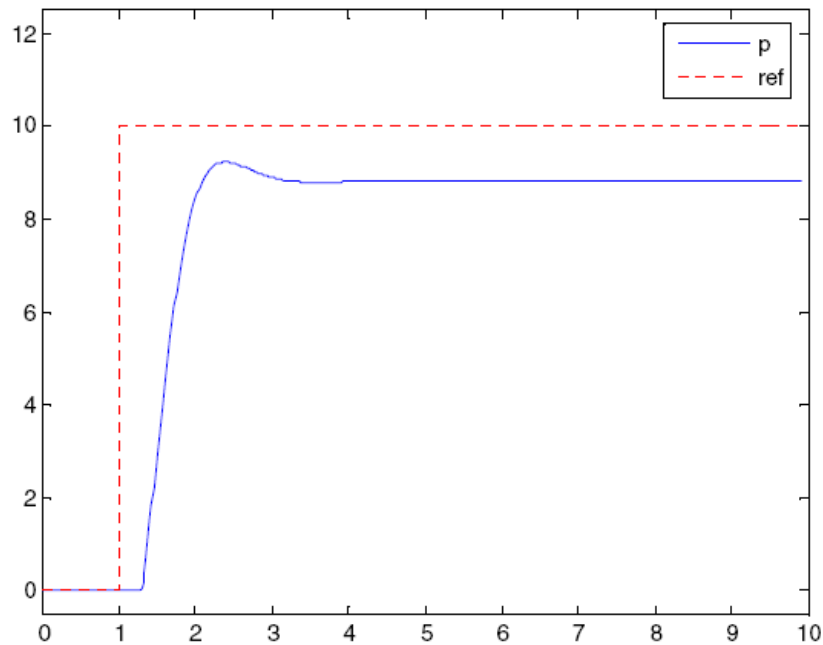


Рисунок 4.2 — Отклик П регулятора на единичный сигнал

Чем больше коэффициент пропорциональности между входным и выходным сигналом (коэффициент усиления), тем меньше статическая ошибка, однако при слишком большом коэффициенте усиления при наличии задержек (запаздывания) в системе могут начаться автоколебания, а при дальнейшем увеличении коэффициента система может потерять устойчивость.

### 3.1.2 Интегрирующая составляющая

Интегрирующая составляющая пропорциональна интегралу по времени от отклонения регулируемой величины. Её используют для устранения статической ошибки. Она позволяет регулятору со временем учесть статическую ошибку.

При пропорциональном регулировании выходной сигнал регулятора изменяется пропорционально изменению входного сигнала. Когда изменения на вводе прекращаются, вывод регулятора также перестает изменяться, это приводит к появлению неустранимой пропорциональным регулятором статической ошибки. Когда в процесс регулирования к пропорциональной составляющей добавляется интегральная, регулятор продолжает корректировать вывод до полного возврата регулируемой переменной процесса к заданной. Интегральное регулирование добавляет корректирующее действие к пропорциональному действию. Тем не менее, интегрирующая составляющая также может приводить к автоколебаниям при неправильном выборе её коэффициента.

$$u(t) = K_p e(t) + K_i \int_0^t e(t) dt ,$$

где  $K_i$  - коэффициент интегральной составляющей.

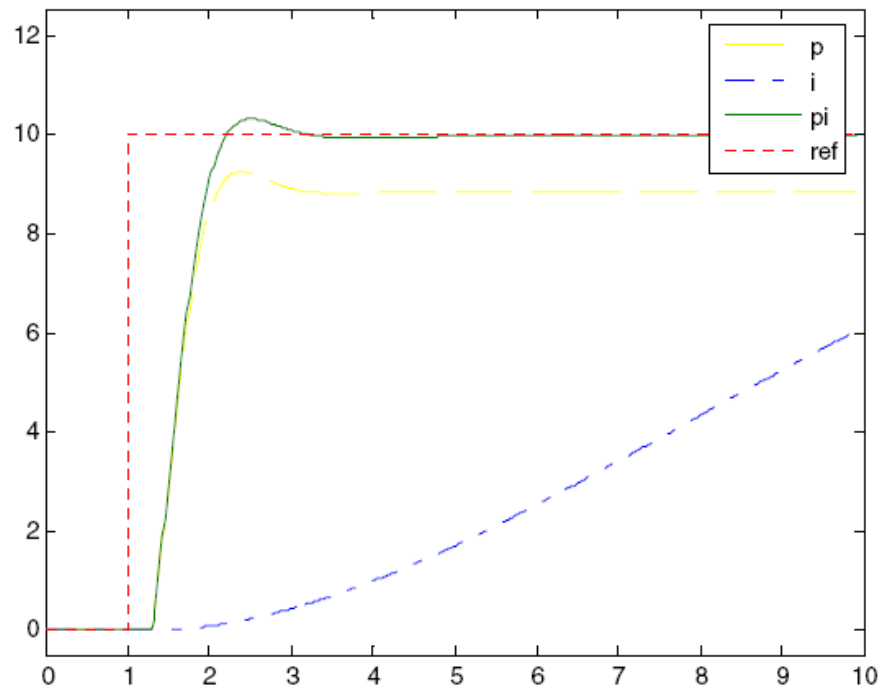


Рисунок 4.3 — Отклик И и ПИ-регулятора на единичный сигнал

### 3.1.3 Дифференцирующая составляющая

Дифференцирующая составляющая пропорциональна темпу изменения отклонения регулируемой величины и предназначена для противодействия отклонениям от целевого значения, которые прогнозируются в будущем. Отклонения могут быть вызваны внешними возмущениями или запаздыванием воздействия регулятора на систему.

Как и в случае интегрального регулирования, дифференциальное регулирование не существует непосредственно само по себе: оно всегда объединяется с пропорциональным регулированием. При дифференциальной составляющей, добавленной к пропорциональному регулятору, когда происходит изменение регулируемой переменной, регулятор измеряет скорость изменения и производит мгновенное наращивание пропорционального выходного сигнала.

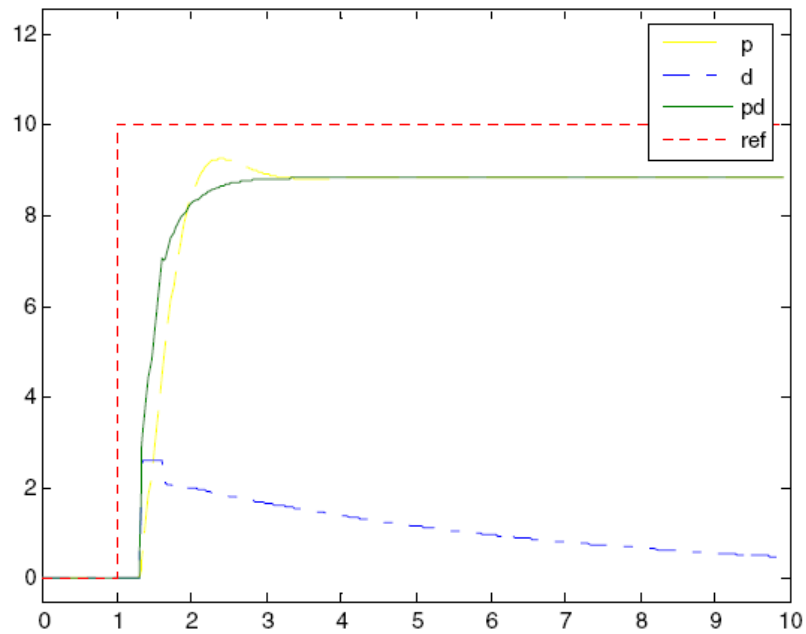


Рисунок 4.4 — Отклик Д и ПД-регулятора на единичный сигнал

Эта составляющая противодействует изменению входного сигнала регулятора (регулируемой переменной процесса) и пытается останавливать изменения, как только они обнаружены. Когда изменения входного сигнала прекращаются, корректирующее воздействие, производимое дифференциальным регулятором, исчезает, и остается только часть выходного сигнала, выработанная пропорциональным регулятором.

$$u(t) = K_p e(t) + K_d \frac{de}{dt},$$

где  $K_d$  - коэффициент дифференциальной составляющей.

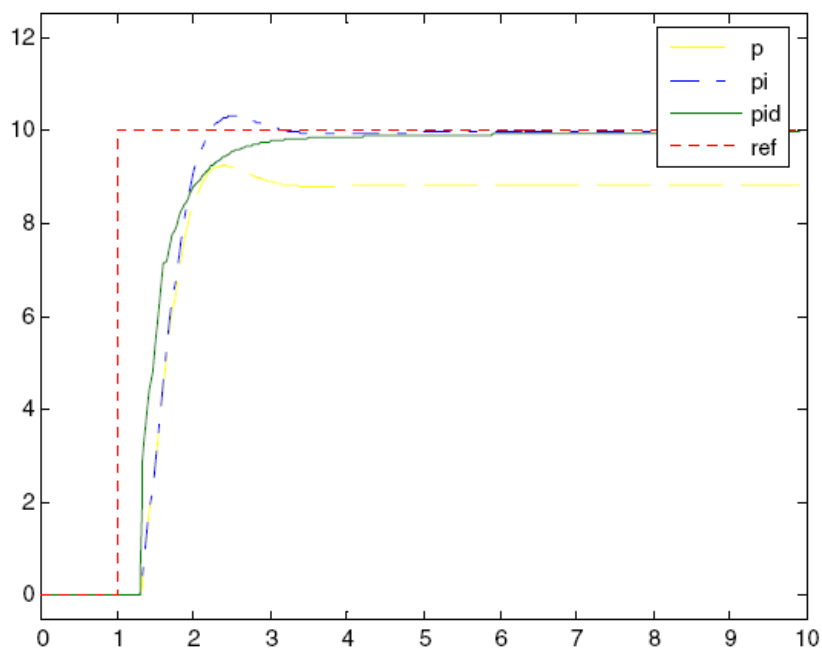


Рисунок 4.5 — Отклик П, ПИ и ПИД-регулятора на единичный сигнал

Собрав все составляющие вместе, мы получаем формулу непрерывного ПИД регулятора:

$$u(t) = K_p e(t) + K_i \int_0^t e(t) dt + K_d \frac{de}{dt},$$

далее преобразовываем в операторную форму по Лапласу:

$$W(s) = K_p + \frac{K_i}{s} + K_d s = \frac{K_d s^2 + K_p s + K_i}{s}$$

### 3.1.4 Дискретная форма регулятора

Непрерывные переменные удобно использовать для анализа и синтеза ПИД регуляторов. Для технического воплощения необходимо перейти к дискретной форме уравнений, поскольку основой всех регуляторов является микроконтроллер, контроллер или компьютер, который оперирует с переменными, полученными из аналоговых сигналов после их дискретизации по времени и дискретизации по уровню. Вследствие конечного времени вычисления управляющего воздействия в микроконтроллере и задержки аналого-цифрового преобразования между моментом поступления аналогового сигнала на вход регулятора и появлением управляющего воздействия на его выходе появляется нежелательная задержка, которая увеличивает общую задержку в контуре регулирования и снижает запас устойчивости. Основным эффектом, который появляется при дискретизации является появление алиасных частот в спектре дискретного сигнала в случае, когда частота дискретизации недостаточно высока.

Аналогичный эффект возникает при киносъёмке вращающегося колеса автомобиля. Частота алиасного сигнала равно разности между частотой помехи и частотой квантования. При этом высокочастотный сигнал помехи смещается в низкочастотную область, где накладывается на полезный сигнал и создаёт большие проблемы, поскольку отфильтровать его на этой стадии невозможно. Для устранения алиасного эффекта перед входом аналого-цифрового преобразователя необходимо установить аналоговый фильтр, который бы ослаблял помеху, по крайней мере, на порядок на частоте, равной половине частоты дискретизации. Обычно используют фильтр Баттерворта второго или более высокого порядка. Вторым вариантом решения проблемы является увеличение частоты дискретизации так, чтобы она, по крайней мере, в 2 раза была выше максимальной частоты спектра помехи.

Это позволяет применить после дискретизации цифровой фильтр нижних частот. При такой частоте дискретизации полученный цифровой сигнал с точки зрения количества информации полностью эквивалентен аналоговому, и все свойства аналогового регулятора можно распространить на цифровой.

### 3.1.5 Погрешность дифференцирования и шум

Проблема численного дифференцирования является достаточно старой и общей как в цифровых, так и в аналоговых регуляторах. Суть её заключается в том, что производная вычисляется обычно как разность двух близких по величине переменных, поэтому относительная погрешность производной всегда оказывается больше, чем относительная погрешность численного представления дифференцируемой переменной. В частности, если на вход дифференциатора поступает синусоидальный сигнал  $A\sin(\omega t)$ , то на выходе получим  $A\omega \cdot \cos(\omega t)$ , то есть с ростом частоты  $\omega$  увеличивается амплитуда сигнала на выходе дифференциатора. Иначе говоря, дифференциатор усиливает высокочастотные помехи, короткие выбросы и шум. Если помехи, усиленные дифференциатором, лежат за границей диапазона рабочих частот ПИД регулятора, то их можно ослабить с помощью фильтра верхних частот.

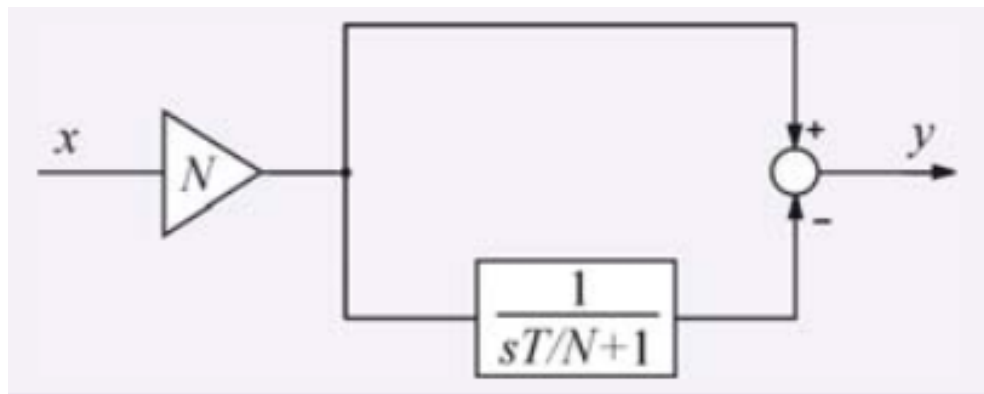


Рисунок 4.6 — Структурная схема дифференциатора с фильтром

$$y = Nx \left( 1 - \frac{1}{sK_d / N + 1} \right) = \left( \frac{sK_d}{sK_d / N + 1} \right) x,$$

то есть передаточная функция полученного дифференциатора  $K_d s$  может быть представлена в виде произведения передаточной функции идеального дифференциатора и передаточной функции фильтра первого порядка:

$$W(s) = K_d s \left( \frac{1}{K_d s / N + 1} \right)$$

где коэффициент  $N$  задаёт граничную частоту фильтра и обычно выбирается равным 2...20,  $s$  — оператор Лапласа.

$$W(s) = K_p + \frac{K_i}{s} + K_d s \left( \frac{1}{K_d s / N + 1} \right)$$

Кроме шумов дифференцирования, на характеристики ПИД регулятора влияют шумы измерений. Через цепь обратной связи эти шумы поступают на вход системы и затем проявляются как дисперсия управляющей переменной  $u$ . Высокочастотные шумы вредны тем, что вызывают ускоренный износ трубопроводов и электродвигателей. Поскольку объект управления обычно

является низкочастотным фильтром, шумы измерений редко проникают по контуру регулирования на выход системы. Однако они увеличивают погрешность измерений и снижают точность регулирования.

### 3.1.6 Переход к конечно-разностным уравнениям.

Переход к дискретным переменным в уравнениях аналогового регулятора выполняется путём замены операторов производных и интегралов их дискретными аналогами. Существует множество способов аппроксимации производных и интегралов их дискретными аналогами, которые изложены в курсах численных методов решения дифференциальных уравнений. В ПИД регуляторах наиболее распространёнными являются простейшие виды аппроксимации производной конечной разностью и интеграла – конечной суммой. В данной работе будет использоваться билинейное преобразование (метод трапеций), используемое для преобразования передаточной функции линейной стационарной системы непрерывной формы в передаточную функцию линейной системы в дискретной форме. Оно отображает точки  $j\omega$ -оси,  $\text{Re}[s] = 0$ , на  $s$ -плоскости в окружность единичного радиуса,  $|z| = 1$ , на  $z$ -плоскости.

Билинейное преобразование представляет собой функцию, аппроксимирующую натуральный логарифм, который является точным отображением  $z$ -плоскости на  $s$ -плоскость. При применении преобразования Лапласа над дискретным сигналом (представляющего последовательность отсчётов), результатом является  $z$ -преобразование с точностью до замены переменных:

$$z = \frac{e^{\frac{sT}{2}}}{e^{-\frac{sT}{2}}} \approx \frac{1 + \frac{sT}{2}}{1 - \frac{sT}{2}},$$

где  $T$  - период квантования

Аппроксимация, приведённая выше и является билинейным преобразованием. Обратное преобразование из  $s$ -плоскости в  $z$ -плоскость и его билинейная аппроксимация записываются следующим образом:

$$s = \frac{1}{T} \ln(z) = \frac{2}{T} \left[ \frac{z-1}{z+1} + \frac{1}{3} \left( \frac{z-1}{z+1} \right)^3 + \frac{1}{5} \left( \frac{z-1}{z+1} \right)^5 + \frac{1}{7} \left( \frac{z-1}{z+1} \right)^7 + \dots \right] \approx \frac{2}{T} \left( \frac{z-1}{z+1} \right)$$

Билинейное преобразование использует это соотношения для замены передаточной функции непрерывной системы на её дискретный аналог:

$$s \leftarrow \frac{2}{T} \left( \frac{z-1}{z+1} \right)$$

Далее преобразовываем передаточную функцию ПИД регулятора:



$$W(s) = K_p + \frac{K_i}{s} + sT \left( \frac{1}{sT / N + 1} \right) = K_p + K_i \frac{T}{2} \left( \frac{z+1}{z-1} \right) + K_d \frac{N}{1 + N \frac{T}{2} \left( \frac{z+1}{z-1} \right)},$$

Эта формула будет использована при расчете ПИД регулятора в математическом пакете Matlab Simulink.

Величина периода квантования  $T$  выбирается как можно меньше, это улучшает качество регулирования. Для обеспечения хорошего качества регулирования он не должен быть больше чем  $1/15 \dots 1/6$  от времени установления переходной характеристики объекта по уровню 0,95 или  $1/4 \dots 1/6$  от величины транспортной задержки. Однако при увеличении частоты квантования более чем в 2 раза по сравнению с верхней частотой спектра возмущающих сигналов дальнейшего улучшения качества регулирования не происходит. Если на входе регулятора нет антиалиасного фильтра, то частоту квантования выбирают в 2 раза выше верхней граничной частоты спектра помехи, чтобы использовать цифровую фильтрацию. Необходимо учитывать также, что исполнительное устройство должно успеть отработать за  $T$ . Если контроллер используется не только для регулирования, но и для аварийной сигнализации, то период квантования не может быть меньше, чем допустимая задержка срабатывания сигнала аварии. При малом периоде квантования увеличивается погрешность вычисления производной.

### 3.2 ПОСТРОЕНИЕ СХЕМЫ ПИД РЕГУЛЯТОРА

Передаточная функция насос-регулятора состоит из передаточных функций дозирующей иглы  $W_1(s)$  и клапана постоянного перепада давления  $W_2(s)$  соединенных последовательно:

$$W(s) = W_1(s) \cdot W_2(s),$$

где  $W_1(s) = \Delta\lambda_{ou} / \Delta I = K_1 / (s + 1/\tau_1)$ ;

$$W_2(s) = \Delta G_m / \Delta\lambda_{ou} = K_2 / (\tau_1 s + 1);$$

$\Delta G_m$  - изменение дозируемого расхода топлива, кг/ч;

$\Delta I$  - изменение тока управления, мА;

$\Delta\lambda_{ди}$  - изменение перемещения датчика положения дозирующей иглы, см;

$K_1 = (0.032 \pm 0.012)$  – коэффициент усиления по перемещению дозирующей иглы, см/(мА·с);

$\tau_1 = (15.0 \pm 3.0)$  – постоянная времени дозирующей иглы, с;

$K_2 = 1200^{+250}_{-150}$  - коэффициент усиления по расходу топлива, кг/(ч·см);

$\tau_2 \leq 0.1$  – постоянная времени клапана постоянного перепада, с.

#### 3.2.1 Порядок построения схемы

1. В основном окне программы Matlab запускаем вкладку Simulink

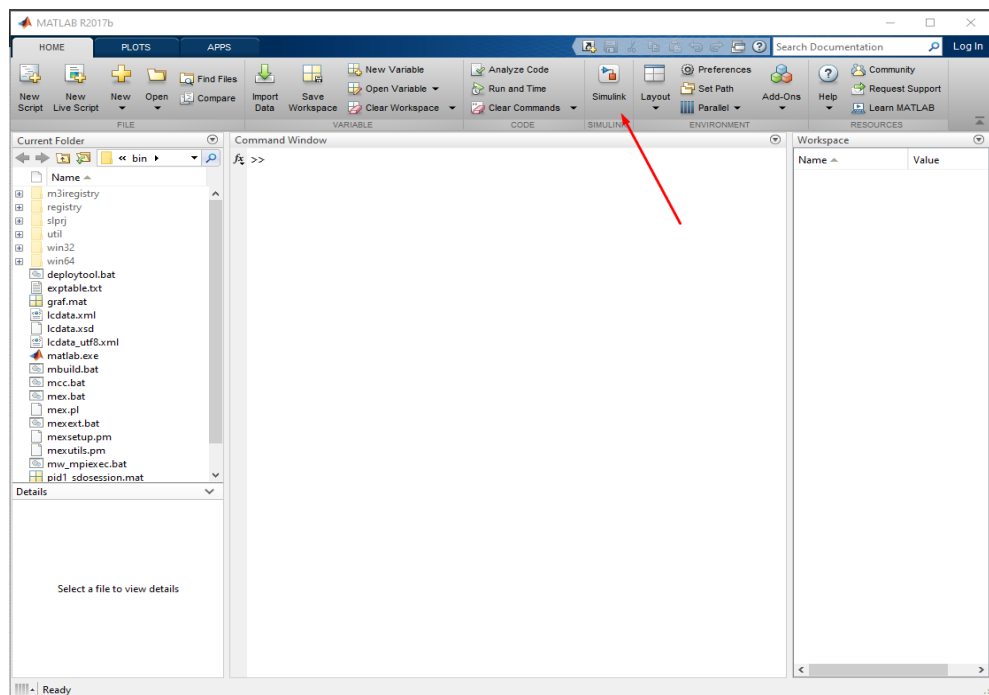


Рисунок 4.7 — Основное окно Matlab

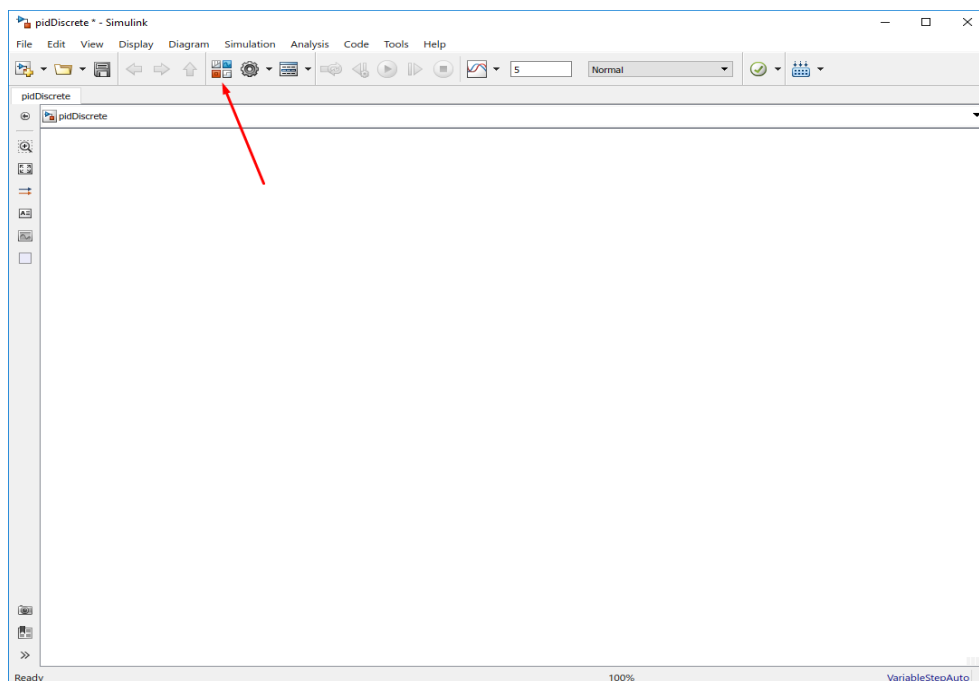


Рисунок 4.15 — Основное окно Simulink

2. В открывшемся окне запускаем библиотеку компонентов Simulink, из которой будут взяты составные элементы схемы контура регулирования. Выбираем источник единичного сигнала Step и перетаскиваем его в окно ранее открытого Simulink.

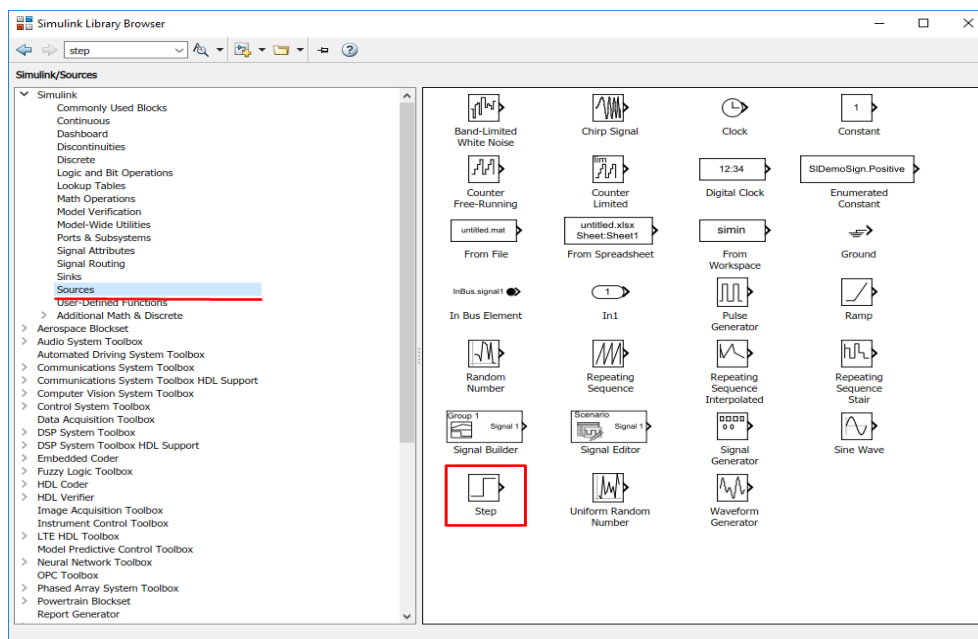


Рисунок 4.7 — Библиотека компонентов Simulink

3. Двойной щелчок мышкой в окне Simulink на Step компоненте. После чего откроется окно настройки. В котором вводим шаг времени Step time = 0.1, начальное значение Initial value = 0 и конечное значение Final value = 1.
4. Добавляем компонент Sum который будет суммировать заданное значение с измеренным значением.

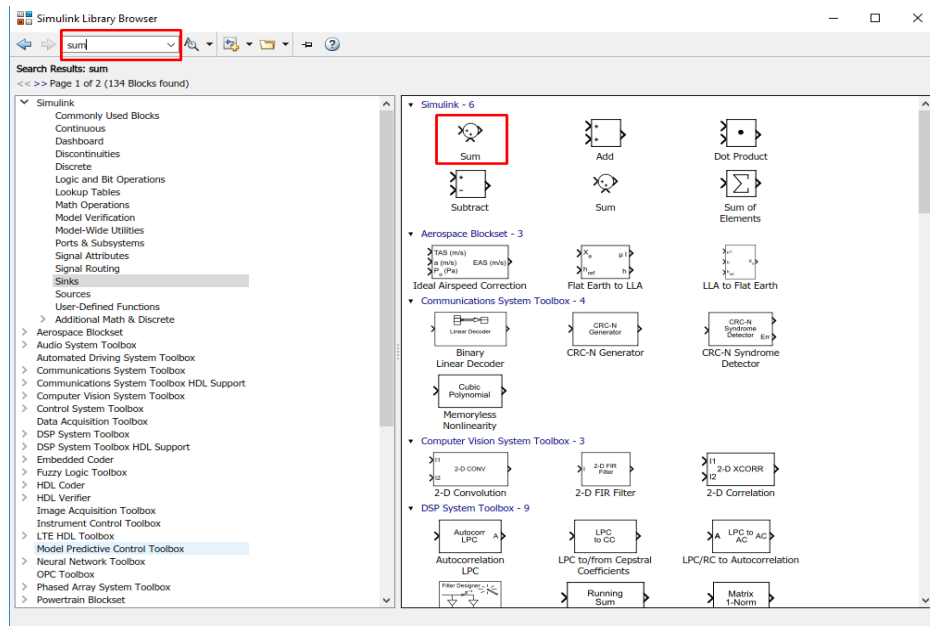


Рисунок 4.8 — Библиотека компонентов Simulink

В настройках компонента задаем знак сигналов, в результате чего мы будем получать разницу между заданным и измеренным значением.

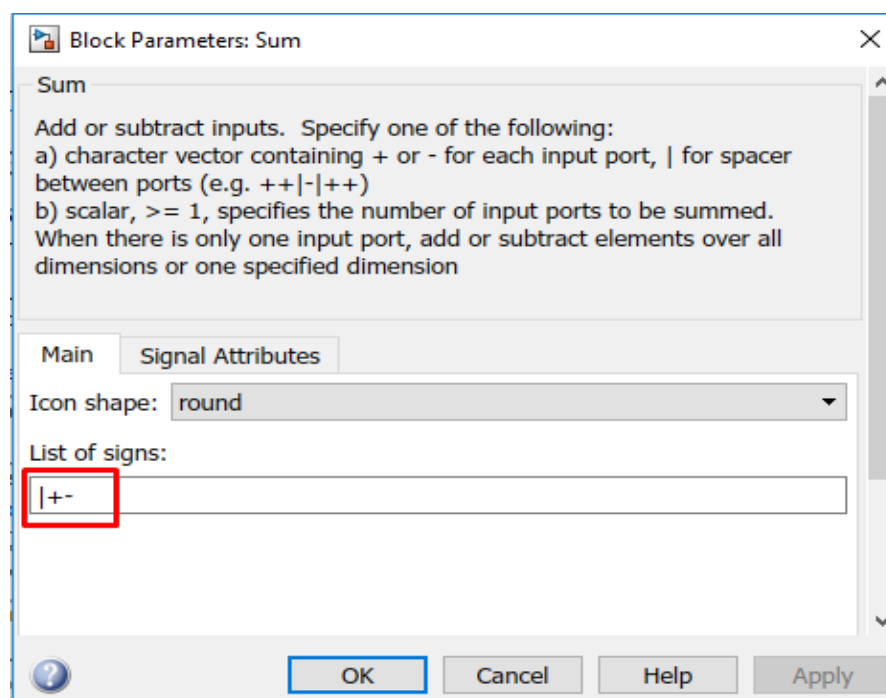


Рисунок 4.9 — Настройки компонента SUM

- Добавляем ПИД регулятор на схему. Настройки компонента будут рассмотрены позднее.

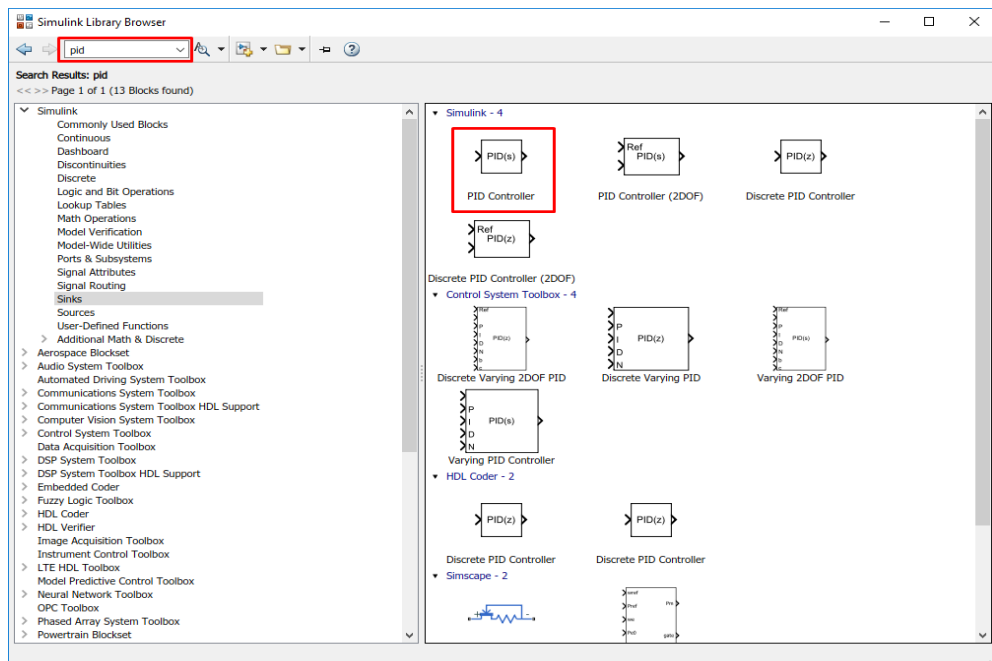


Рисунок 4.10 — Библиотека компонентов Simulink

6. Добавляем компонент Transfer Fcn в котором будет заданная передаточная функция дозирующей иглы.

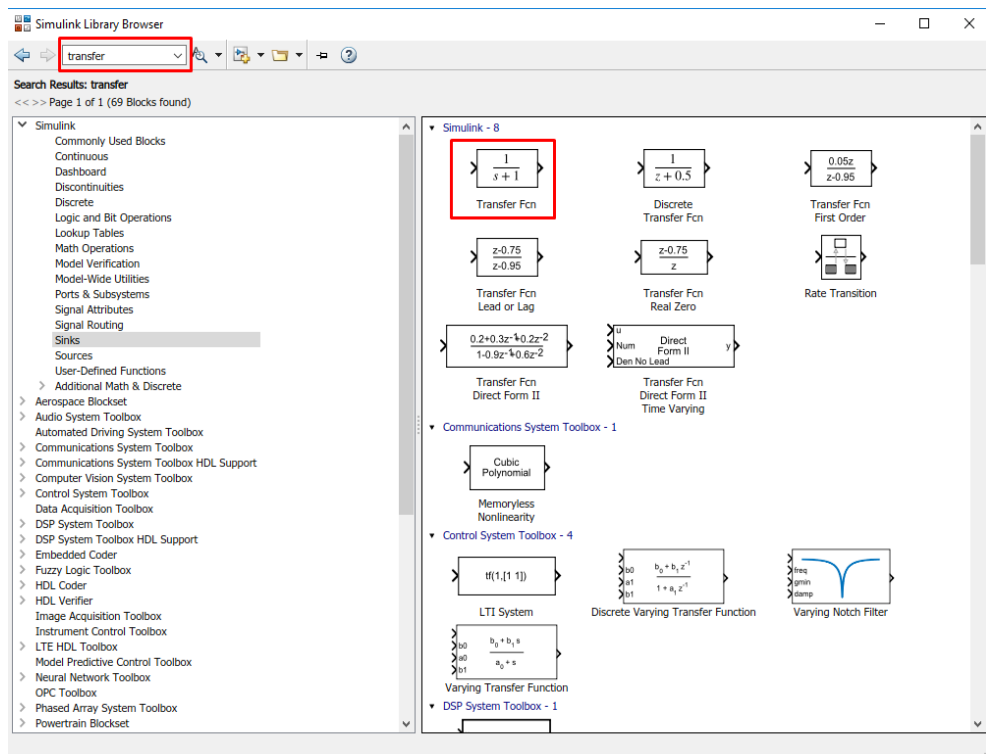


Рисунок 4.11 — Библиотека компонентов Simulink

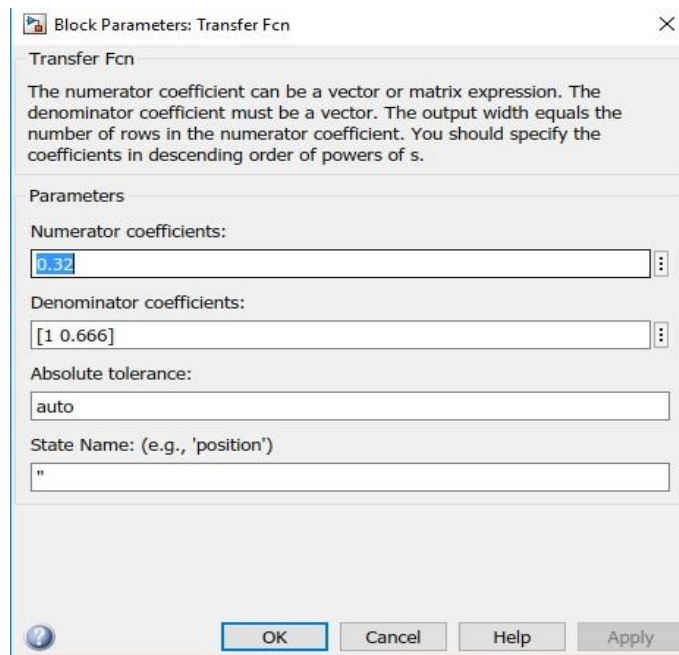


Рисунок 4.12 — Настройка компонента Transfer Fcn

В настройках задаем параметры функции. Numeration coefficients будет соответствовать числителю, а Denomination coefficients знаменателю. Выражение в знаменателе [1 (пробел соответствует знаку +) 0.666] соответствует выражению  $s + 0.666$ .

7. Повторяем то же самое для клапана постоянного перепада давления и задаем его настройки.

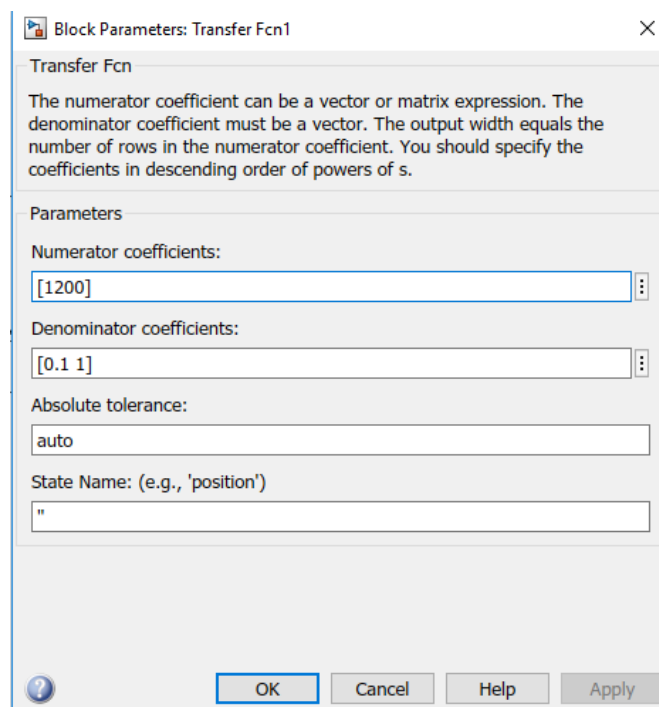


Рисунок 4.12 — Настройка компонента Transfer Fcn1

8. Добавляем на схему Мux компонент который объединяет входные сигналы одного и того же типа данных и числового типа в виртуальный вектор.

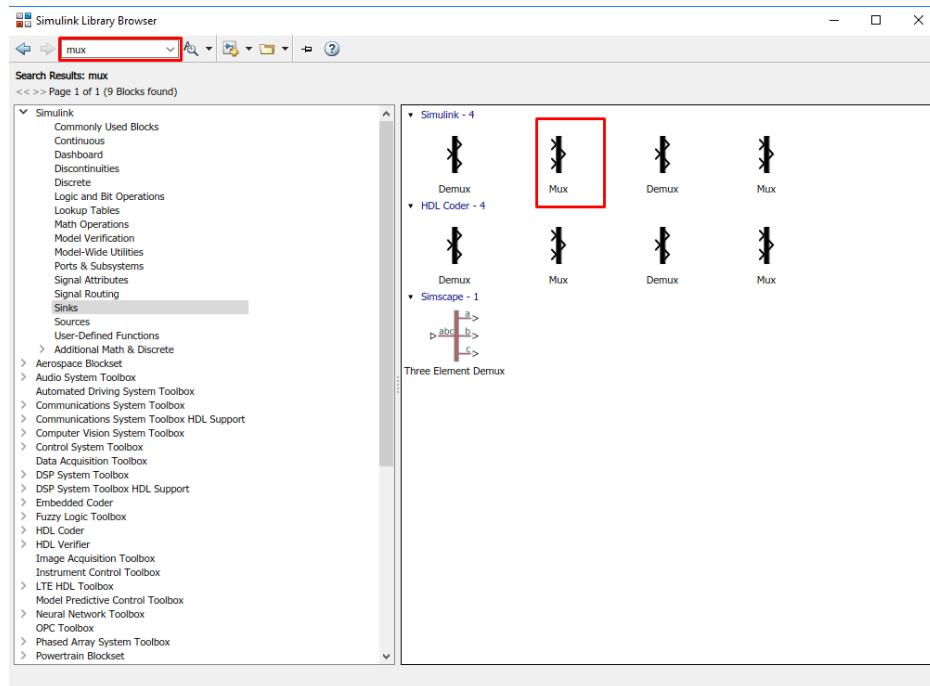


Рисунок 4.13 — Библиотека компонентов Simulink

Настройки для Mux компонента. Вводим количество сигналов равное 2.

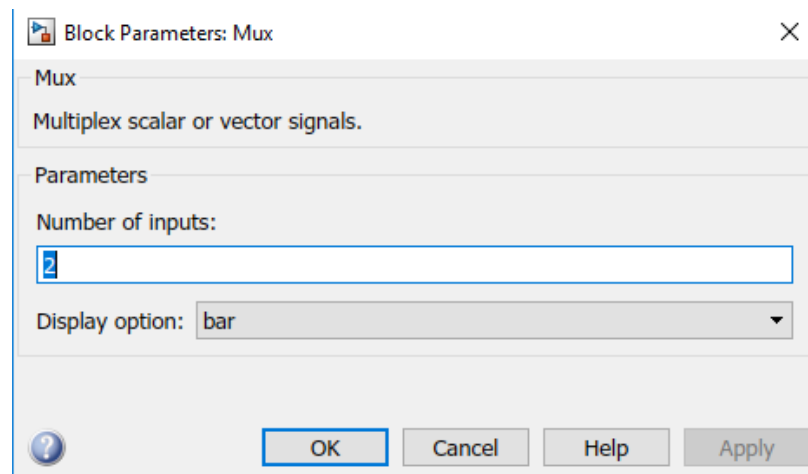


Рисунок 4.14 — Настройка компонента Mux

9. Добавляем на схему Scope компонент который требуется для построения графика.

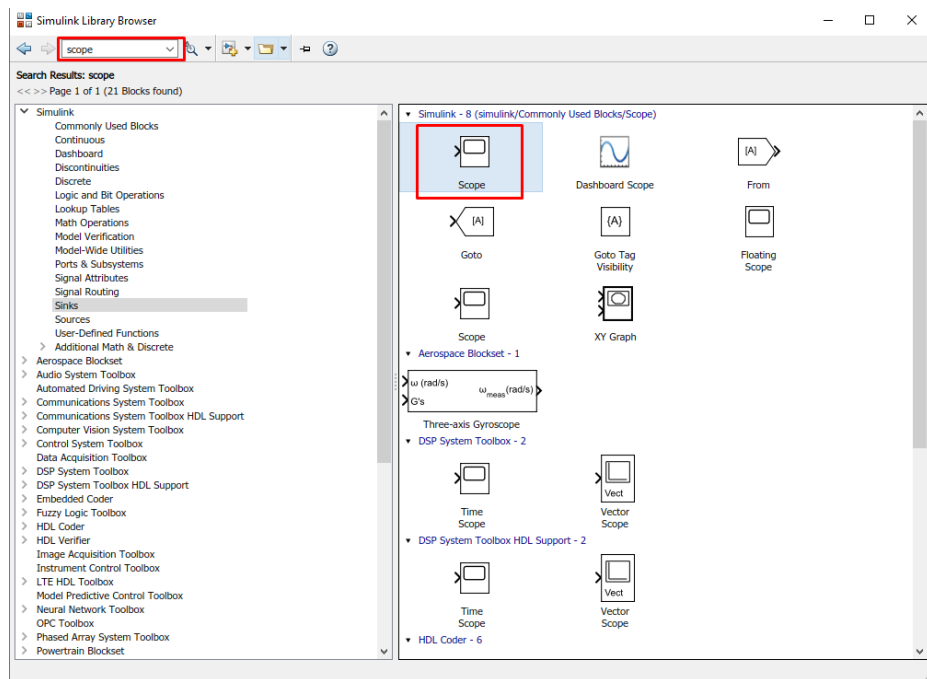


Рисунок 4.15 — Библиотека компонентов Simulink

10. Добавляем компонент Check Step Response Characteristics. В этом компоненте будет проводится оптимизация передаточной функции ПИД регулятора.

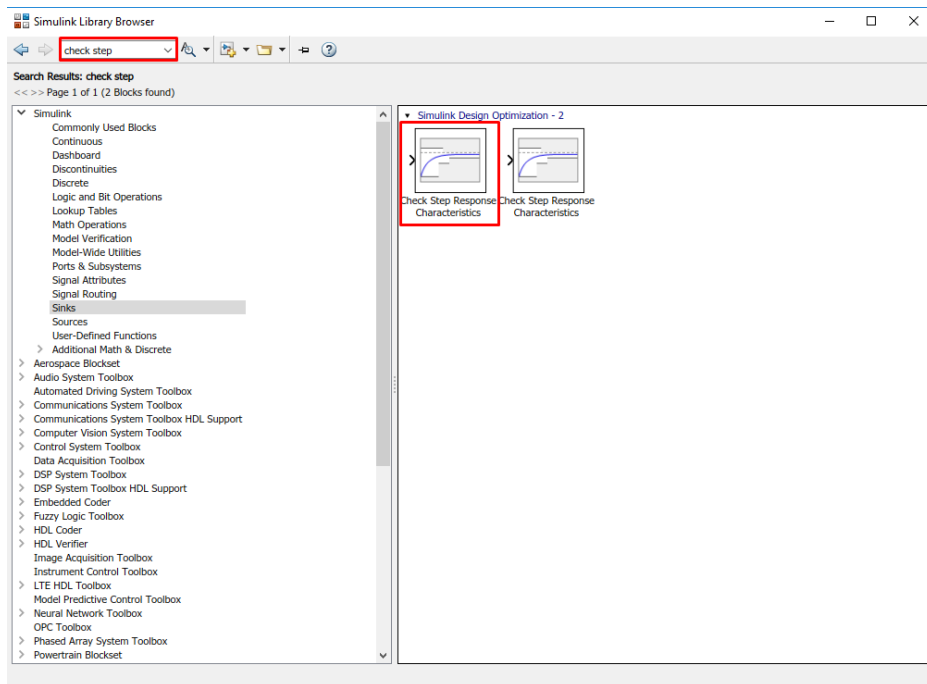


Рисунок 4.16 — Библиотека компонентов Simulink

11. Соединяем все компоненты как показано на рисунке 4.17



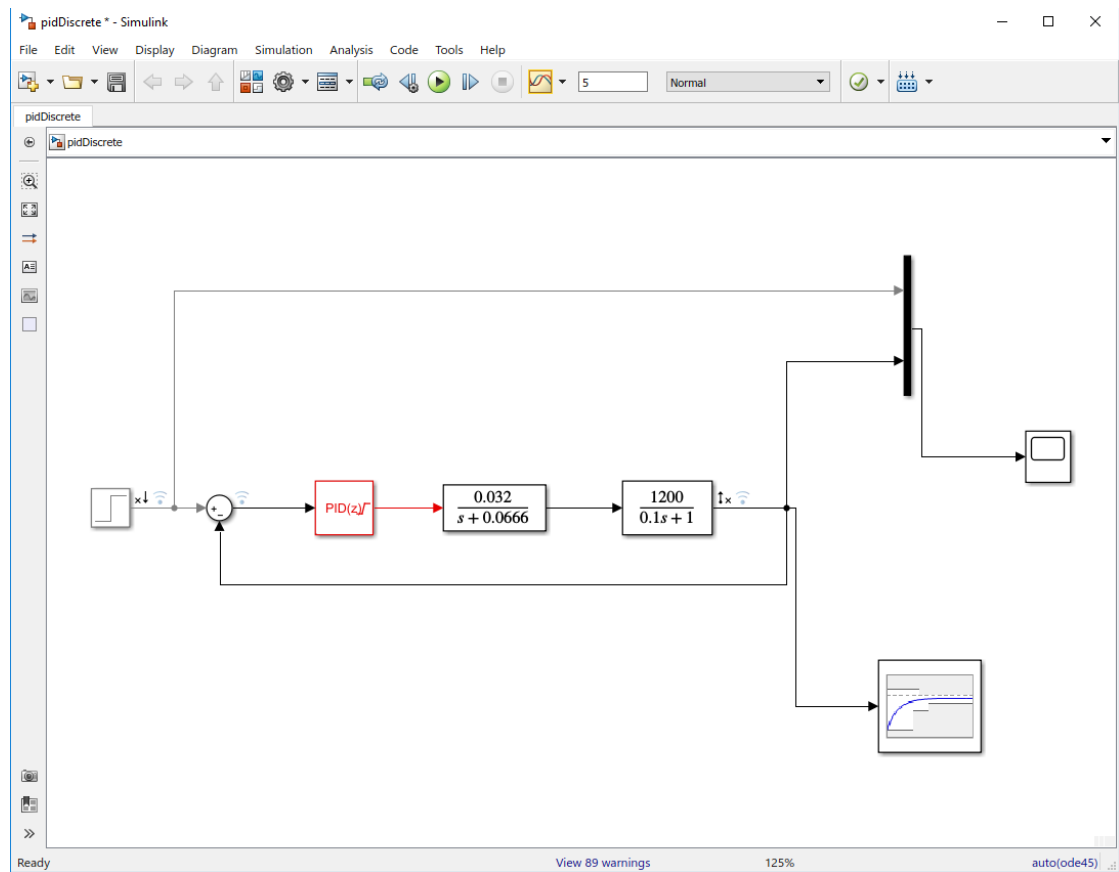


Рисунок 4.17 — Структурная схема контура регулирования

### 3.3 РАСЧЕТ ПИД РЕГУЛЯТОРА В ПЕРВОМ ПРИБЛИЖЕНИИ

В качестве начального приближения зададимся коэффициентами дифференциальной составляющей  $K_d = 0.1$ , интегральной составляющей  $K_i = 0.1$ , пропорциональной составляющей  $K_p = 0.1$  и коэффициент фильтрации  $N=1$ .

#### 3.3.1 Порядок расчета модели

1. Открываем вкладку Model Workspace в которой будут храниться глобальные переменные модели.

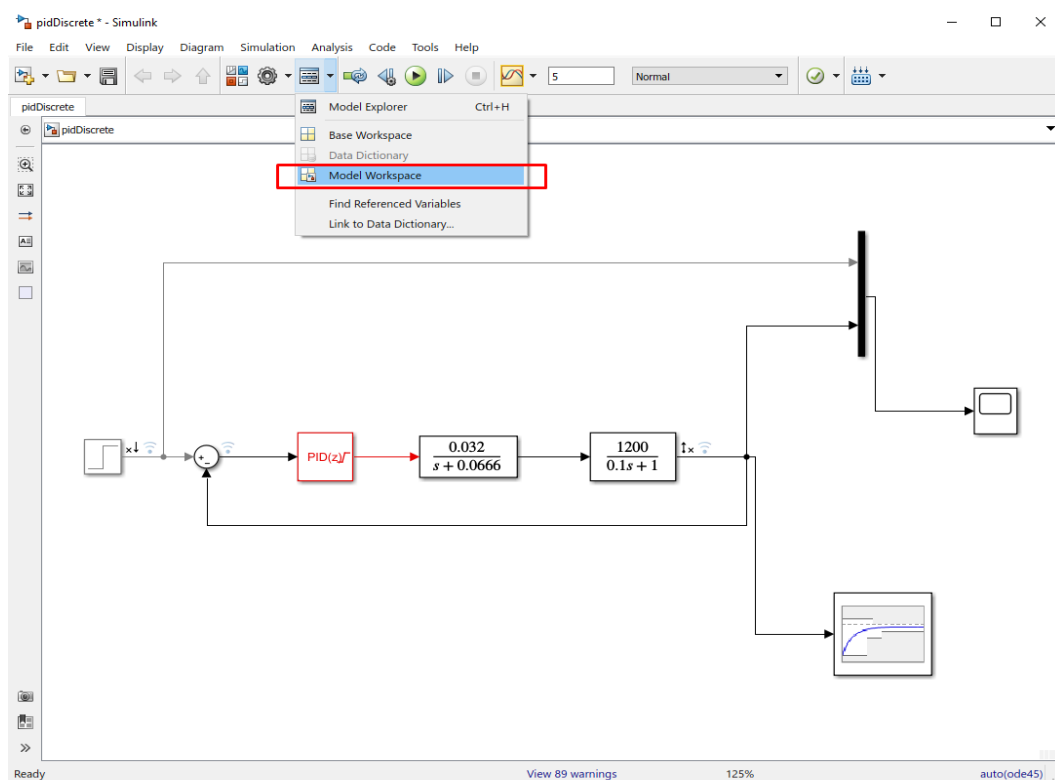


Рисунок 4.18 — Основное окно Simulink

2. В окне Model Explorer нажимаем кнопку Add MATLAB Variable. В появившееся поле вводим имя переменной и ее значение. Повторяем это для всех переменных.

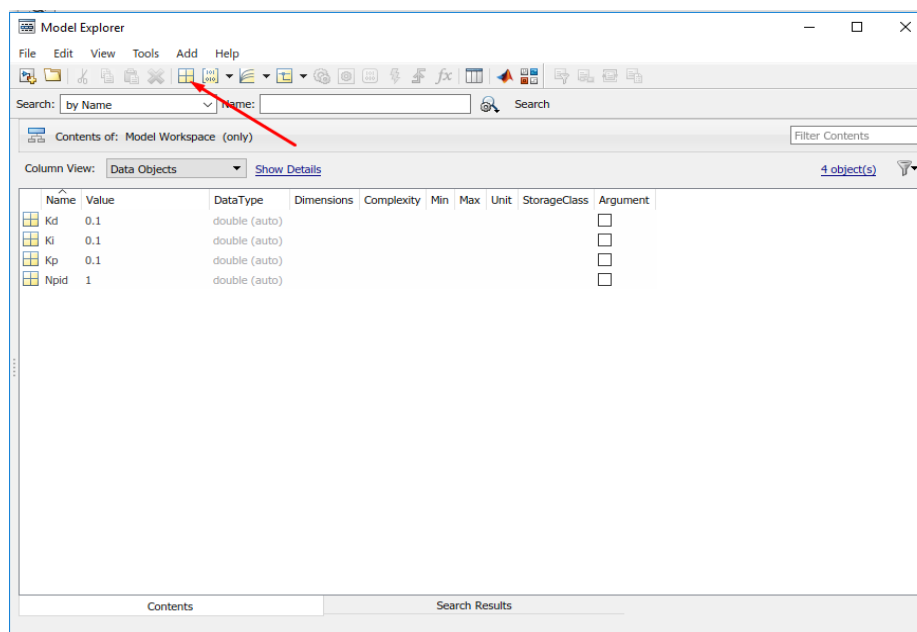


Рисунок 4.19 —Окно Model Explorer

3. Переходим на схему контура регулирования(которую построили ранее) и открываем окно настройки ПИД регулятора:
  - а) Переключаемся на дискретное время в области Time domain.
  - б) Заполняем поля Proportional, Integral, Derivative и Filter coefficient глобальными переменными которые создали в предыдущем шаге.
  - в) В поле Sample time вводим время дискретизации равное 20мс(0.02с)
  - г) Пререключаем Integration method и Filter method на Trapezoidal.

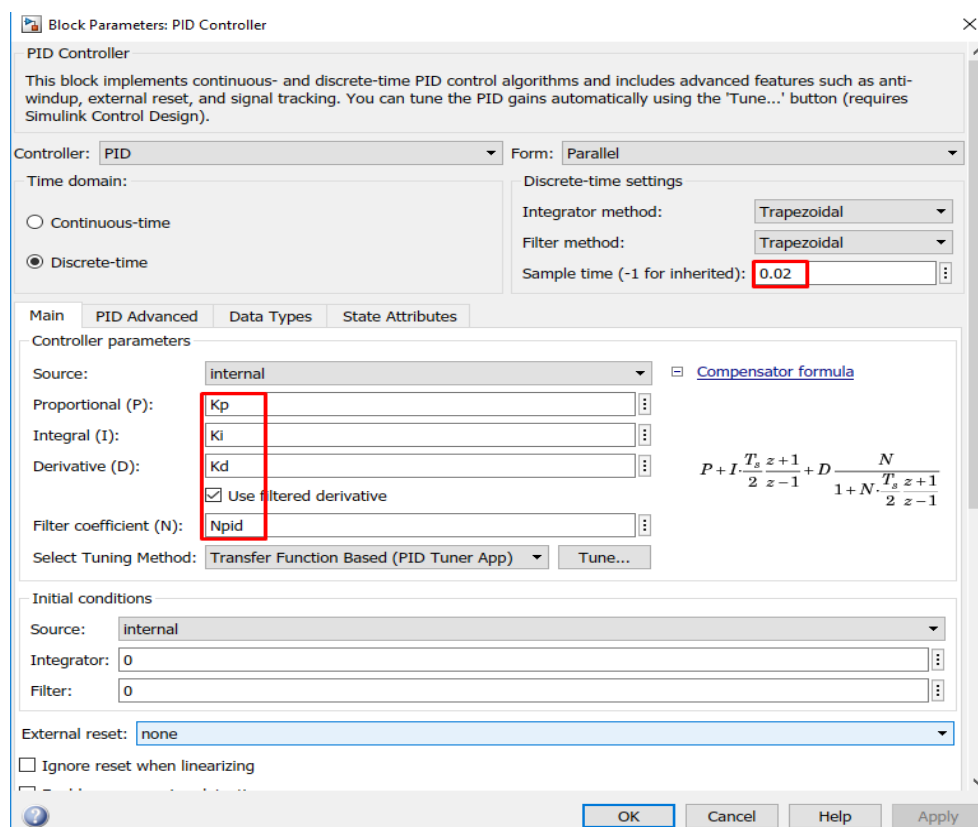


Рисунок 4.20 — Настройка компонента PID Controller

4. В верхнем меню нажимаем кнопку Run и запускаем симуляцию.

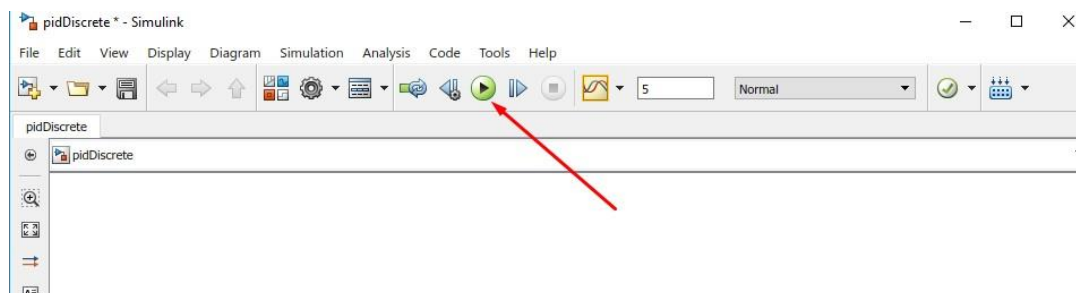


Рисунок 4.21 — Основное меню окна Simulink

5. В результате получаем график значений передаточной функции и единичного воздействия от времени. Если график не отобразился то нужно нажать дважды на компонент Scope на схеме.

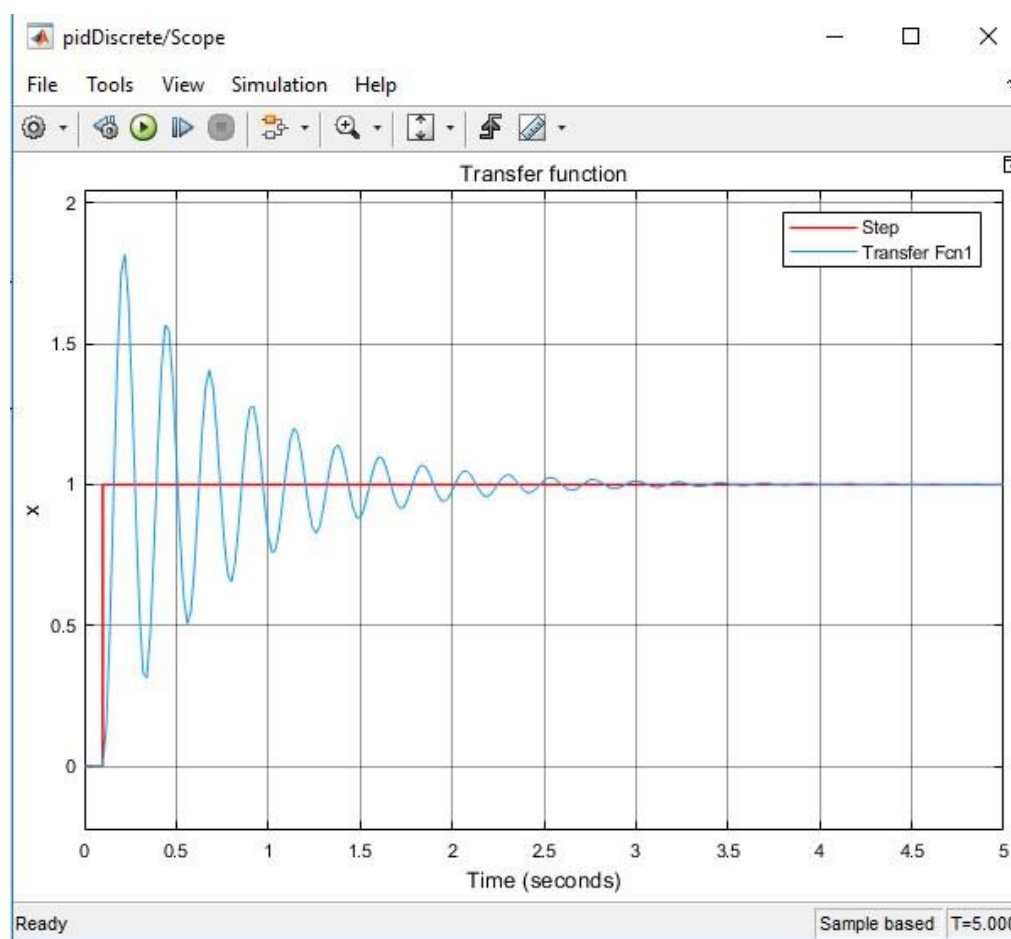


Рисунок 4.22 — График зависимости передаточной функции и единичного воздействия от времени

С этого графика можно сделать вывод: в начале временной шкалы присутствует большое перерегулирование и колебательность, что приводит к забросу температуры и неустойчивости соответственно. Также большое время выхода на заданный режим, порядка 4 секунд.

### 3.4 ИНТЕГРАЛЬНЫЕ ОЦЕНКИ КАЧЕСТВА ПЕРЕХОДНОГО ПРОЦЕССА

Этот способ оценки качества регулирования дает возможность сделать заключение о скорости затухания и величине отклонений регулируемой величины от установившегося значения. Данный способ может быть применен как к линейным, так и к нелинейным системам автоматического регулирования. В некоторых случаях, например, когда система регулирования находится в режиме управления, интегральные оценки удобнее рассматривать не по параметру  $x(t)$ , определяемому уравнением вынужденного движения (4.42), а по параметру  $z(t)$ . Связь между параметрами  $z(t)$  и  $x(t)$  видна из рис. 4.24, а и устанавливается выражением

$$z(t) = x(\infty) - x(t).$$

Вид переходного процесса  $z(t)$  (рис. 4.24, б) соответствует уравнению свободного движения

$$(a_n p^n + a_{n-1} p^{n-1} + \dots + a_1 p + a_0) z = 0,$$

которое получается из предположения, что возмущение в системе регулирования было вызвано единичной функцией до начала отсчета времени, т.е., при  $t < 0$  возмущение  $x_1 = 1(t)$ , а при  $t \geq 0$  возмущение  $\delta_1 = 0$ .

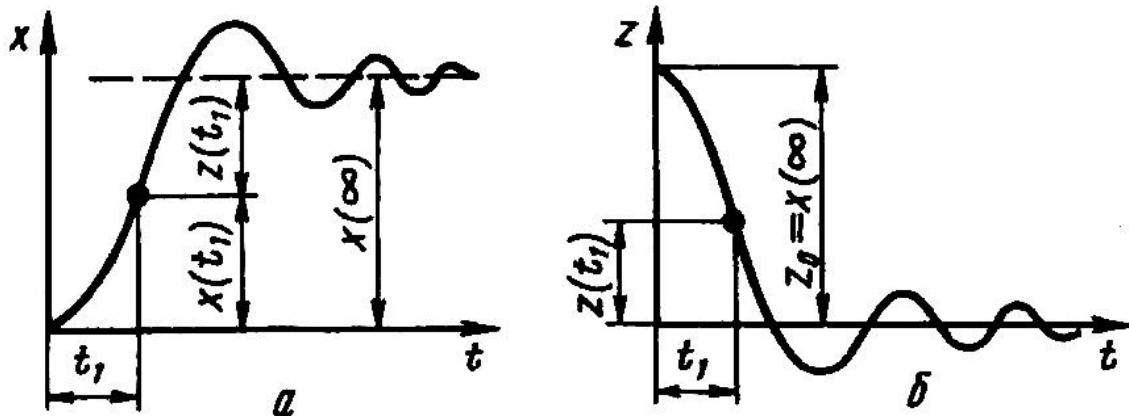


Рисунок 4.23 — График зависимости передаточной функции и регулирующего воздействия.

Допустим, что движения замкнутой системы регулирования описываются уравнением. Предположим, что можно определить величину какого-либо из интегралов, в подынтегральную функцию которых будет входить переменная  $z$  :

$$I_0 = \int_0^{\infty} |z| dt; \quad I_1 = \int_0^{\infty} z dt; \quad I_2 = \int_0^{\infty} z^2 dt.$$

Примем без доказательства положение, что качество регулирования системы тем выше, чем меньше величина одного из перечисленных интегралов. Достоинства интегральной оценки определяются тем, насколько она отражает качества переходного процесса и какие трудности встают на пути ее

определения. Наиболее просто определяется интеграл  $I_1$ . Действительно, применим преобразование Лапласа к уравнению движения системы (4.54). Так как

$$L\{z(t)\} = \int_0^{\infty} e^{-st} z(t) dt,$$

то при заданных начальных условиях и при  $s \rightarrow 0$

$$I_1 = \int_0^{\infty} z(t) dt = L\{z(t)\}.$$

Имеется и другой путь определения  $I_1$ . Проинтегрировав почленно уравнение свободного движения, получим

$$a_n \int_0^{\infty} z^{(n)} dt + a_{n-1} \int_0^{\infty} z^{(n-1)} dt + \dots + a_1 \int_0^{\infty} z^{(1)} dt + a_0 \int_0^{\infty} z dt = 0,$$

или

$$a_n z^{(n-1)} \Big|_0^{\infty} + a_{n-1} z^{(n-2)} \Big|_0^{\infty} + \dots + a_1 z \Big|_0^{\infty} + a_0 \int_0^{\infty} z dt = 0.$$

Подстановка верхнего предела во всех слагаемых дает нули, так как система устойчива и все ее производные при  $t \rightarrow \infty$  стремятся к нулю. Поэтому

$$I_1 = \int_0^{\infty} z dt = \frac{a_n z^{(n-1)}(0) + a_{n-1} z^{(n-2)}(0) + \dots + a_1 z(0)}{a_0},$$

где  $z^{(n-1)}(0)$ ,  $z^{(n-2)}(0)$ , ...,  $z(0)$  - заданные начальные условия.

Интеграл  $I_2$  можно вычислить с помощью вещественной частотной функции замкнутой системы и интеграла Фурье. Вычисление интеграла  $I_0$  трудоемко, поэтому он обычно не применяется. По величине интегралов  $I_0$ ,  $I_1$  и  $I_2$  нельзя высказать какие-либо строгие суждения о характере переходного процесса. Так, например, из двух переходных процессов, показанных на рисунке 4.24, а и б, предпочтение следует отдать аperiodическому переходному процессу рисунок 4.24, а. Однако величина интеграла  $I_1$  для колебательного процесса рисунок, 4.24, б будет меньше, чем для аperiodического, так как составляющие площади под кривой процесса суммируются алгебраически, с учетом знака.

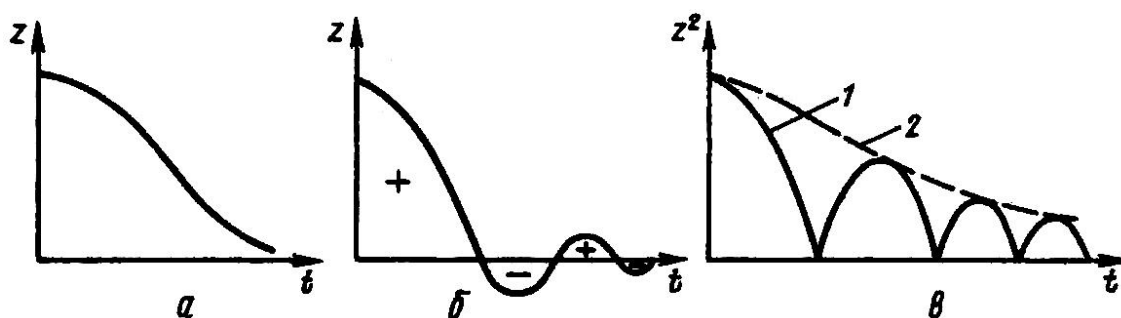


Рисунок 4.24 — График зависимости интегральных критериев.

Этого недостатка, присущего интегральной оценке  $I_1$ , нет у интегральной оценки  $I_2$ . Но и здесь не всегда получается объективная картина. Так, расчеты показывают, что  $I_{2\min}$  получается при изменении  $z(t)$  по закону

$$z(t) = z(0) \frac{\sin \omega t}{\omega t},$$

который характеризует медленно затухающий колебательный процесс. Соответствующая этой переходной функции подынтегральная функция  $z^2$  - кривая 1 на рисунке 4.24, в; желательное изменение  $z^2$  - кривая 2. И хотя величина  $I_2$  для колебательного процесса будет меньше, чем для апериодического, соответствующего кривой 2, предпочтение следует отдать апериодическому переходному процессу.

Более объективный результат дают те интегральные оценки, которые учитывают не только изменение функции  $z$ , но и ее производных. Если отразить влияние только первой производной на качество регулирования, то интегральная оценка (назовем ее обобщенной интегральной оценкой) примет вид

$$I_V = \int_0^{\infty} \left[ z^2 + \tau^2 \left( \frac{dz}{dt} \right)^2 \right] dt.$$

Значение интеграла  $I_V$  будет мало в том случае, если не будет длительных отклонений параметра  $z$  и не будет длительного существования больших значений производной  $dz/dt$ . Коэффициентом  $\tau$  следует задаться. Он определяет влияние производных в величине интеграла  $I_V$ .

### 3.5 РАСЧЕТ ИНТЕГРАЛЬНЫХ КРИТЕРИЕВ НА НАЧАЛЬНОЙ МОДЕЛИ

В качестве начальной модели будем использовать результаты которые получили ранее в п.4.3

#### 3.5.1 Порядок расчета

1. На схеме открываем настройки компонента Check Step Response Characteristics и переходим на вкладку Response Optimization

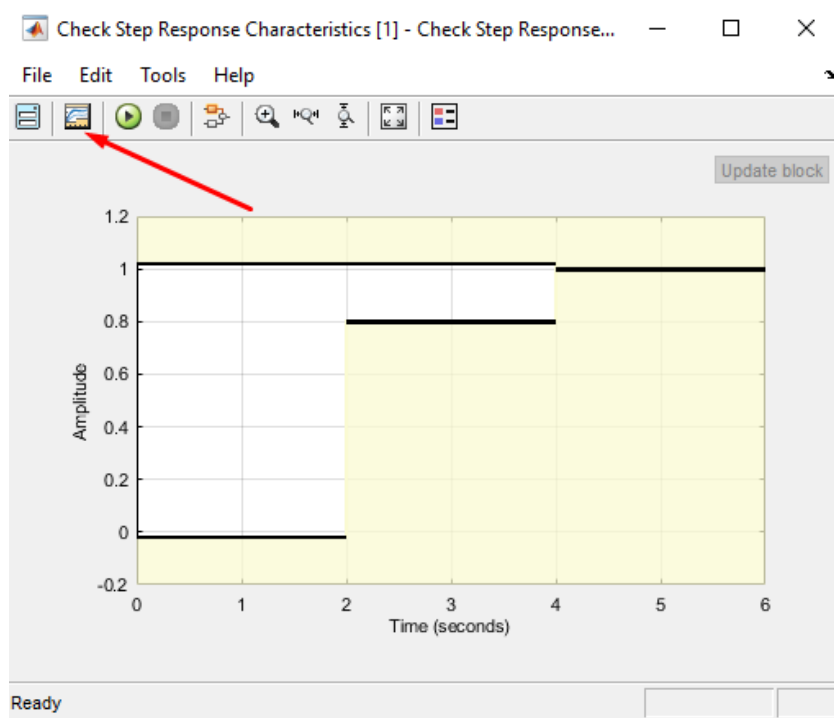


Рисунок 4.25 — Настройки компонента Check Step Response Characteristics.

2. В окне Response Optimization находим поле Design Variables Set, в котором создаем набор переменных которые будут изменяться в процессе оптимизации модели. С левого окна с помощью кнопки стрелочка перемещаем переменные в правое окно. Эти переменные были созданы на тапе наспройки компонента ПИД регулятор.



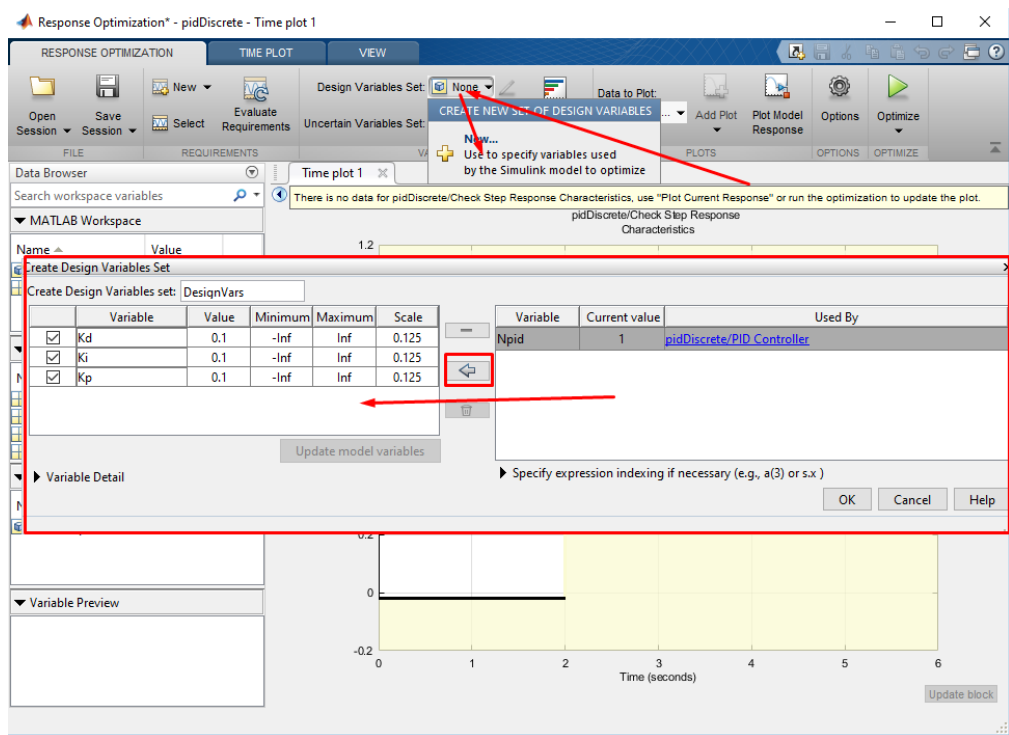


Рисунок 4.26 — Окно добавления оптимизируемых переменных в модель

3. Переходим к настройке ограничений. В верхнем меню нажимаем кнопку Select, в открывшемся окне представлены ограничения по модели. Редактируем требование нажав на кнопку Edit. В открывшемся окне задаем ограничения по модели:
  - Начальное значение Initial value = 0
  - Конечное значение Final value = 1
  - Время нарастания Rise time = 2с
  - Величина нарастания Rise = 80%
  - Время установки Setting time = 4с
  - Величина установки Setting 0.1%
  - Перерегулирование Overshoot = 2%
  - Недерегулирование Undershoot = 2%

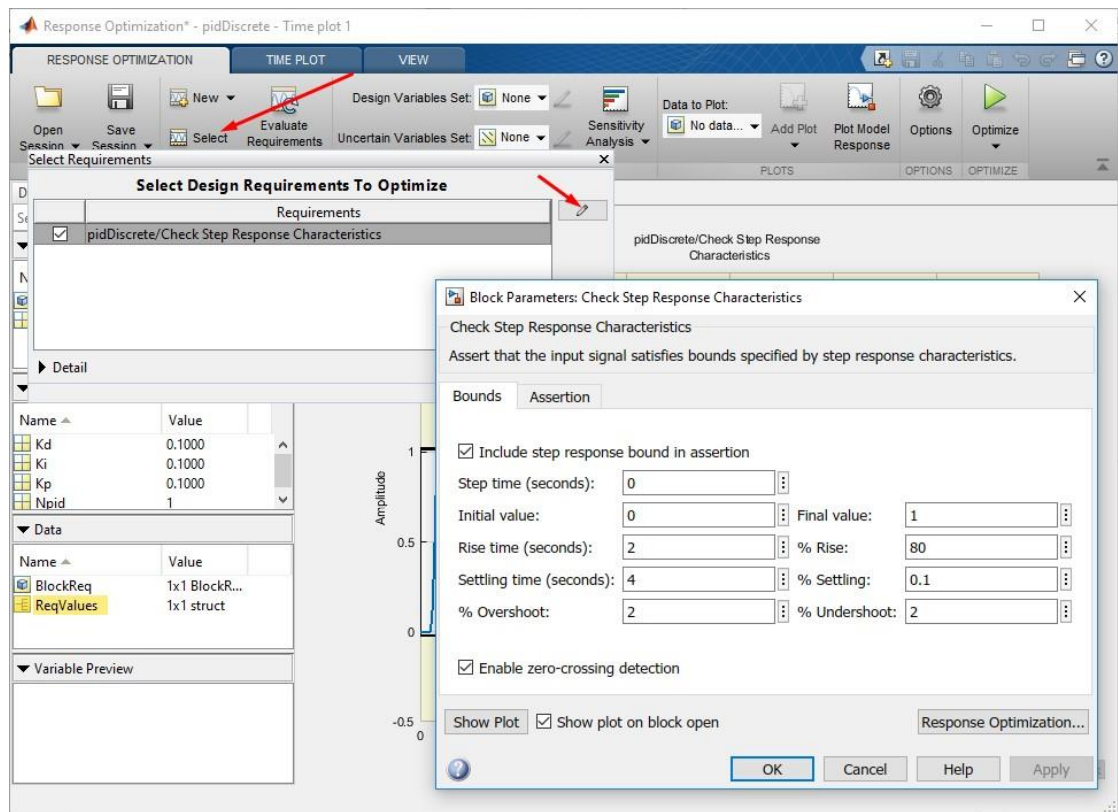


Рисунок 4.27 — Настройка встроенных ограничений

4. В папке в которой хранится проект создаем файл `integralCriterion.m` (.m – расширение файла для m-функций Matlab) и отрываем его в любом удобном текстовом редакторе (в моем случае это Visual Studio Code).

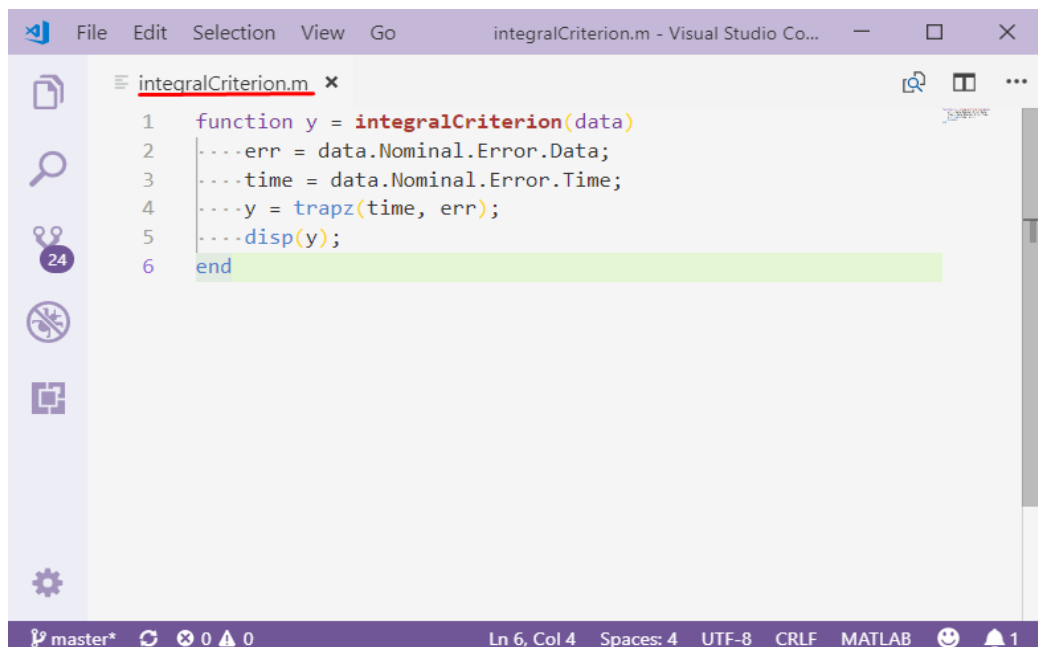


Рисунок 4.28 — Файл скрипта m-функции для  $I_1$ .

В строке 1 объявляем функцию `integralCriterion` (должна соответствовать имени файла) которая принимает аргумент `data` и возвращает `y`. Аргумент `data` в функцию передает сам Matlab при каждом вызове ее в процессе оптимизации.

Аргумент data - это dataset (объект) в котором нас интересует поле Nominal. В этом поле будут храниться все сигналы, которые мы позже зададим в настройках. Error – это сигнал(dataset) который мы передаем с модели (имя Error может быть любое, в зависимости от того как вы сами назовете свой сигнал) в котором есть два поля Data и Time данные и время соответственно.

Data и Time содержат в себе массивы чисел.

В строке 2 и 3 создаем переменные err и time присваиваем им значение данных(ошибки) и времени.

В строке 4 переменной y (которую возвращает функция) присваиваем внутреннюю функцию trapz() и передаем в нее аргументы time и err. Это

выражение будет эквивалентно интегральному критерию  $I = \int_0^t z dt$ ;

trapz(x, y) – это функция Matlab которая вычисляет интеграл от функции y по переменной x, используя метод трапеций. Аргументы x и y могут быть одномерными массивами одинакового размера.

В строке 5 выводим на печать y

End завершает тело функции в строке 6

5. Переходим на основное меню и создаем новое собственное ограничение.

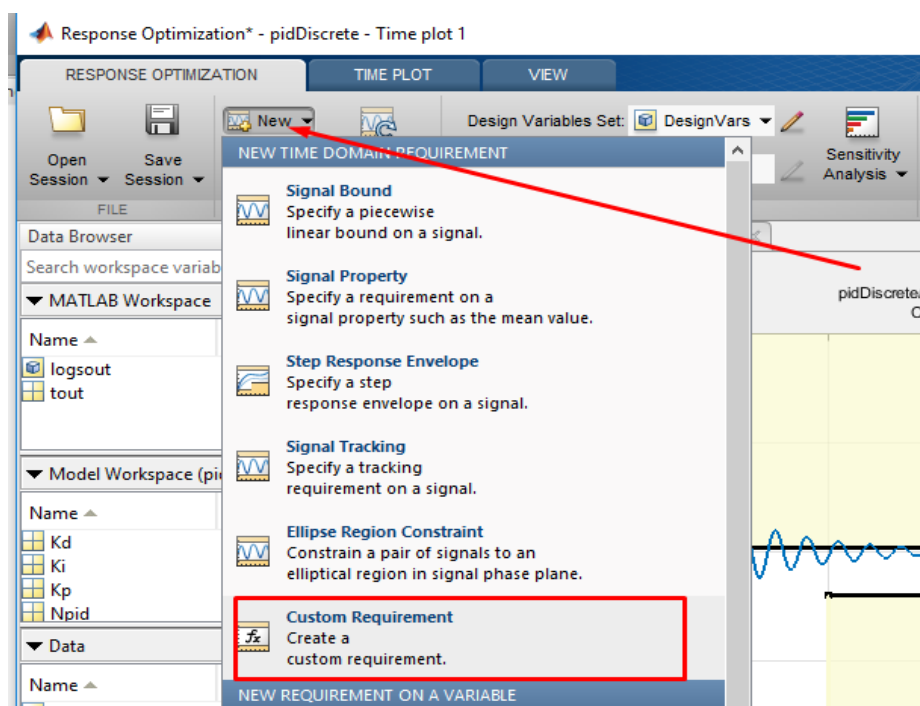


Рисунок 4.29 — Меню добавления новых ограничений.

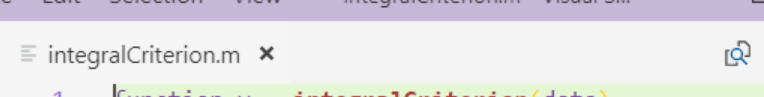
6. В открывшемся окне задаем:

- имя (может быть произвольным)
- Type – Minimize the function output (т.к мы ищем минимум функции)
- Function – имя функции должно соответствовать имени файла который мы создали ранее и начинаться с знака @

- 
- The screenshot displays the MATLAB/Simulink 'Response Optimization' tool interface. The main window shows the 'TIME PLOT' tab. A 'Custom Requirement' dialog box is open, allowing the user to define a custom requirement. The dialog has the following fields and options:
- Name:** 'integralCriterion' (highlighted with a red box and arrow 1).
  - Specify Function:**
    - Type: 'Minimize the function output' (dropdown menu).
    - Function: '@integralCriterion' (highlighted with a red box and arrow 2).
    - ☐ Error if constraint is violated.
  - Select Signals and Systems to Bound (Optional):**
    - Signal: 'Error (pidDiscrete/Sum:1)' (highlighted with a red box and arrow 3).
- A 'Create Signal Set' dialog box is also open, showing the signal set 'Error' (highlighted with a red box and arrow 4) and the signal 'pidDiscrete/Sum:1' (highlighted with a red box and arrow 5). The 'Create Signal Set' dialog has the following fields and options:
- Name:** 'Error' (highlighted with a red box and arrow 6).
  - Signal set:** 'Error' (highlighted with a red box and arrow 6).
  - Currently selected signals:** 'pidDiscrete/Sum:1' (highlighted with a red box and arrow 4).
  - Signal:** 'pidDiscrete/Sum:1' (highlighted with a red box and arrow 5).
- Red arrows and numbers 1 through 6 indicate the sequence of steps for creating the custom requirement. The background shows a Simulink model with a 'PID(2)' block and a 'Scope' block.

7. В верхнем меню нажимаем кнопку Evaluate Requirements. В основную консоль Matlab выведется значение интегрального критерия.
8. Подменяя файлы m – функции рассчитываем все критерии.

$$I_2 = \int_0^\infty |z| dt; \quad I_3 = \int_0^\infty z^2 dt; \quad I_4 = \int_0^\infty \left[ z^2 + \tau^2 \left( \frac{dz}{dt} \right)^2 \right] dt,$$



The screenshot shows the MATLAB editor window with the file 'integralCriterion.m' open. The script content is as follows:

```
1 function y = integralCriterion(data)
2     ...err = data.Nominal.Error.Data;
3     ...time = data.Nominal.Error.Time;
4     ...y = trapz(time, abs(err));
5     ...disp(y);
6 end
```

The status bar at the bottom indicates the current position is 'Ln 1, Col 1', the file encoding is 'UTF-8', and the editor is running 'MATLAB'.

Рисунок 4.31 — Файл скрипта m-функции для  $I_2$

```
1 function y = integralCriterion(data)
2     ....err = data.Nominal.Error.Data.^2;
3     ....time = data.Nominal.Error.Time;
4     ....y = trapz(time, err);
5     ....disp(y);
6 end
```

Рисунок 4.32 — Файл скрипта m-функции для  $I_3$

```
1 function y = integralCriterion(data)
2     ....err = data.Nominal.Error.Data;
3     ....time = data.Nominal.Error.Time;
4     ....tau = 2;
5     ....d = tau^2.*diff(err).^2;
6     ....d(end+1) = 0;
7     ....er = err.^2;
8     ....f = er + d;
9     ....y = trapz(time, f);
10    ....disp(y);
11 end
```

Рисунок 4.33 — Файл скрипта m-функции для  $I_4$

Для построения графиков сохранения результата расчета будем использовать вспомогательный скрипт myplot.m.

```
1 function y = intPlots(time, data)
2     ....err = data;
3     ....y = trapz(time, data);
4     ....plot(time, err);
5     ....disp(y);
6     ....file=fopen('Area.txt','w');
7     ....fprintf(file, 'Area=%f', y);
8     ....fclose(file);
9 end
```

Рисунок 4.34 — Файл вспомогательного скрипта для myplot.m

Запускаем его в основной консоли Matlab передавая в качестве аргументов время и данные с глобального объекта `logout`. `Logout` является датасетом и содержит в себе все выходные данные расчета.

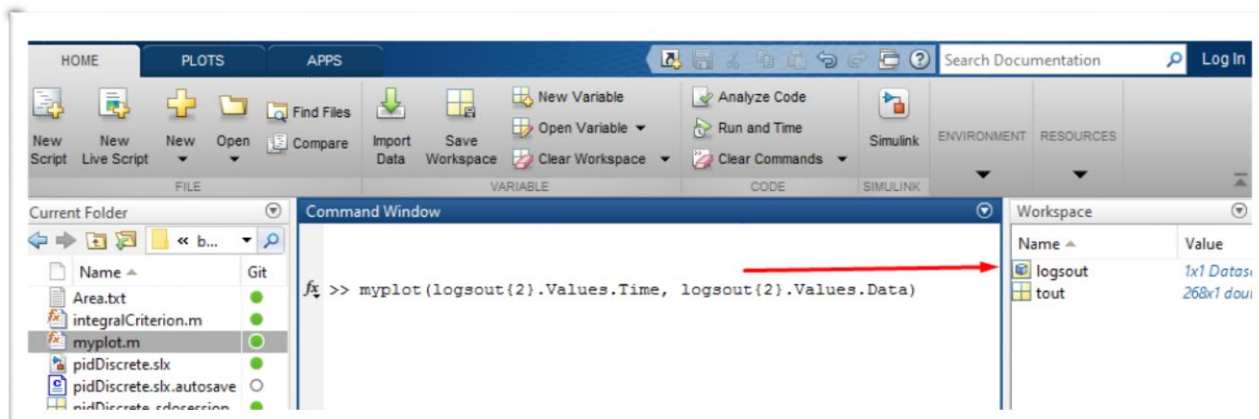


Рисунок 4.35 — Основная консоль Matlab.

В результате получаем график и файл `Area.txt` результата со значением интегрального критерия.

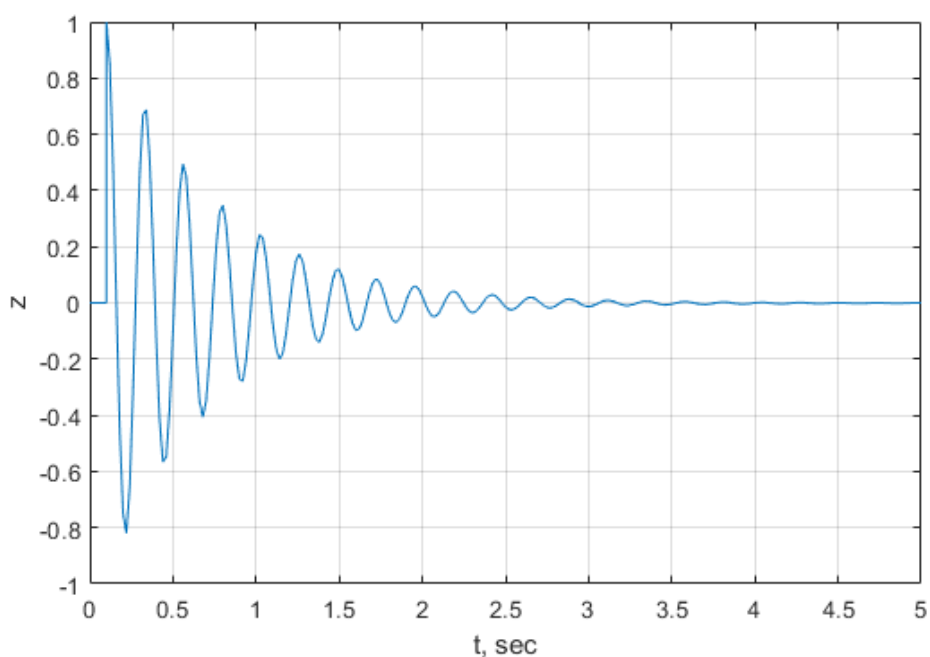


Рисунок 4.36 — График зависимости интегрального критерия  $I_1$ .

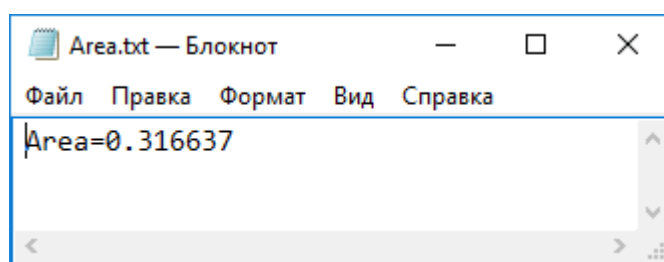


Рисунок 4.37 — Файл результата для интегрального критерия  $I_1$ .

Файлы скриптов и файлы результатов для критериев  $I_2, I_3, I_4$  представлены на рисунках 4.38 - 4.43 соответственно.

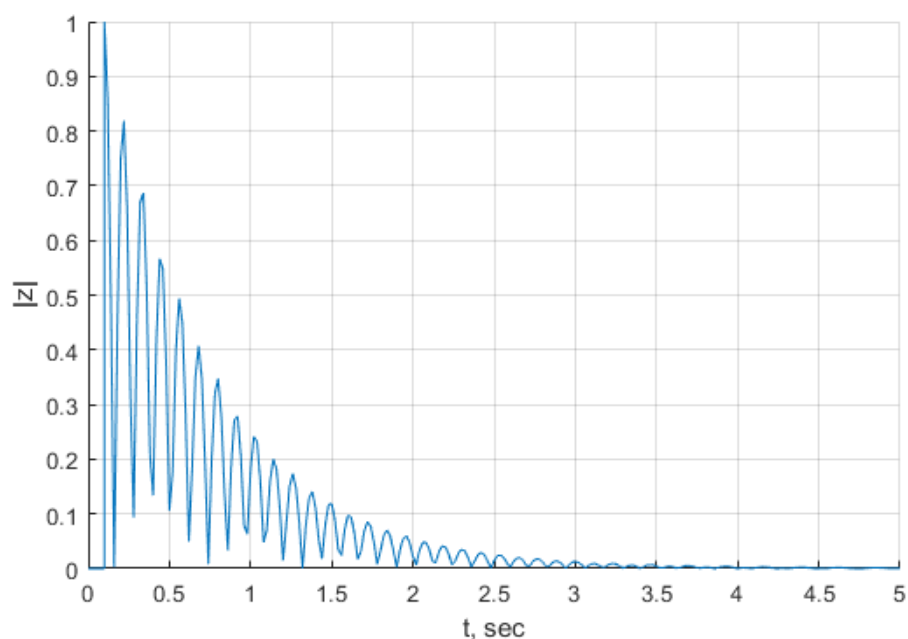


Рисунок 4.38 — График зависимости интегрального критерия  $I_2$ .

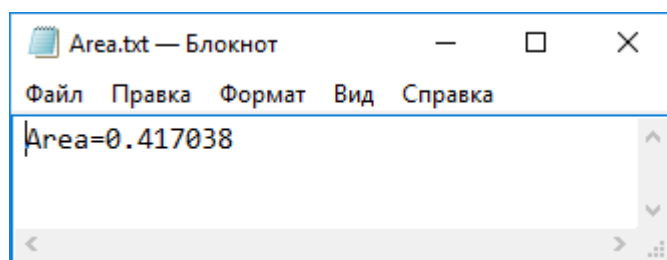


Рисунок 4.39 —Файл результата для интегрального критерия  $I_2$ .

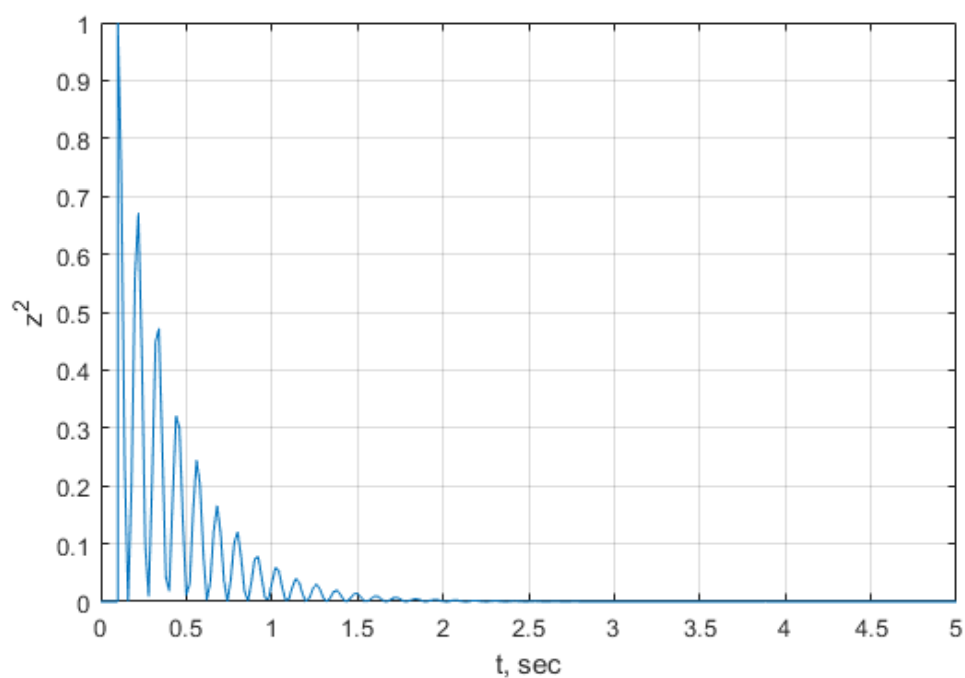


Рисунок 4.40 — График зависимости интегрального критерия  $I_3$ .

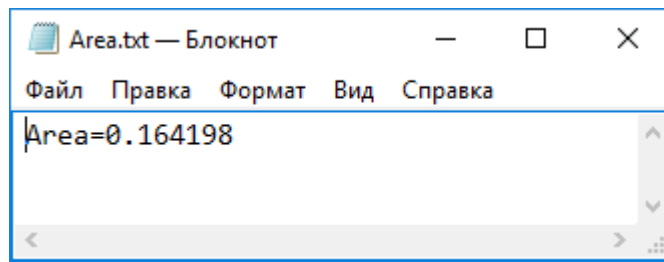


Рисунок 4.41 —Файл результата для интегрального критерия  $I_3$ .

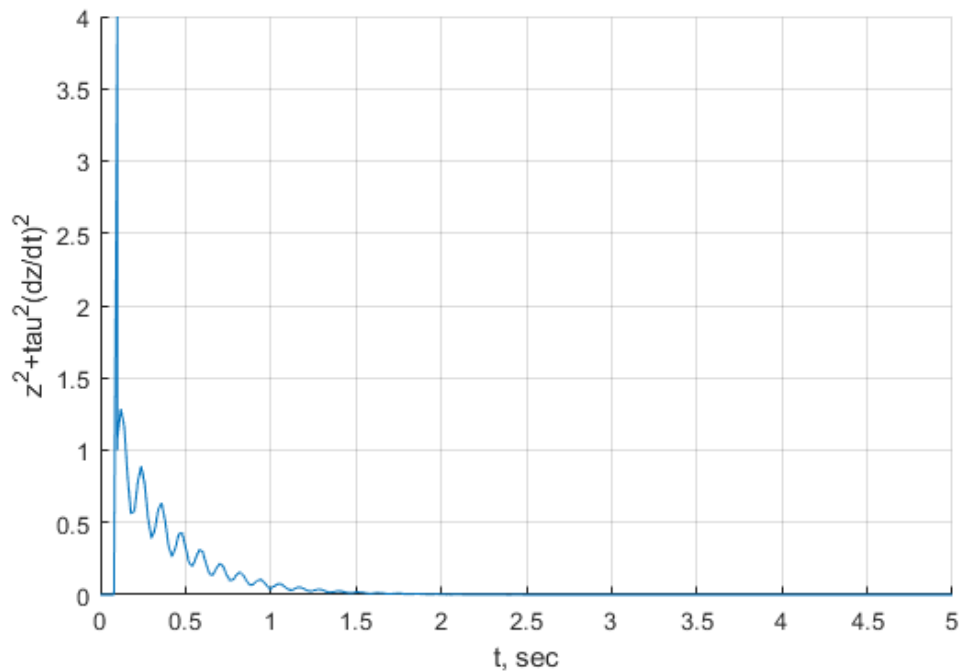


Рисунок 4.42 — График зависимости интегрального критерия  $I_4$ .

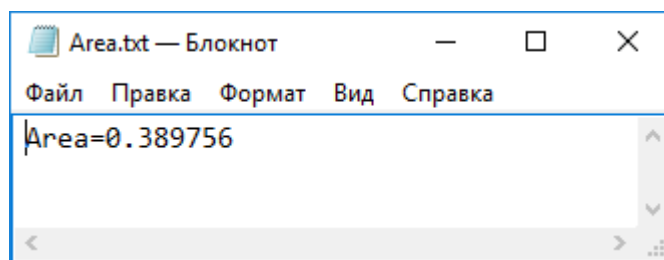


Рисунок 4.43 —Файл результата для интегрального критерия  $I_3$ .

В качестве интегрального критерия для дальнейших расчетов выбираем последний  $I_4$ . Этот критерий позволяет учитывать скорость изменения параметра  $z$  и задавать влияние скорости на качество переходного процесса. Так же является менее колебательным чем все остальные.



### 3.6 ОПТИМИЗАЦИЯ ПИД РЕГУЛЯТОРА

Для задачи оптимального синтеза ПИД регулятора будем использовать встроенные в Response Optimisation методы оптимизации. В работе будут использоваться:

- Генетический алгоритм
- Градиентный спуск
- Латинский гиперкуб
- Симплексный метод
- Метод Нелдера – Мида

#### 3.6.1 Порядок проведения оптимизации:

1. Продолжая п.4.5.1 предыдущей главы, уже с выбранным интегральным критерием, в основном меню нажимаем на вкладку Options.

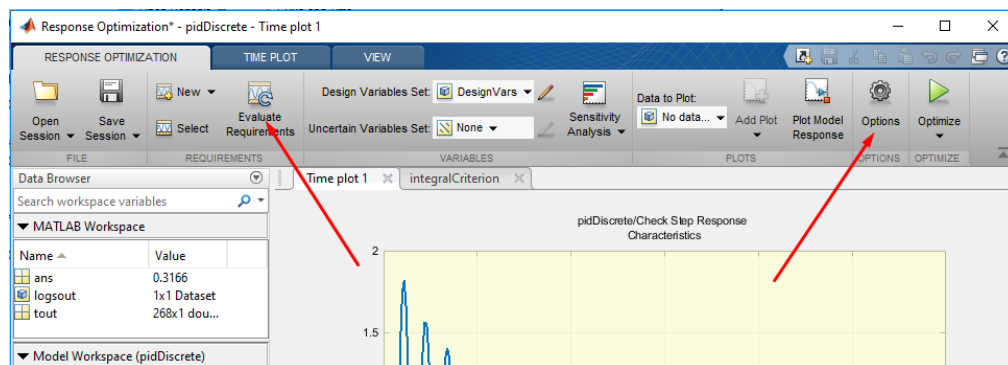


Рисунок 4.44 — Основное меню окна Simulink

2. В открывшемся окне переходим на вкладку Optimization Options выбираем оптимизационный метод в выпадающем списке Method.  
Задаем точность расчета для параметров, констант и функций. В нашем случае в размере 0.001.  
Так же задаем максимальное количество итераций для одного расчета (под количеством расчетов имеется ввиду количество перезапусков Restarts, задается несколько для случайных методов) равное 100.  
Для поиска максимально возможных значений отмечаем поле Look for maximally feasible solution.  
В поле Display level выбираем частоту отображения результата. Выставляем iteration что бы видеть результат при каждой итерации.

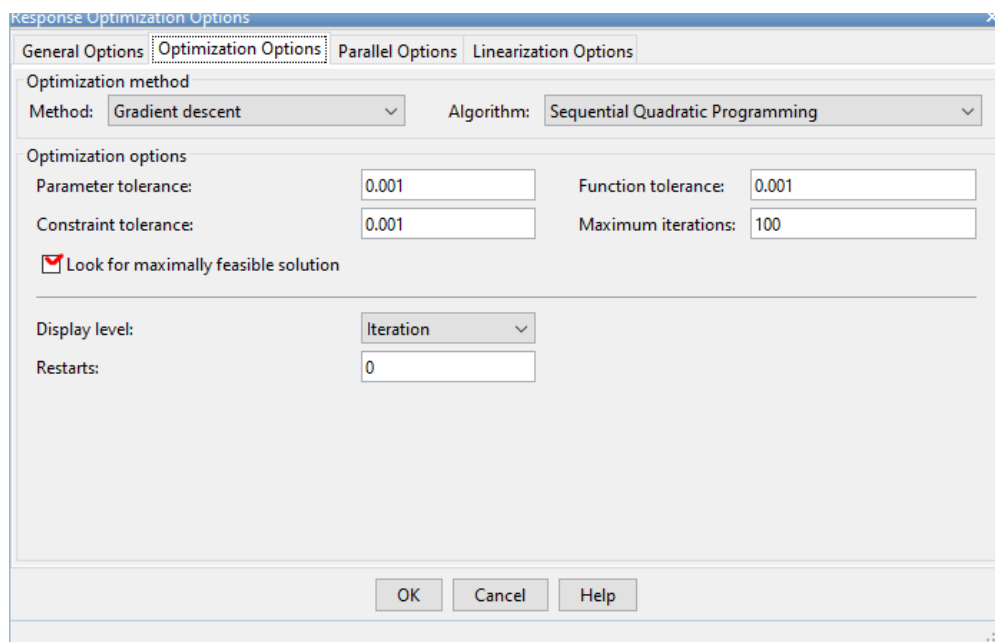


Рисунок 4.45 — Окно настройки оптимизации.

3. Под кнопкой Optimize нажимаем на стрелочку вниз. В открывшемся списке выбираем вкладку Open Optimization Report для того что бы следить за ходом расчета. Если вы не заинтересованы в этом можно сразу запустить расчет.

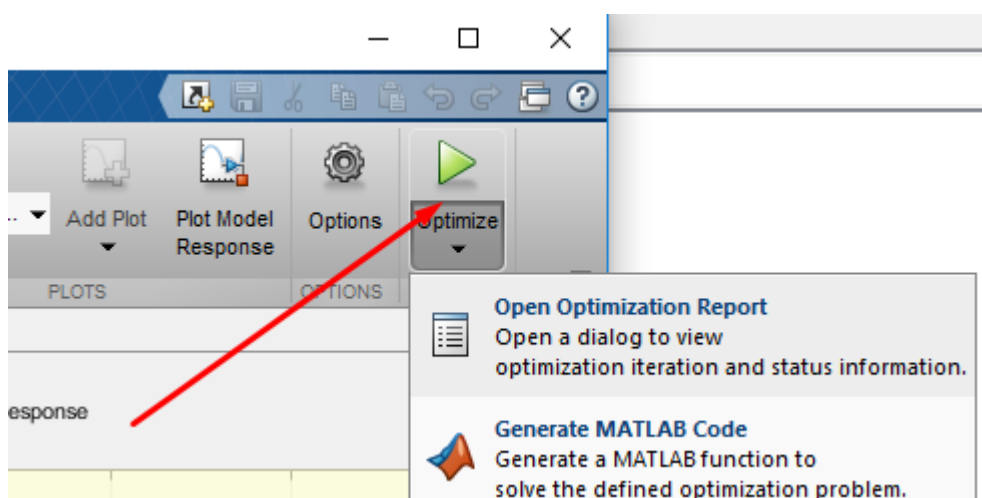


Рисунок 4.46 — Кнопка запуска оптимизации.

4. В открывшемся окне нажимаем кнопку Optimize тем самым запуская расчет. После окончания расчета в основном окне Simulink нажимаем кнопку Save для сохранения результатов оптимизации на нашей модели.

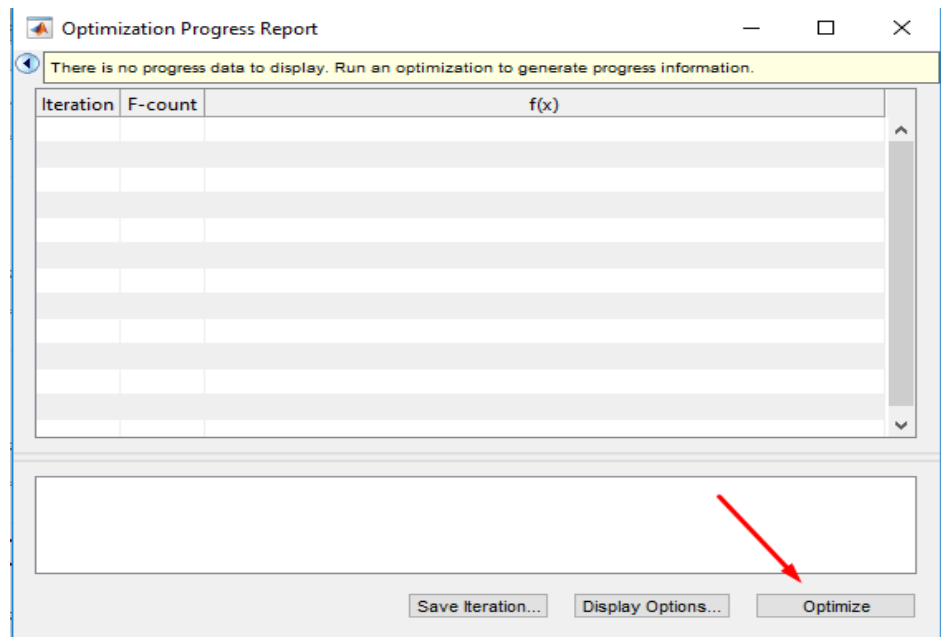


Рисунок 4.47 — Окно вывода результатов процесса оптимизации.

### 3.7 РЕЗУЛЬТАТЫ ОПТИМИЗАЦИИ ПИД РЕГУЛЯТОРА ВСТРОЕННЫМИ МЕТОДАМИ

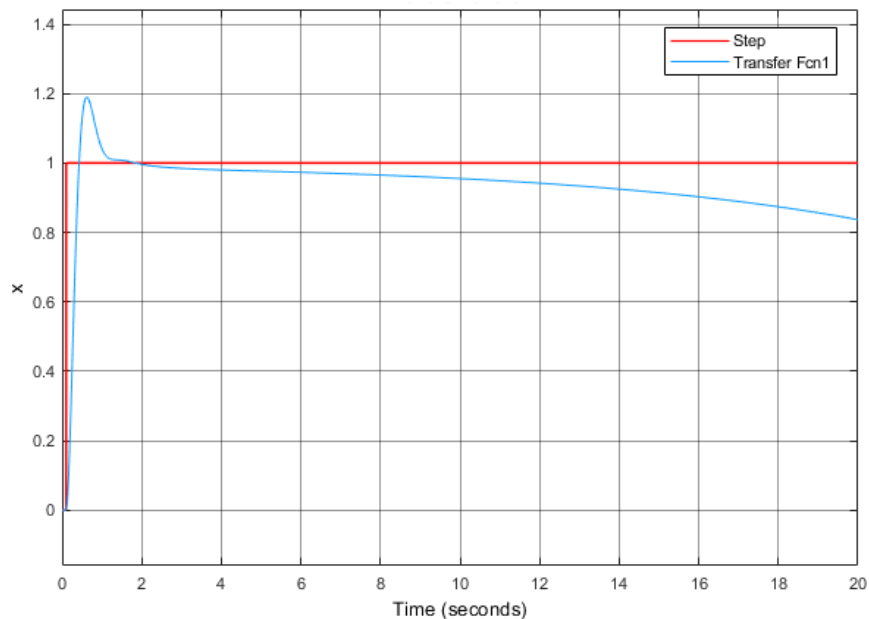


Рисунок 4.48 — График передаточной функции оптимизированной генетическим алгоритмом.

Генетический алгоритм не дал корректного результата. Дальнейшие расчеты по нему не проводились. Система не стабильна.

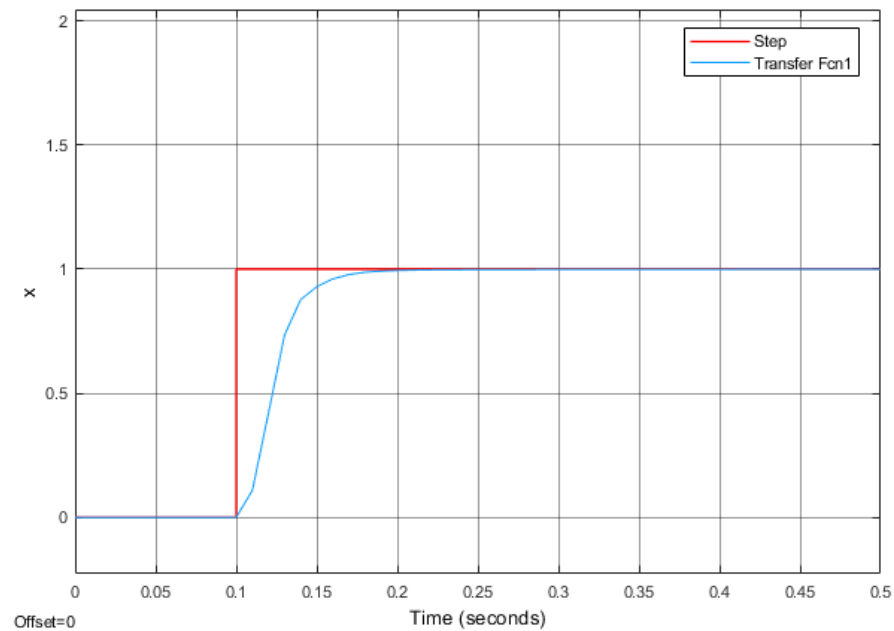


Рисунок 4.49 — График передаточной функции оптимизированной методом градиентного спуска (увеличен)

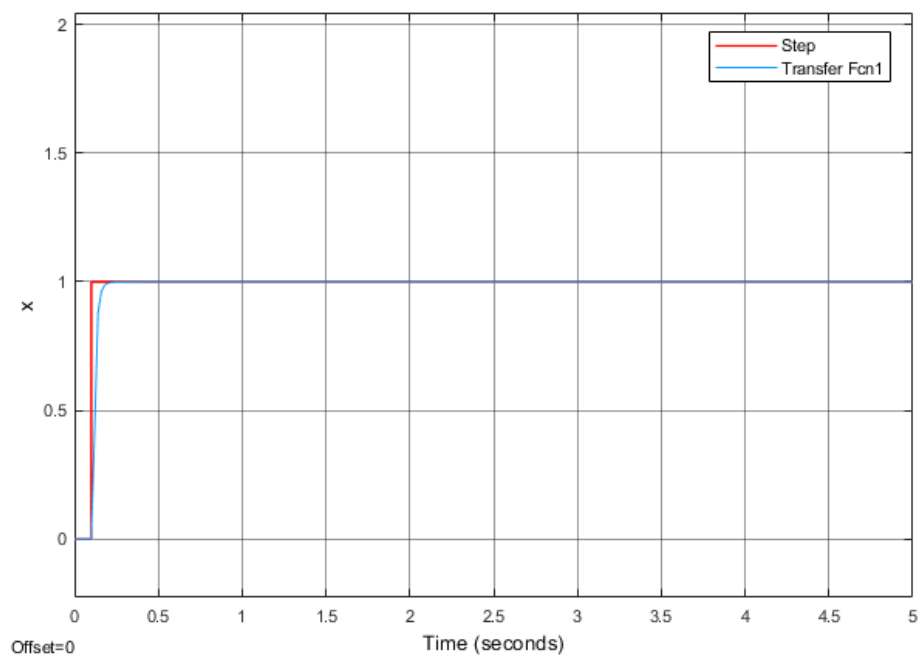


Рисунок 4.50 — График передаточной функции оптимизированной методом градиентного спуска (оригинал)

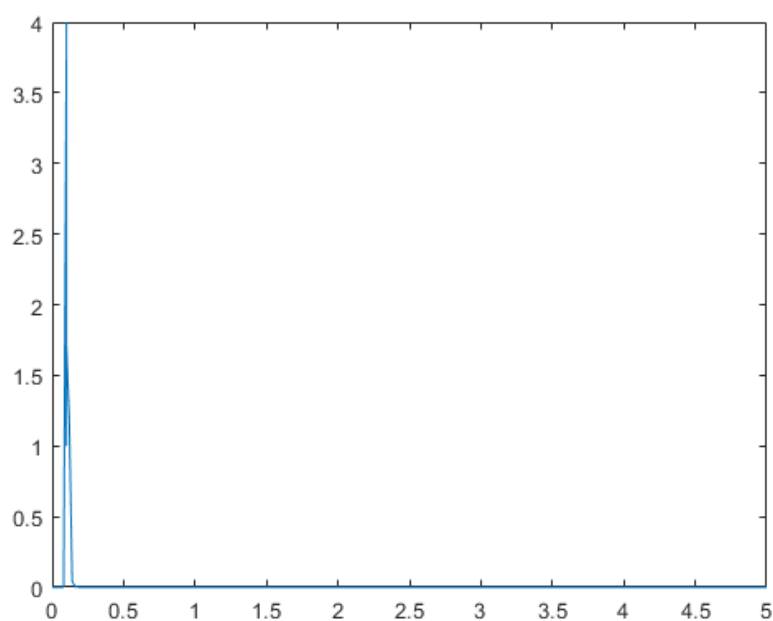


Рисунок 4.50 — График функции интегрального критерия оптимизированной методом градиентного спуска.

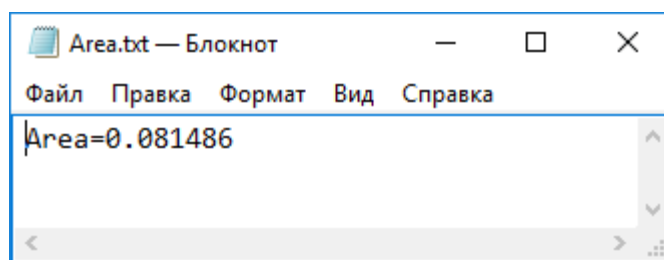


Рисунок 4.51 — Файл результата интегрального критерия, оптимизированного методом градиентного спуска.

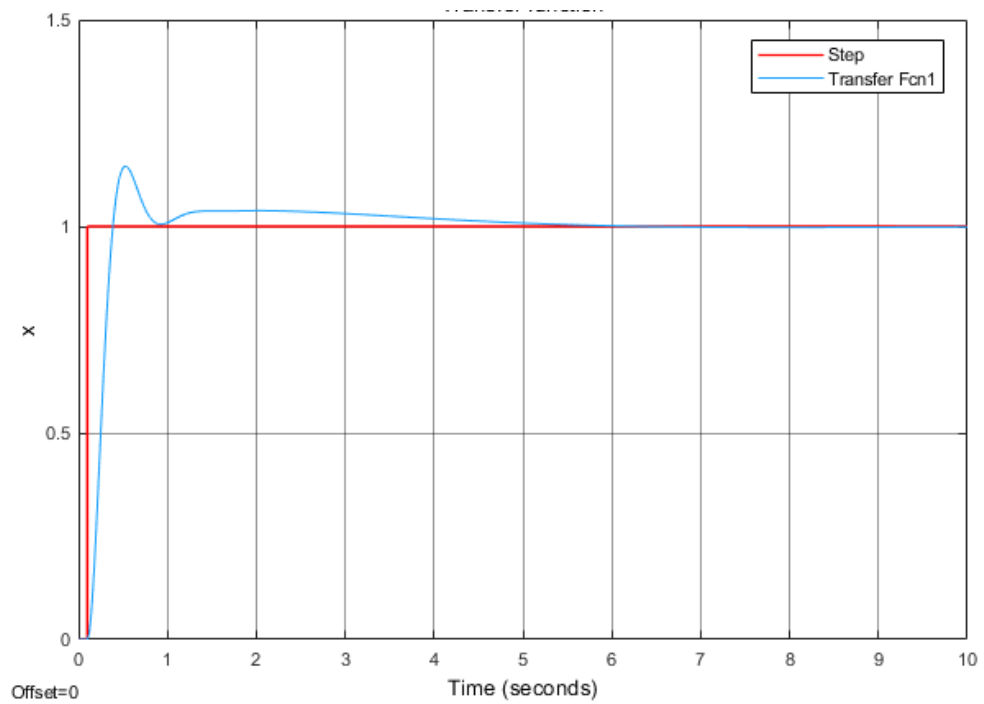


Рисунок 4.52 — График передаточной функции оптимизированной методом латинского гиперкуба

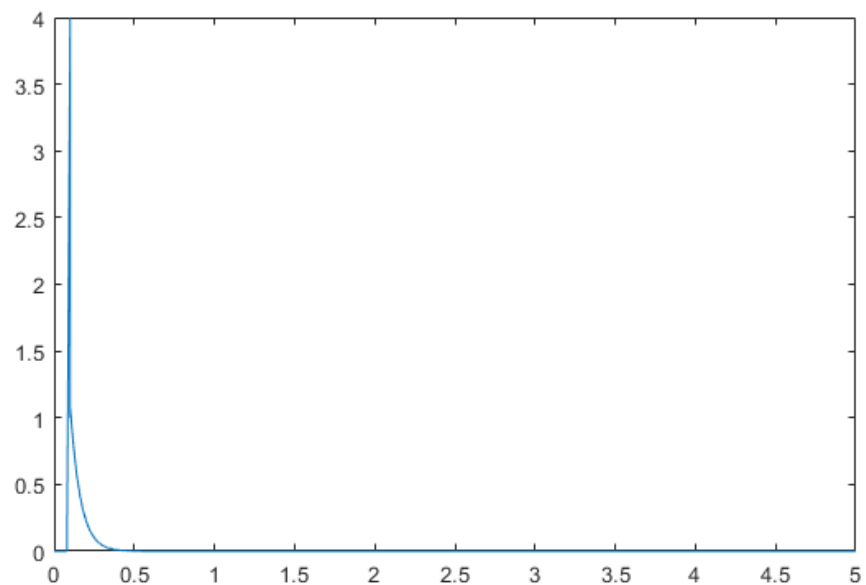


Рисунок 4.53 — График функции интегрального критерия оптимизированной методом латинского гиперкуба.

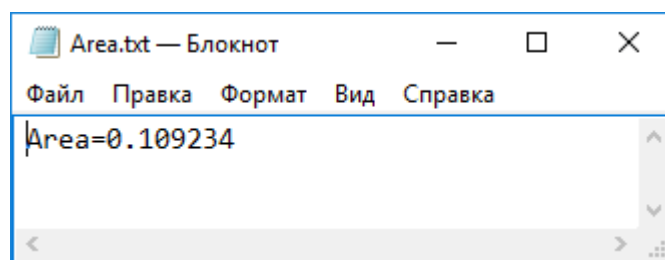


Рисунок 4.54 — Файл результата интегрального критерия, оптимизированного методом латинского гиперкуба.

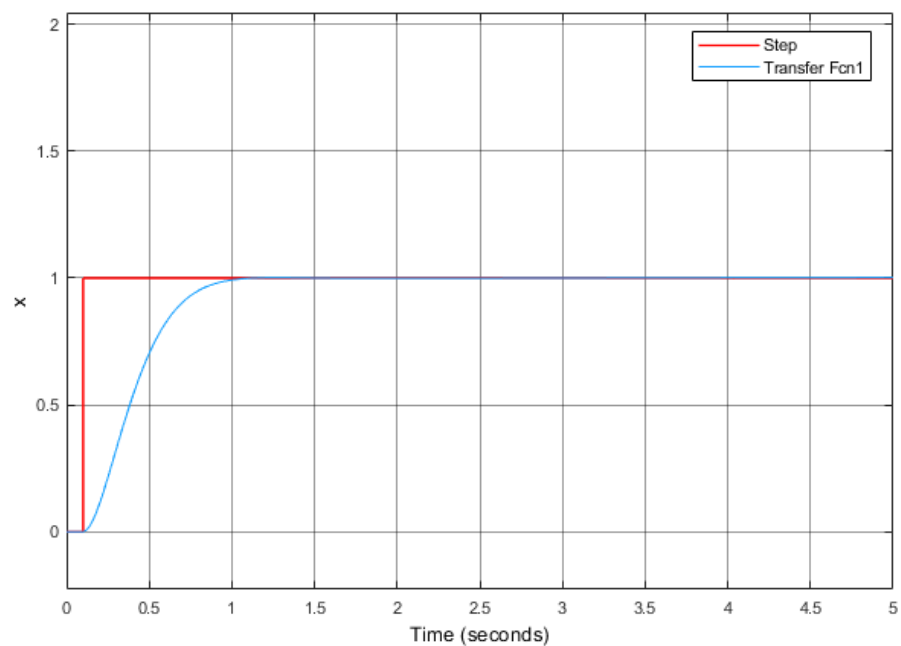


Рисунок 4.55 — График передаточной функции оптимизированной симплексным методом.

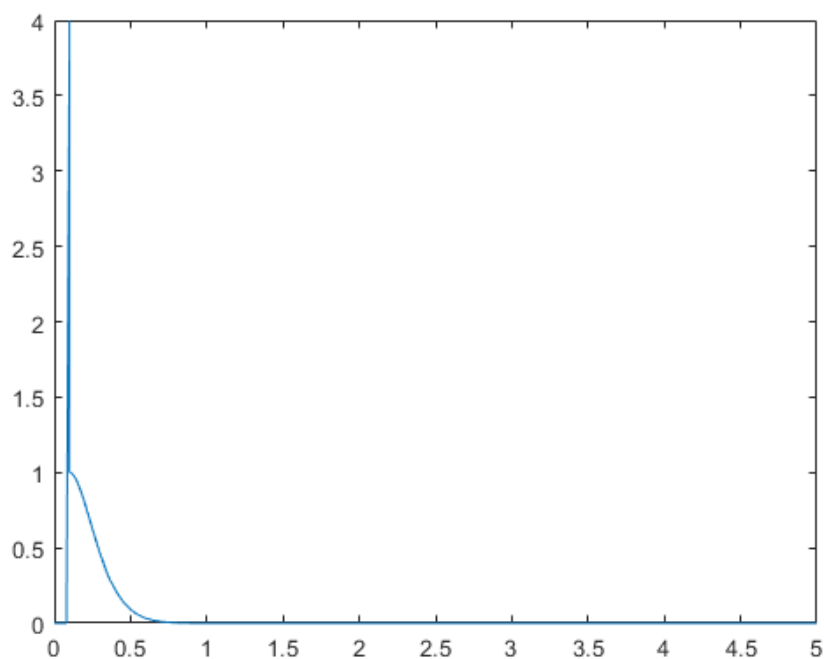


Рисунок 4.56 — График функции интегрального критерия оптимизированной симплексным методом.

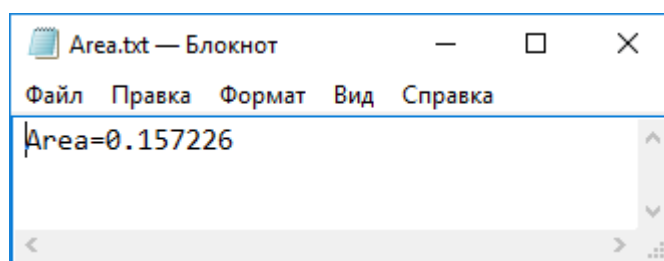


Рисунок 4.57 — Файл результата интегрального критерия, оптимизированного симплексным методом.

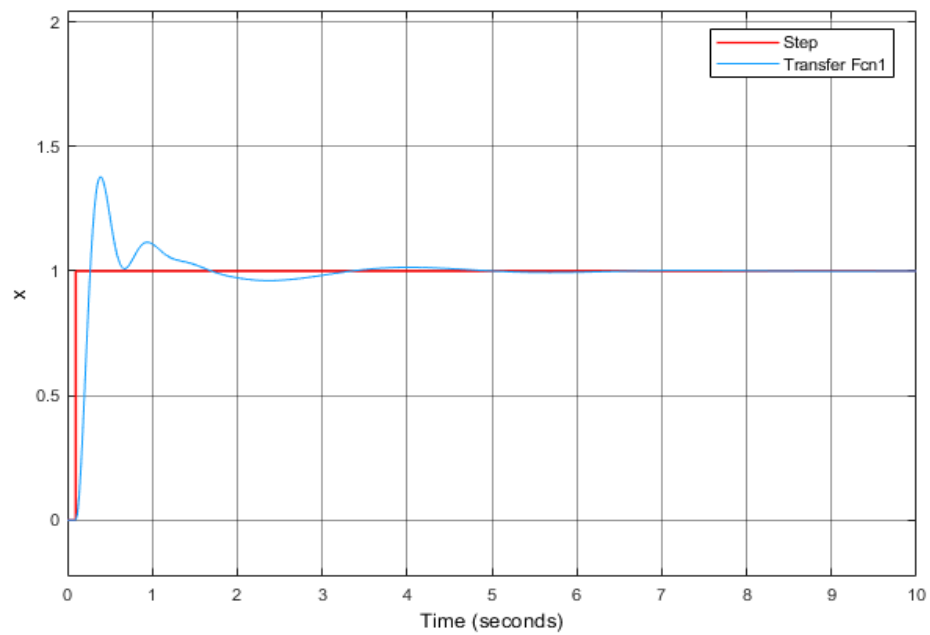


Рисунок 4.58 — График передаточной функции оптимизированной методом Нелдера – Мида

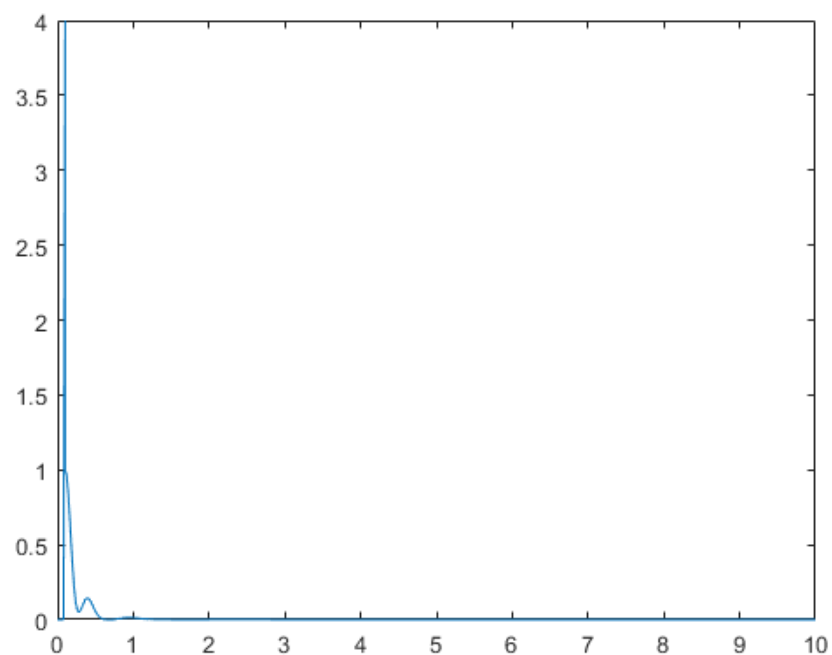


Рисунок 4.59 — График функции интегрального критерия оптимизированной методом Нелдера – Мида.

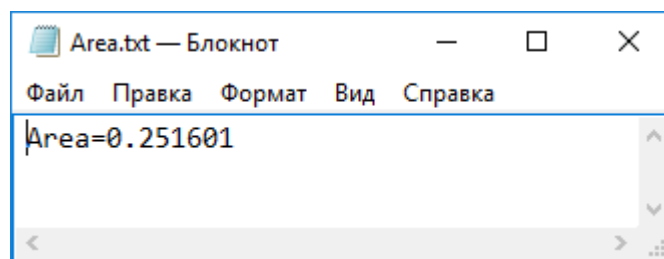


Рисунок 4.60 — Файл результата интегрального критерия, оптимизированного методом Нелдера – Мида.



Таблица 4.1— Результаты оптимизации встроенными методами

Метод	P	I	D	N	$I_4$
Генетический алгоритм	-	-	-	-	-
Градиентный спуск	0.739973	0.044392	0.07167	247.018166	0.081486
Латинский гиперкуб	0.194674	0.011667	0.018668	2287.377	0.109234
Симплексный метод	-0.787756	1.347698	1.099402	1.057434	0.157226
Метод Нелдера – Мида	0.054007	0.007649	0.216072	0.121223	0.251601

Проанализировав таблицу результатов можно сделать вывод что:

1. Генетический алгоритм не дал корректного результата. Дальнейшие расчеты по нему не проводились.
2. Наиболее оптимальное решение было получено методом градиентного спуска. Время выхода на режим около 0.2 секунд. Значения не выходят за заданные в расчете границы. Минимальное значение интегрального критерия из всех использованных методов.
3. При расчете методом латинского гиперкуба время выхода на режим составило около 5 секунд. При этом в начале времени наблюдается перерегулирование более 10%. Процесс колебательный, но стабилизируется к 7 секунде.
4. При расчете симплексным методом время выхода на режим составило 1 секунда. Колебательность не наблюдается.
5. При расчете методом Нелдера – Мида. Время выхода на режим около 5 секунд. При этом в начале времени наблюдается перерегулирование более 20%, что не допустимо, т.к. приведет к большому забросу температуры. Процесс колебательный, но стабилизируется к 7 секунде.

## ВЫВОД ПО РАЗДЕЛУ

В результате проведенной работы можно сказать, что улучшение системы управления и переход с аналоговых устройств на цифровые повышает срок службы двигателя и уменьшает затраты на проектирование и производство САУ.

Применение цифровой системы позволяет использовать универсальный контроллер для всех изделий данного типа двигателя. Изменяя только исполняемый код. Что существенно ускоряет процесс настройки САУ. Также позволяет увеличить контролируемость переходного процесса.

При синтезе ПИД регуляторов применялось несколько оптимизационных методов, т.к. в зависимости от конкретной модели контура регулирования методы могут выдавать разные результаты. Предугадать какой метод будет самым оптимальным не представляется возможным. Поэтому при синтезе нужно использовать несколько оптимизационных методов.

Применение программного пакета Matlab Simulink для оптимизации позволяет: не обладая знаниями специального математического аппарата и языка программирования Matlab, провести расчеты пользуясь только графическим интерфейсом Matlab. Позволяет расширить недостающий функционал просто добавлением нужного скрипта в специально отведенные для этого места. Сгенерировать исполняемый код на языке C и поместить его сразу на контроллер.

По результатам оптимизации получили: графики передаточных функций от времени при единичном воздействии и функций интегральных критериев от времени, величины интегрального критерия для каждого метода.