Обработка сигналов. Лекция 3.

Алексей Орлов

19 февраля 2021 г.

Содержание

1 Свойства линейных стационарных систем

Поскольку линейные стационарные системы описываются свёрткой, свойства этого класса систем определяются свойствами дискретной свертки. Следовательно, импульсная характеристика конкретной системы содержит всю полноту информации.

Некоторые общие свойства класса линейных стационарных систем можно обнаружить, изучая свойства операции свертки. Например, свертка – коммутативная операция:

$$x[n] * h[n] = h[n] * x[n]. \tag{1}$$

Таким образом, порядок последовательностей в свертке несущественен, т. е. реакция системы не изменится, если поменять местами входную последовательность и отклик системы на еди- ничный импульс. Иными словами,

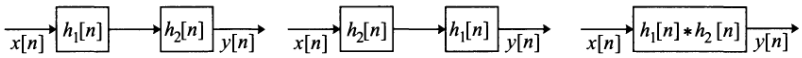

Рис. 1: Три линейных стационарных системы с одной и той же импульсной характеристикой.

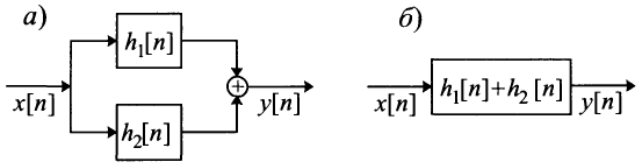

Рис. 2: а) параллельное соединение линейных стационарных систем; б) система, эквивалентная системе а).

отклик системы на сигнал $x[n]$ при импульсной характеристике $h[n]$ будет тем же, что и реакция на сигнал $h[n]$ при импульсной характеристике $x[n]$.

Свёртка удовлетворяет свойству дистрибутивности относительно сложения, а именно

$$x[n] * (h_1[n] + h_2[n]) = x[n] * h_1[n] + x[n] * h_2[n] \tag{2}$$

Это свойство непосредственно вытекает из формулы суперпозиции и является прямым следствием линейности и коммутативности свертки.

При *каскадном соединении* (последовательном) систем реакция первой системы подается на вход второй, отклик второй — на вход третьей и т. д. Отклик последней системы служит реакцией всей цепочки систем. Две линейные стационарные системы, подключенные последовательно, образуют одну линейную стационарную систему, чья импульсная характеристика совпадает со свёрткой импульсных характеристик обеих систем, что иллюстрирует рис. 1. На первом блоке диаграмм рисунка реакция первой системы на $x[n] = \delta[n]$ равна $h_1[n]$. Поэтому выходная последовательность из второй системы (и, по определению, импульсная характеристика каскада систем) должна быть равна

$$h[n] = h_1[n] * h_2[n]. \tag{3}$$

Как следствие коммутативности свертки импульсная характеристика каскада линейных стационарных систем не зависит от порядка, в котором они подключаются друг к другу. Этот факт отражен на рис. 1, где три системы имеют одну и ту же импульсную характеристику.

При параллельном соединении системы имеют общий вход, а их выходные последовательности складываются и дают реакцию всего соединения. Как следует из дистрибутивности свёртки, параллельное соединение двух линейных стационарных систем можно заменить одной линейной стацио-

нарной системой, чья импульсная характеристика равна сумме характеристик компонент соединения (рис. 2), т. е.

$$h[n] = h_1[n] + h_2[n]. \tag{4}$$

Требования линейности и стационарности выделяют класс систем с весьма специфическими свойствами. Устойчивость и детерминированность предоставляют вспомогательные свойства, и часто бывает важно знать, является ли данная линейная стационарная система устойчивой или детерминированной. Напомним, что устойчивой системой называют систему с ограниченной реакцией на каждый ограниченный входной сигнал. Линейная стационарная система является устойчивой тогда и только тогда, когда её импульсная характеристика — абсолютно суммируемая последовательность, т.е. если

$$S = \sum_{k=-\infty}^{\infty} |h[k]| < \infty. \tag{5}$$

Это является достаточным и необходимым условием устойчивости.

Детерминированная система была определена нами в предыдущей лекции как система, отсчёт $y[n_0]$ реакции которой зависит только от отсчетов $x[n]$ сигнала с $n \leqslant n_0$. Ввиду формулы суперпозиции и из $y[n] = h[n] * x[n]$ это определение влечет утверждение: импульсная характеристика детерминированной линейной стационарной системы должна удовлетворять условию

$$h[n] = 0, \quad n < 0. \tag{6}$$

По этой причине последовательность с нулевыми членами $h[n]$ при $n < 0$ удобно иногда называть детерминированной, подразумевая, что она является откликом на единичный импульс детерминированной линейной стационарной системы.

Хотя для нелинейных или нестационарных систем импульсная характеристика тоже может быть вычислена, она представляет лишь ограниченный интерес, поскольку формула свертки, а также условия суперпозиции линейной системы и (5)), обеспечивающие устойчивость и детерминированность, не применимы к таким системам.

Для иллюстрации свойств линейной стационарной системы в ответе на единичный импульс вновь обратимся к рассмотренным ранее примерам таких систем. Найдём сначала реакцию систем на единичный импульс. Сделать это можно, опираясь непосредственно на формулы, задающие соответствующие системы.

Идеальная система задержки:

$$h[n] = \delta[n - n_d], \tag{7}$$

где n_d - фикисированное натуральное число.

Скользящее среднее:

$$h[n] = \frac{1}{M_1 + M_2 + 1} \sum_{k=-M_1}^{M_2} \delta[n-k] = \begin{cases} \dfrac{1}{M_1 + M_2 + 1}, & M - 1 \leqslant n \leqslant M_2, \\ 0, & \text{в других случаях.} \end{cases} \tag{8}$$

Сумматор:

$$h[n] = \sum_{k=-\infty}^{n} \delta[k] = \begin{cases} 1, & n \geqslant 0, \\ 0, & n < 0 \end{cases} = u[n]. \tag{9}$$

Правая разностная система:

$$h[n] = \delta[n+1] - \delta[n]. \tag{10}$$

Левая разностная система:

$$h[n] = \delta[n] - \delta[n-1]. \tag{11}$$

Зная импульсные характеристики стационарных систем, мы можем проверить системы на устойчивость, вычисляя сумму S.

Ясно, что для систем идеальной задержки, скользящего среднего, правой и левой разностных систем указанная сумма меньше бесконечности, поскольку в этих случаях импульсная характеристика имеет лишь конечное число ненулевых отсчетов. Такие системы называют *системами с конечной импульсной характеристикой* (КИХ-системами). Ясно, что КИХ-системы будут устойчивыми тогда и только тогда, когда их отклики на единичный импульс конечны по абсолютной величине. С другой стороны, сумматор не является устойчивой системой, так как

$$S = \sum_{n=0}^{n} u[n] = \infty. \tag{12}$$

Ранее при проверки устойчивости мы уже доказали неустойчивость сумматора, предъявив пример ограниченного сигнала (единичный скачок), реакция на который неограничена.

Импульсная характеристика сумматора неограничена по длительности. Сумматор – пример класса систем, которые называют системами с *бесконечной импульсной характеристикой* (БИХ-системами). Примером устойчивой БИХ-системы служит такая система, импульсная характеристика которой имеет вид $h[n] = a^n u[n]$ с $|a| < 1$. В этом случае

$$S = \sum_{n=0}^{\infty} |a|^n. \tag{13}$$

Если $|a| < 1$, то формула суммы членов бесконечной геометрической прогрессии даёт

$$S = \frac{1}{1 - |a|} < \infty \tag{14}$$

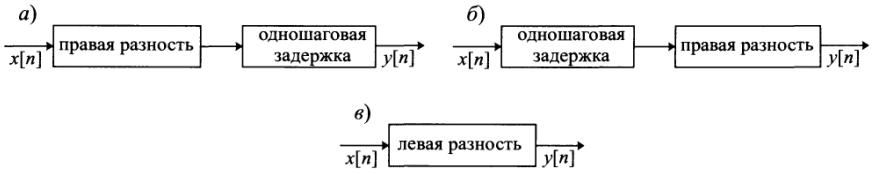

Рис. 3: Эквивалентные системы, получающиеся друг из друга благодаря коммутативности свертки.

С другой стороны, если $|a| \geqslant 1$, то сумма соответствующего ряда бесконечна и система неустойчива.

Для проверки детерминированности линейных стационарных систем нам нужно убедиться, что $h[n] = 0$ при $n < 0$. Как было выяснено, ИСЗ детерминирована при $n_d \geqslant 0$. Если $n_d < 0$, эта система недетерминирована. В случае скользящего среднего свойство детерминированности накладывает ограничения: $-M_1 \geqslant 0$ и $M_2 \geqslant 0$. Сумматор и левая разностная система являются детерминированными системами, а правая разностная система недетерминирована.

Концепция свертки как операции над двумя последовательностями подводит нас к упрощению многих задач, связанных с системами. Особенно полезный результат можно сформулировать относительно ИСЗ. Поскольку отклик этой системы имеет вид $y[n] = x[n - n_d]$, а её импульсная характеристика равна $h[n] = \delta[n - n_d]$, то

$$x[n] * \delta[n - n_d] = \delta[n - n_d] * x[n] = x[n - n_d]. \tag{15}$$

Иными словами, свертка сдвинутого на n_d позиций единичного импульса с сигналом $x[n]$ приводит к задержке сигнала на то же число n_d.

Так как задержка является одной из основных операций при реализации линейных систем, предыдущий результат полезен при анализе и упрощении комплексов линейных стационарных систем. В качестве примера рассмотрим систему, изображенную на рис. 3(а), которая состоит из правой разностной системы, последовательно соединенной с одношаговой задержкой.

Ввиду коммутативности свёртки порядок подключения систем в последовательной цепи несущественен, поскольку они линейны и стационарны. Следовательно, поменяв местами одношаговую задержку и правую разностную систему (рис. 3(а)), мы получим ту же результирующую систему (рис. 3(б)). Кроме того, формула (3) влечет, что импульсная характеристика всей цепи систем совпадает со свёрткой импульсных характеристик каждой из составных частей. Поэтому

$$h[n] = (\delta[n + 1] - \delta[n]) * \delta[n - 1] = \delta[n - 1] * (\delta[n + 1] - \delta[n]) \tag{16}$$

$$= \delta[n] - \delta[n - 1]. \tag{17}$$

Мы видим, что $h[n]$ тождественна импульсной характеристике левой разностной системы, т.е. цепи систем, изображенные на рис. 2.13, а) и б), можно заменить левой разностной системой, как показано на рис. 3(в).

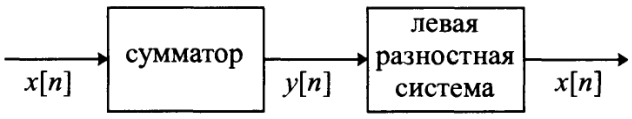

Рис. 4: Каскад сумматора и левой разностной системы.

Заметим, что недетерминированная правая разностная система, будучи последовательно соединена с задержкой (рис. 3(а) и (б)), превращается в детерминированную систему. В общем случае недетерминированная КИХ-система может быть преобразована в детерминированную систему, если к ней присоединить достаточно большую задержку.

Другой пример каскада систем вводит понятие инверсивной системы. Рассмотрим каскад систем (рис. 4). Его импульсная характеристика вычисляется по правилу:

$$h[n] = u[n] * (\delta[n] - \delta[n-1]) = u[n] - u[n-1] = \delta[n]. \tag{18}$$

То есть каскадная комбинация сумматора с левой разностной системой (в любом порядке) дает систему, чья импульсная характеристика совпадает с единичным импульсом. Таким образом, выход этой каскадной комбинации будет всегда совпадать со входом, поскольку $x[n]*\delta[n] = x[n]$. В этом случае действие левой разностной системы полностью компенсируется (или инвертируется) действием сумматора. При этом говорят, что левая разностная система — инверсивная система к сумматору.

Благодаря коммутативности свёртки сумматор служит инверсивной системой для левой разностной системы. В общем случае, если импульсная характеристика линейной стационарной системы – $h[n]$, то её инверсивная система (если она существует) реагирует на единичный импульс последовательностью $h_i[n]$, определенной соотношением

$$h[n] * h_i[n] = h_i[n] * h[n] = \delta[n]. \tag{19}$$

Инверсивные системы полезны в ситуациях, когда необходимо компенсировать эффект линейной системы. В общем случае разрешить уравнение (19) относительно $h_i[n]$ очень трудно. Однако в следующих лекциях мы увидим, что z-преобразование даёт прямой метод построения инверсивной системы.

2 Средства Simulink

Компонент Simulink является подсистемой (расширением) MATLAB, предназначенной для блочного моделирования, однако, благодаря своим уникальным возможностям, этот компонент часто воспринимают как самостоятельную систему и называют ядром Simulink. Слово “simulink” образовано из комбинации первых четырех букв слова “simulation” (моделирование) и

“link” (соединение). Simulink автоматически установится при инсталляции пакета MATLAB.

Моделирование в Simulink выполняется средствами блочного моделирования. Средства Simulink созданы на основе программных средств MATLAB, но позволяют исключить или минимизировать использование языка MATLAB в явном виде, что существенно упрощает технологию моделирования. Специфика моделирования в Simulink определяется его основным предназначением — моделирование динамических систем (Dynamic Systems). Технология моделирования в Simulink заключается в построении модели системы из стандартных блоков и позволяет следить за ее работой с помощью стандартных средств наблюдения, поэтому моделирование в Simulink часто называют «визуальным». В большинстве приложений динамическую систему можно фактически или условно представить как систему обработки сигналов, полагая, что понятия «сигнал» и «система» имеют широкое толкование, а именно:

- под сигналом понимается воздействие любой физической природы или последовательность данных;
- под системой понимается физическое устройство или математическое преобразование, выполняющее требуемое преобразование воздействия.

Компьютерная модель динамической системы строится на основе математических моделей сигналов и систем.

Компьютерную модель системы, созданную средствами Simulink, будем называть S-моделью системы — сокращение от Simulink-модель. В русскоязычной литературе в теории аналоговых, дискретных и цифровых систем под «моделированием системы» по умолчанию подразумевают моделирование процесса её работы.

В справочной системе MATLAB по Simulink понятие «моделирование системы» (Modeling a System) относится скорее к созданию S-модели системы (т. е. к описанию статической системы средствами Simulink), а для моделирования процесса работы системы добавляют уточняющий термин — «моделирование динамической системы» (Modeling a Dynamic System).

Во избежание путаницы будем пользоваться русскоязычной терминологией, и в технологии моделирования системы в Simulink выделим две части:

1. Создание S-модели системы
2. Моделирование системы

Для создания S-моделей линейных систем используется математическая модель $y[n] = T\{x[n]\}$, в которой параметры оператора T однозначно определяются *передаточной функцией* (с ней мы познакомимся далее по курсу), а математическая модель системы принимает вид линейного уравнения: *дифференциального* для аналоговых и *разностного* – для дискретных систем.

В математической модели линейной системы мы выделяем три основные составляющие:

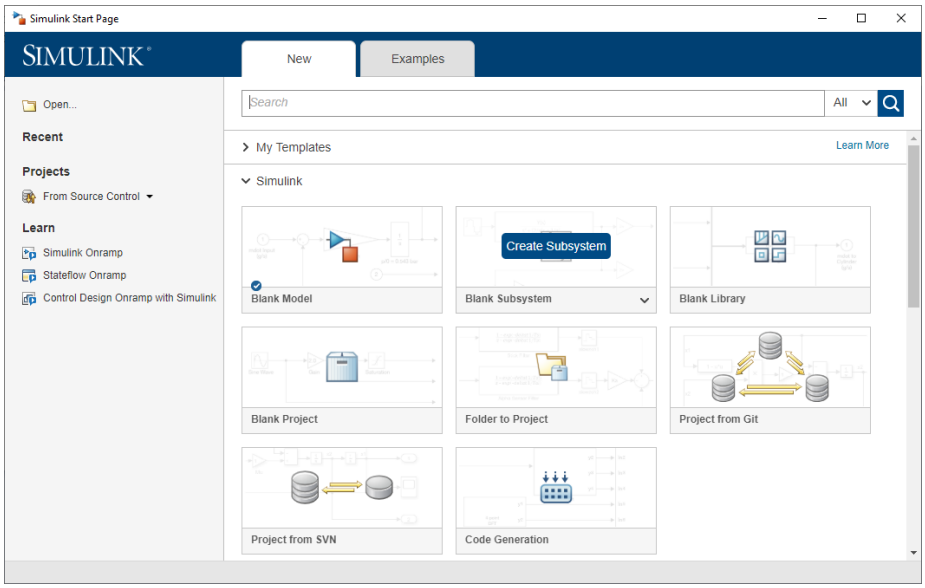

Рис. 5: Стартовая страница Simulink

1. входной сигнал X
2. структура линейной системы, определяемая видом ее передаточной функции
3. выходной сигнал Y

В S-модели линейной системы им будут соответствовать три отличающиеся своим функциональным назначением компоненты:

1. S-модель входного сигнала X
2. S-модель структуры линейной системы
3. S-модель средств анализа выходного сигнала Y

3 Основные этапы создания S-модели системы

Для запуска Simulink необходимо нажать кнопку Simulink в верхней панели инструментов либо же выполнить команду simulink в командном окне MATLAB. После этого откроется окно Simulink Start Page (рис. 5). Здесь мы можем создать новый пустой проект или пролистать вниз и выбрать шаблон из каталога библиотеки Simulink.

В общем случае технология создания S-модели системы включает в себя следующие основные этапы:

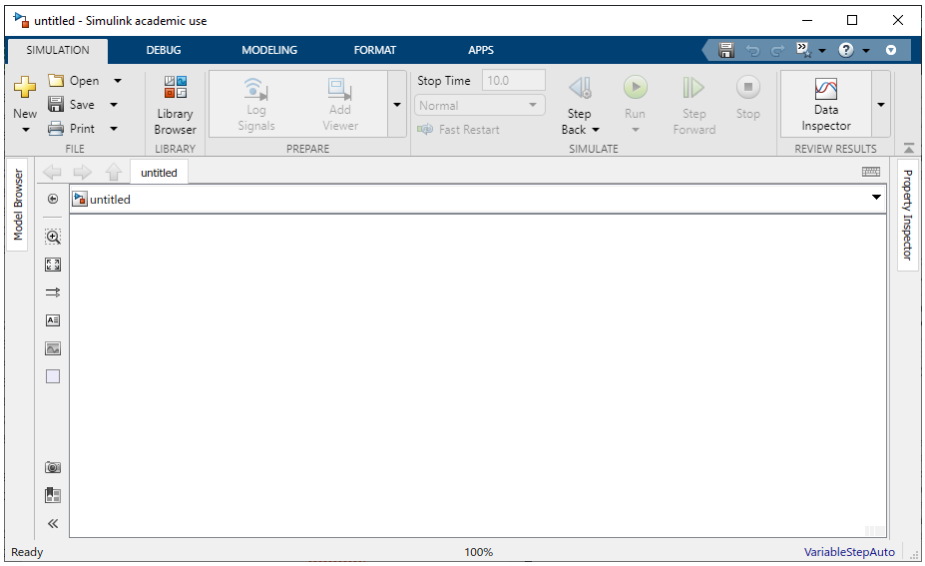

Рис. 6: Окно создания S-модели.

1. Открытие окна S-моделей

По нажатию кнопки “Create Model”, появляющейся при наведении на область “Blank Model” окна Simulink Start Page (см. рис. 5), открывается окно S-моделей untitled (безымянное) для создания новой S-модели системы (рис.6).

В строке состояния окна S-моделей (нижняя строка) отображаются (слева направо):

- стадия моделирования — Ready (Завершено) или Running (Выполнение), а в статическом состоянии при обращении к командам меню — краткая информация о выбранной команде
- масштаб S-модели, по умолчанию — 100%
- индикатор процесса моделирования (активен при запуске модели)
- индикатор текущего времени моделирования (активен при запуске модели)
- выбранный решатель (на рисунке 6 это VariableStepAuto)

2. Перетаскивание блоков (компонентов S-модели системы) из библиотеки Simulink.

Открыть библиотеку можно нажатием кнопки Library Browser в панели инструментов окна Simulink.

В окне “untitled” новая S-модель системы создается из блоков библиотеки Simulink с помощью технологии drag-and-drop («перетащить и отпустить»). Она заключается в последовательном перетаскивании требуемых

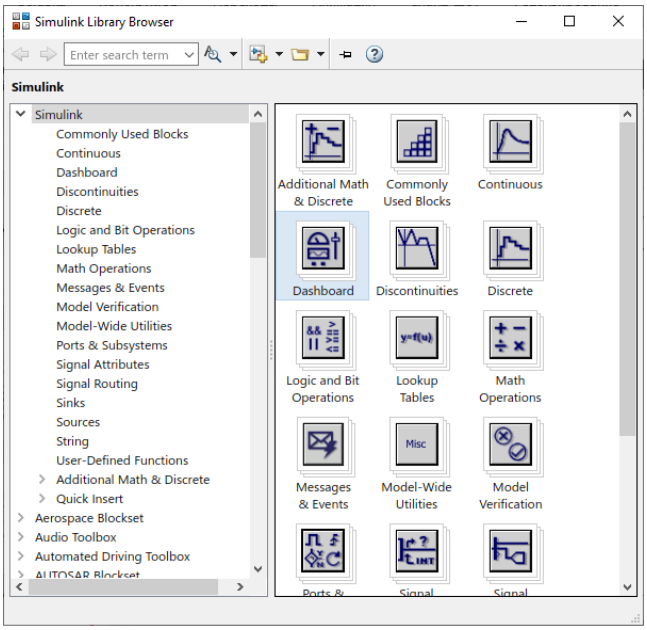

Рис. 7: Библиотека компонентов Simulink.

блоков в окно S-моделей “untitled” при нажатой левой кнопке мыши с правой панели окна Simulink Library Browser.

Перетащим в окно “untitled” блоки, которые нам потребуются для создания S-модели простой *линейной дискретной* системы. Для примера выберем систему задержки Delay, и в качестве S-модели возьмём блок Delay из группы блоков Discrete.

В качестве *S-модели входного сигнала* выберем блок Step из группы блоков Sources, генерирующий *цифровой единичный скачок* для дискретной системы. В качестве S-моделей средств анализа входного и выходного сигналов выберем блок Scope (осциллограф) из группы блоков Sinks.

3. Задание параметров блоков.

Параметры блока, определяющие вид сигнала на его выходе или настройку средства анализа сигнала, задаются в окне Block Parameters (Параметры блока), которое открывается двойным щелчком левой кнопки мыши на пиктограмме блока.

Изменим один из параметров блока Step для S-модели дискретной системы: выставим период дискретизации Sample Time: 0.2. Параметры блока Delay зададим так, чтобы получить единичную задержку: параметр Delay Length установим равным 1. Отметим тут, что период дискретизации Sample Time, который имеет значение –1 в параметрах блока Delay, соответствует наследованию периода дискретизации от блока Step.

Для блока Scope по двойному щелчку вместо окна Block Parameters от-

крывается окно осциллографа. В нём нажмём на пиктограмму шестерни, открыв таким образом Configuration Properties. Далее, установим следующие параметры:

- На вкладке Main. Number of axis (Количество осей) равным 2 для отображения входного и выходного сигналов на двух независимых графиках;
- На вкладке Display. Y-limits (Minimum) и Y-limits (Maximum) (Границы осей ординат) равными –1 и 2 соответственно. Обратите внимание, что эти значения нужно задать для двух дисплеев, переключение между которыми осуществляется выбором Active Display.

4. Соединение блоков.

Соединение блоков можно выполнять вручную или автоматически.

При ручном соединении блоков нужно подвести курсор к выходу соединяемого блока, и после того как появится крестик, фиксирующий местоположение выхода, при нажатой левой кнопке мыши провести прямую линию к входу соответствующего блока. При проведении ломаной линии кнопку мыши в точках изгиба нужно отпускать. При нажатой кнопке мыши местоположение входа (или узла) автоматически фиксируется сдвоенным крестиком.

При автоматическом соединении блоков выделите блок (щелчком левой кнопки мыши), содержащий вход, и при нажатой клавише <Ctrl> выделите блок, содержащий выход, после чего автоматически будет выполнено соединение блоков.

Сначала соединим источник сигнала с системой единичной задержки, которую далее подключим к первому порту осциллографа. Далее, нам необходимо подвести к осциллографу также и входной сигнал, чтобы он отображал его вместе с выходом системы задержки Для этого нажмём на второй порт осциллографа и с зажатой левой кнопкой мыши проведём путь подключения до связи осциллографа с единичной задержкой. Таким образом мы создадим узел разветвления.

5. Сохранение содержимого окна S-моделей

Сохранение содержимого окна S-моделей в файл выполняется по команде меню File | Save (Файл | Сохранить). В открывающемся окне Save as (Сохранить как) файлу S-моделей присваивается имя с расширением slx.

Имя файла может состоять из любой последовательности латинских букв, цифр и символа подчеркивания, начинающейся с буквы. Файлы S-моделей с расширением slx должны храниться в текущей рабочей папке или в папке пользователя, вложенной в неё.

Создание собственной папки в папке work выполняется с помощью контекстного меню в окне Current Folder окна MATLAB, а сохранение пути к ней – по команде контекстного меню Add Path (Добавить путь) или команде главного меню File | Set Path (Файл | Установить путь).

Нашему файлу, хранящему S-модель линейной дискретной системы с соединением блоков, присвоим имя `discrete_delay`.

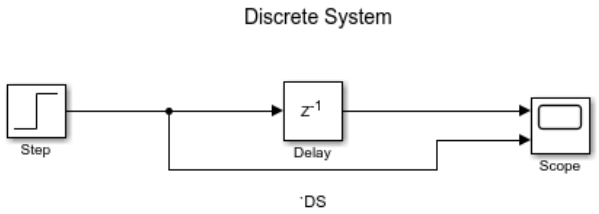

Рис. 8: Конфигурация S-модели после редактирования.

3.1 Операции по оформлению S-моделей системы

Операции по оформлению S-моделей системы можно объединить в следующие три группы:

1. Операции с текстом.

Редактирование текста, в том числе имени блока, выполняется после его активации щелчком левой кнопки мыши в поле текста. Создание текстового поля осуществляется нажатием кнопки Annotation из палитры инструментов. Редактирование текстов в соответствующем поле ввода выполняется теми же средствами, что и в Word. Для выхода из режима редактирования текста следует щёлкнуть левой кнопкой мыши на свободном поле окна S-моделей.

На рисунке 8 изображена S-модель системы, где выполнено следующее редактирование: перемещены блоки; включено отображение имён блоков (Format -> Show Block Name -> On); добавлены заголовки; созданы метки входных сигналов.

2. Операции с цветом

Включают в себя выбор цвета линий для блока, выбор цвета фона блока, выбор цвета фона окна S-моделей, выделение цветом всех блоков (и связанных с ними соединений), у которых параметр Sample Time больше нуля.

3. Операции по выводу дополнительной информации

Включают в себя:

- _Вывод типа данных. Активируется на вкладке Debug верней панели инструментов. В Information Overlays необходимо выбрать Base Data Types. Тип данных будет выведен на выходе блоков.
- _Вывод размерности данных. Операция доступна для сигналов в виде данных — векторов и матриц.
- _Вывод порядка исполнения блоков. Также активируется на вкладке Debug, где неоходимо выбрать Execution Order из меню Information Overlays.

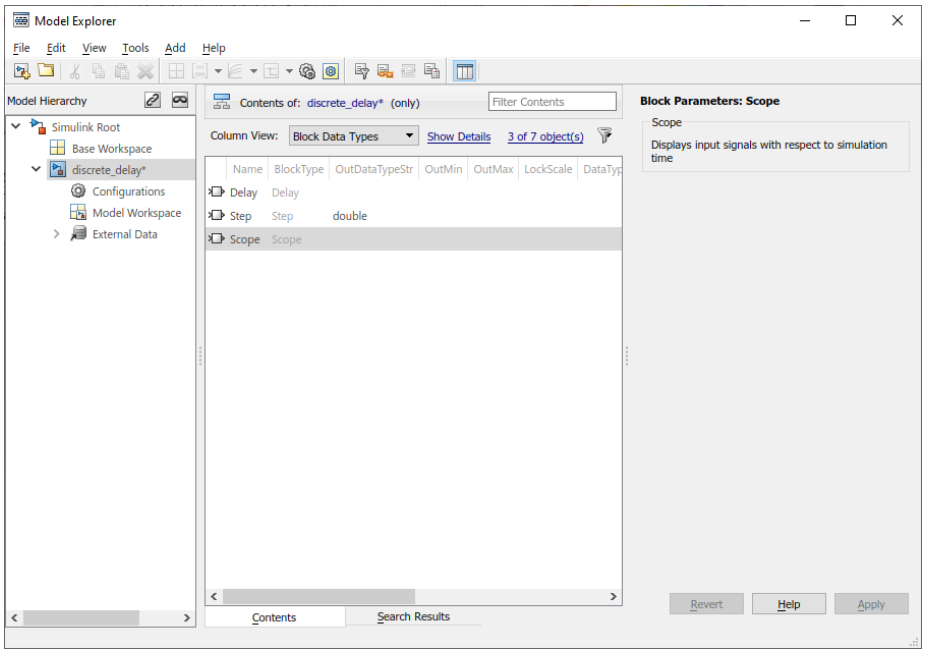

Рис. 9: Окно исследования S-моделей.

Сообщение об ошибке при создании S-модели системы выводится в автоматически открывающемся окне с комментариями об ошибке и одновременно отображается визуально в окне S-моделей.

3.2 Исследование S-модели

Окно S-моделей предоставляет удобные средства для исследования S-модели системы, в том числе средства оперативного просмотра и редактирования параметров блоков и параметров настройки.

Исследование S-модели выполняется с помощью вызова окна Model Explorer из вкладки верхней панели инструментов Modeling. К окну Model Explorer можно также обратиться по команде контекстного меню Explore, выделив в S-модели интересующий блок.

Окно Model Explorer содержит три панели:

- Model Hierarchy (Иерархия модели) — корневой каталог (Simulink Root), представленный в виде дерева;
- Contents of (Содержимое) — содержимое выделенного на панели Model Hierarchy раздела;
- Model Properties (Окно свойств объекта) — информация о разделе, выделенном на панели Model Hierarchy или Contents of.

4 Технология моделирования системы

После создания S-модели системы приступают к моделированию системы, основные этапы технологии которого включают в себя:

- настройку S-модели системы;
- настройку обмена данными S-модели системы с рабочим пространством памяти Workspace;
- настройку диагностики моделирования;
- моделирование системы — запуск, пауза и останов.

Стоит также отметить, что моделирование сложных систем предполагает их предварительную отладку.

4.1 Настройка S-модели системы

Для знакомства с настройкой S-модели системы откроем безымянное окно "untitled" пустой модели и далее, из вкладки верхней панели инструментов Simulation | Prepare, выберем Model Settings (Моделирование | Подготовка | Настройка модели), — откроется окно Configuration Parameters (Параметры конфигурации).

Компоненты настройки S-модели системы в окне Configuration Parameters (рис. 10 представлены на левой панели в виде дерева.

Настройка S-модели системы заключается в задании параметров моделирования в компоненте Solver (Решатель), выделенной на левой панели Select по умолчанию.

Параметры моделирования задаются на правой панели:

1. В группе Simulation time (Время моделирования) указывается интервал моделирования, задаваемый его границами:

- в поле ввода Start time (Начальное время) — начальное время моделирования в секундах, по умолчанию равное нулю;
- в поле ввода Stop time (Конечное время) — конечное время моделирования в секундах

Конечное время выбирается, исходя из конкретной задачи, по умолчанию оно равно 10 с.

Если конечное время заранее неизвестно, то можно установить значение `inf` (машинную бесконечность), а затем, после запуска моделирования, остановить процесс моделирования или сделать в нем паузу.

Интервал моделирования не отражает реального времени и зависит от многих факторов: сложности S-модели, шага моделирования, быстродействия компьютера и др.

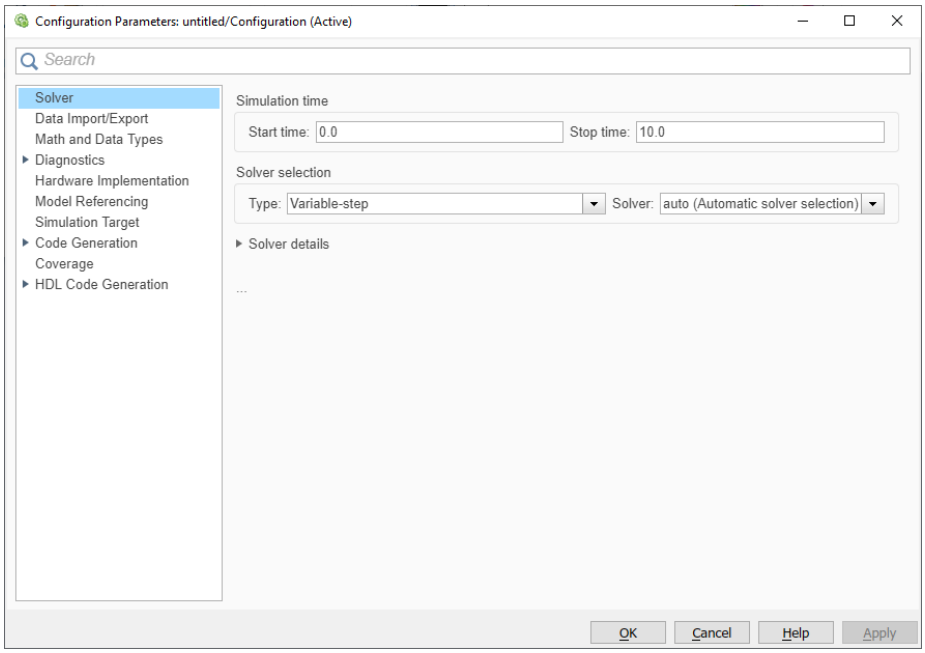

Рис. 10: Окно настройки моделирования.

2. В группе Solver selection (Выбор решателя) выбирается тип решателя. По умолчанию здесь будет задан переменный адаптивный шаг (Variable-step) с автоматическим выбором конретного метода решения. Использование переменного шага позволяет подстраивать точность вычислений под масштаб входного сигнала: увеличивать её там, где сигнал начинает быстро варьироваться, и увеличивать размер шага там, где сигнал меняется слабо, чтобы исключить лишние вычислительные итерации.

При моделировании аналоговых систем, соотношение вход/выход которых описывается обыкновенным дифференциальным уравнением (ОДУ), решателем (Solver) называют функцию MATLAB, реализующую метод численного интегрирования ОДУ.

При моделировании дискретных систем, соотношение вход/выход которых описывается разностным уравнением, непосредственно описывающим алгоритм вычисления реакции системы, решателем называют средства Simulink (скрытые от пользователя), реализующие данный алгоритм.

4.2 Настройка обмена данными

Настройка обмена данными S-модели системы с рабочим пространством памяти Workspace заключается в задании параметров взаимодействия с Workspace с целью экспорта/импорта данных в процессе моделирования и производится при обращении к компоненте Data Import/Export (Импорт

/ Экспорт данных) на левой панели Select окна Configuration Parameters.

Параметры взаимодействия с Workspace задаются на правой панели

В группе Load from workspace (Загрузка из Workspace) выбираются параметры, контролирующие импорт из Workspace перед моделированием:

- Input (Входные данные) — флаг, управляющий импортом данных из Workspace.

  При установке флага Input в поле ввода указываются имена импортируемых из Workspace входных данных, которые могут быть представлены в следующих видах:

  - матрица $[t, u]$ (по умолчанию), где t — вектор-столбец возрастающих значений времени, а u — матрица, векторы-столбцы которой соответствуют значениям сигналов: i-й столбец соответствует значениям i-го сигнала в моменты времени, заданные вектором времени t;
  - структура (массив записей), заданная своим именем;
  - вектор времени t;
  - выражение MATLAB в апострофах, представляющее собой функцию времени, например `'exp(t)+3'`, где аргументу t в процессе моделирования будут автоматически присваиваться значения времени.

  Данные, указанные в поле ввода Input, импортируются из Workspace с помощью блока In.

  Импорт данных может также осуществляться с помощью блока From Workspace или блока Signal From Workspace. В этом случае настройка обмена с Workspace выполняется посредством установки параметров блока;

- Initial state (Начальное состояние) — флаг, управляющий импортом начального состояния (начальных условий) из Workspace.

  При установке флага начальное состояние импортируется из Workspace также с помощью блока In.

В группе Save to workspace (Сохранение в Workspace) выбираются параметры, управляющие экспортом данных в Workspace после моделирования:

- Time (Время) — флаг, управляющий экспортом в Workspace вектора значений времени.

  При установке флага Time (по умолчанию) в поле ввода указывается имя вектора (по умолчанию tout) с экспортируемыми значениями времени;

- States (Переменные состояния) — флаг, управляющий экспортом переменных состояний.

При установке флага States в поле ввода указывается имя переменной (по умолчанию xout) для экспортируемых в Workspace переменных состояний S-модели системы.

Вид представления экспортируемых переменных состояний указывается в выпадающем списке Format;

- Output (Выходные данные) — флаг, управляющий экспортом выходных данных в Workspace.

  При установке флага Output (по умолчанию) в поле ввода указывается имя переменной для экспортируемых в Workspace данных (по умолчанию yout).

  Данные, указанные в поле ввода Output, экспортируются в Workspace с помощью блока Out. Вид представления экспортируемых в Workspace данных указывается в поле Format.

  Экспорт данных может также осуществляться с помощью блока To Workspace или блока Signal To Workspace. В этом случае настройка обмена с Workspace (экспорт) выполняется посредством установки параметров блока;

- Final states (Конечные состояния) — флаг, управляющий экспортом в Workspace переменных состояний на последнем шаге моделирования.

  При установке флага Final states в поле ввода указывается имя переменной для экспортируемых в Workspace конечных состояний (по умолчанию xFinal) и активизируется флаг Save final operating point (Сохранить состояние моделирования в последнем положении), управляющий сохранением текущего состояния моделирования;

- Signal logging (Регистрация сигнала) — флаг, управляющий регистрацией сигнала в Workspace в процессе моделирования.

  При установке флага (по умолчанию) в поле ввода Signal logging указывается имя регистрируемого в Workspace сигнала (по умолчанию logsout).

Более подробную информацию можно получить с помощью справочной системы MATLAB по Simulink.

4.3 Моделирование системы — запуск, пауза и останов

Запуск моделирования системы осуществляется нажатием кнопки Run (Запуск моделирования) на вкладке Simulation панели инструментов окна S-моделей.

Пауза в процессе моделирования системы осуществляется нажатием кнопки Pause (Пауза в моделировании) на панели инструментов, а продолжение процесса моделирования — нажатием на кнопку Continue (Продолжение моделирования) на панели инструментов или по команде меню Simulation | Continue.

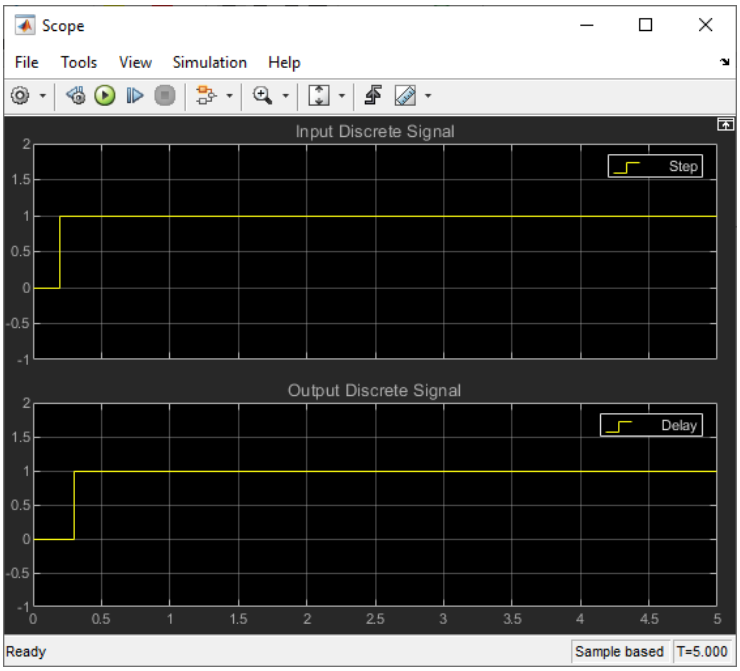

Рис. 11: Результаты моделирования системы единичной задержки.

Останов (завершение) моделирования системы осуществляется нажатием кнопки Stop (Останов моделирования) на панели инструментов.

4.4 Моделирование системы единичной задержки

Выполним моделирование простой линейной дискретной системы, созданной ранее и сохраненной в файле S-модели `discrete_delay.sxl`.

Выполним настройку S-модели дискретной системы:

- в поле ввода Stop time окна Configuration Parameters установим конечное время моделирования 5.0;
- в поле ввода Solver выберем решатель discrete (no continues states).

Остальные параметры оставим по умолчанию.

Запустим процесс моделирования, результатами которого будут цифровой единичный скачок (воздействие) и его задержанный вариант (реакция), представленные на рис. 11.

Таким образом, мы смоделировали работу важного звена в цифровой обработке сигналов – единичной задержки. Моделирование произвольных систем в Simulink осуществляется посредством задания передаточной функции системы, что будет рассмотрено далее по курсу.

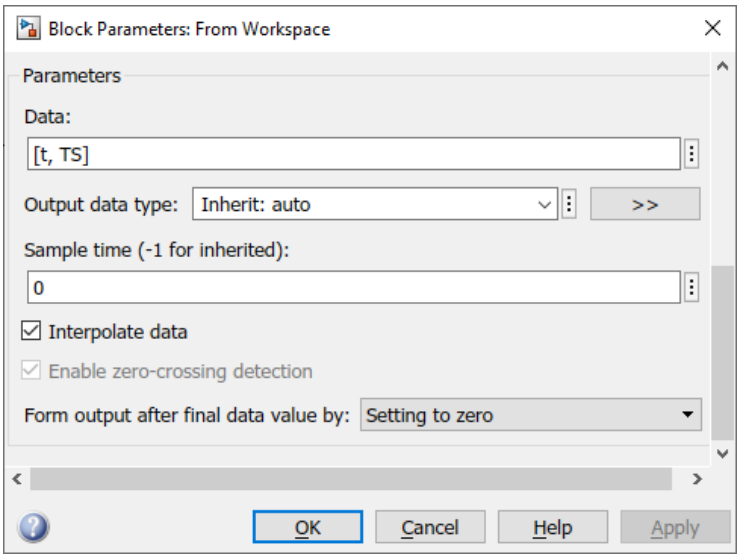

Рис. 12: Окно настройки источника сигнала From Workspace.

4.5 Моделирование системы с фильтром скользящего среднего

Уже сейчас, однако, мы имеем возможность смоделировать систему с фильтром входящего сигнала. В его роли будет выступать блок, выполняющий операцию скользящего усреднения. В прошлой лекции мы выполняли данную операцию в приложении MATLAB Signal Analyzer. Здесь же мы смоделируем цифровую систему обработки сигналов, выполняющую эту операцию.

Прежде всего, нам необходимо научиться загружать в Simulink данные из рабочего пространства MATLAB. Сделать это можно при помощи блока From Workspace из категории Sources библиотеки Simulink. В качестве данных будет выступать переменная TS, созданная в прошлой лекции, в которую записаны наблюдения за температурой в Санкт-Петербурге.

Блок From Workspace импортирует сигналы из рабочей области памяти Workspace и имеет следующие параметры (рис. 12):

- Data (Данные) — имя матрицы (по умолчанию) или структуры (массива записей), импортируемой из Workspace и хранящей значения сигнала (сигналов).

  Импортируемая из Workspace матрица должна быть представлена в виде: `[T U]`, где `T` — столбец возрастающих значений времени; `U` — матрица, столбцы которой соответствуют значениям сигналов в моменты времени, заданные вектором `T`.

  В нашем случае имеем переменную TS, хранящую значения сигнала.

Осталось только добавить временные отсчёты. Для этого создадим переменную `t`, которая будет их откладывать, начиная с нуля, с шагом 0.1 с.:

```
t = [0:0.1:(length(TS)-1)*0.1]';
```

Обратите внимание, было осуществлено транспонирование вектор-строки. Сделано это ввиду того, что `T` должен быть именно столбцом возрастающих значений времени, а не строкой.

Таким образом, в Workspace следует указать Data: [t TS].

Если используется структура, импортируемая из Workspace, то она должна содержать два поля:

- time — вектор-столбец возрастающих значений времени;
- signals — вложенная структура с двумя полями:
  - values — матрица, столбцы которой соответствуют значениям сигналов в моменты времени, заданные в поле time;
  - dimensions — количество сигналов (скаляр).

- Sample time – для источников следует оставить 0.

- Interpolate data (Интерполяция данных) — флаг интерполяции данных, управляющий согласованием интервала между соседними моментами времени и шагом моделирования.

  При установке флага (по умолчанию) незаданные значения сигнала в промежуточных точках вычисляются посредством линейной интерполяции между соседними точками. Если начальный момент времени больше начального времени моделирования, незаданные начальные значения сигнала вычисляются посредством линейной экстраполяции по первым двум точкам.

  При сбросе флага незаданные значения сигналов в промежуточных точках дублируют его предыдущее значение. Если начальный момент времени больше начального времени моделирования, незаданные начальные значения сигнала дублируют его первое значение;

- Enable zero crossing detection (Включить обнаружение пересечения нуля) — флаг контроля монотонности сигнала на каждом шаге моделирования.

  Установка флага (по умолчанию) позволяет определять моменты скачков сигнала (резкого изменения значения) и автоматически уменьшать шаг моделирования при выборе типа решателя с переменным шагом (Variable-step);

- Form output after final data value by (Формировать сигнал на выходе после последнего значения данных посредством) — определение метода вычисления незаданных конечных значений сигнала.

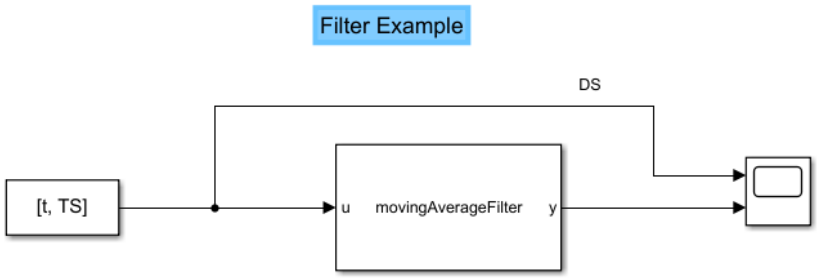

Рис. 13: S-модель системы с фильтром входного сигнала.

Если конечный момент времени меньше конечного времени моделирования, то незаданные конечные значения сигнала будут вычисляться по-разному, в зависимости значения этого параметра, а именно:

- Extrapolate (Экстраполяция) — посредством линейной экстраполяции;
- Setting to zero (Установка нуля) — добавлением нулей;
- Hold final data value (Удерживание последнего значения) — дублированием последних значений сигнала;
- Cyclic repetition (Циклическое повторение) — периодическим повторением значений сигнала.

Установленное по умолчанию значение Extrapolate нам нужно сменить на другое ввиду отсутствия данных для экстраполяции, что приведёт к ошибке при запуске. Установим здесь Setting to zero.

Теперь нам необходимо добавить, а точнее создать, блок, который будет являться фильтром и выполнять операцию скользящего усреднения входного сигнала.

Необходимый нам компонент системы находится в Library Browser под именем MATLAB System. Перетащите его в редактор S-модели системы.

Откройте редактор его свойств двойным щелчком на блоке. В окне свойств блока нажмите кнопку New. Откроется редактор нового файла с загруженным шаблоном системного объекта. Скопируйте туда листинг из приложения А. Сохраните документ под именем `movingAverageFilter.m` в рабочей директории. Вернитесь далее в окно свойств блока MATLAB System1 и откройте сохранённый файл.

У созданного нами блока усреднения имеется окно настройки, вызываемое двойным щелчком на пиктограмме блока, где можно указать значение окна усреднения. По умолчанию выставлено значение 5. Однако нам уже известно, что оптимальным значением для нашего входного метеорологического сигнала является 120.

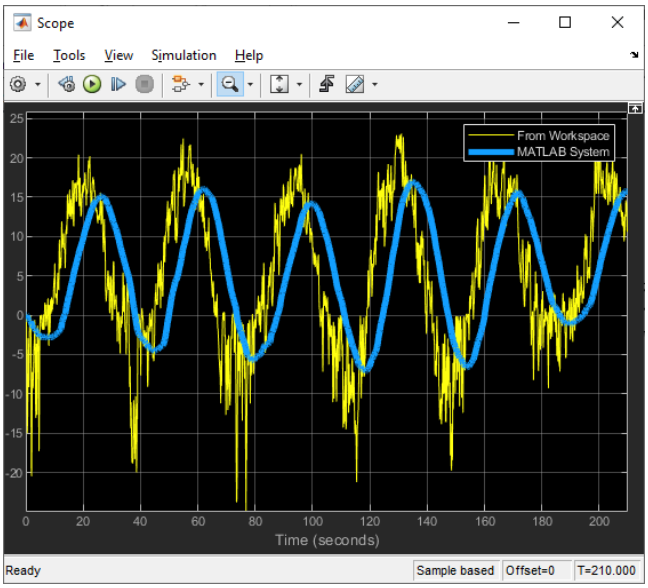

Рис. 14: Окно осциллографа, отображающее два сигнала, входной и после прохождения им фильтра.

Соединим все блоки и, как в предыдущем примере, создадим второй порт на осциллографе и подключим в него ответвление патча с входным сигналом. В итоге собранная нами система с фильтром скользящего среднего будет выглядеть следующим образом, представленным на рисунке 13.

Для запуска моделирования системы осталось только настроить параметры решателя и выставить время моделирования.

Зададим параметры моделирования в окне Configuration Properties: время моделирования (Stop time) - 210, тип (Type) - Fixed-Step, решатель (Solver) - discrete. Также зададим шаг, раскрыв группу Solver Details: значение Fixed-step size установим равным 0.1.

Теперь можно смело нажимать на кнопку Run. В итоге на осциллографе получаем картинку с исходным и отфильтрованным сигналами, что представлено на рисунке 14.

А ПРИЛОЖЕНИЕ

Листинг кода системного объекта MATLAB, выполняющего операцию скользящего усреднения.

`movingAverageFilter.m`

```
classdef movingAverageFilter < matlab.System
    % Unweighted moving average filter of 1- or 2D input.

    % Public, tunable properties
    properties(Nontunable)
        % WindowLength Moving window length
        WindowLength (1,1){mustBeInteger,mustBePositive}
            = 5
    end

    % Pre-computed constants
    properties(Access = private, Nontunable)
        pCoefficients;
    end

    properties(Access = private)
        State;
        pNumChannels = -1;
    end

    methods
        function obj = movingAverageFilter(varargin)
            % Support name-value pair arguments when
                constructing object
            setProperties(obj,nargin,varargin{:})
        end
    end

    methods(Access = protected)
        function setupImpl(obj,x)
            % Perform one-time calculations, such as
                computing constants
            obj.pNumChannels = size(x,2);
            obj.pCoefficients = ones(1,obj.WindowLength)/
                obj.WindowLength;
            obj.State = zeros(obj.WindowLength-1,obj.
                pNumChannels,'like',x);
        end

        function y = stepImpl(obj,u)
```

```
            % Implement algorithm. Calculate y as a
                function of input u and
            % states.
            [y,obj.State] = filter(obj.pCoefficients,1,u,
                obj.State);
        end

        function resetImpl(obj)
            % Initialize / reset discrete-state
                properties
            obj.State(:) = 0;
        end

        function validateInputsImpl(~, u)
            validateattributes(u,{'double','single'}, {'2
                d',...
                'nonsparse'},'','input');
        end

        function s = saveObjectImpl(obj)
            s = saveObjectImpl@matlab.System(obj);
            if isLocked(obj)
                s.pCoefficients = obj.pCoefficients;
                s.pNumChannels = obj.pNumChannels;
                s.State = obj.State;
            end
        end

        function loadObjectImpl(obj,s,wasLocked)
            if wasLocked
                obj.pCoefficients = s.pCoefficients;
                obj.pNumChannels = s.pNumChannels;
                obj.State = s.State;
            end
            loadObjectImpl@matlab.System(obj,s,wasLocked)
                ;
        end

    end
end
```