Introduction to Simulink and adaptive filtering

Signal Processing Systems Fall 2025

Lecture 12 (Thursday 4.12.)

Outline

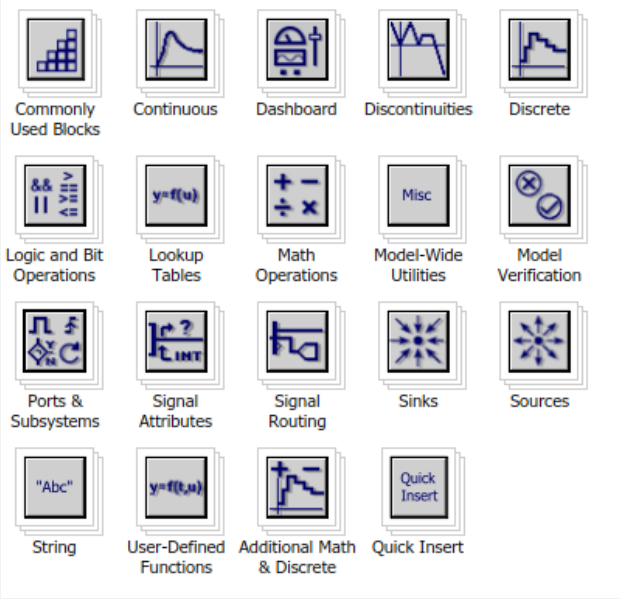
- Matlab's Simulink tool
  - Features, startup, basic blocks, DSP system toolbox
  - Model organization & execution

- Introduction to adaptive filters
  - Why needed?
  - General structure & error-based adaptation
  - Configurations for applications

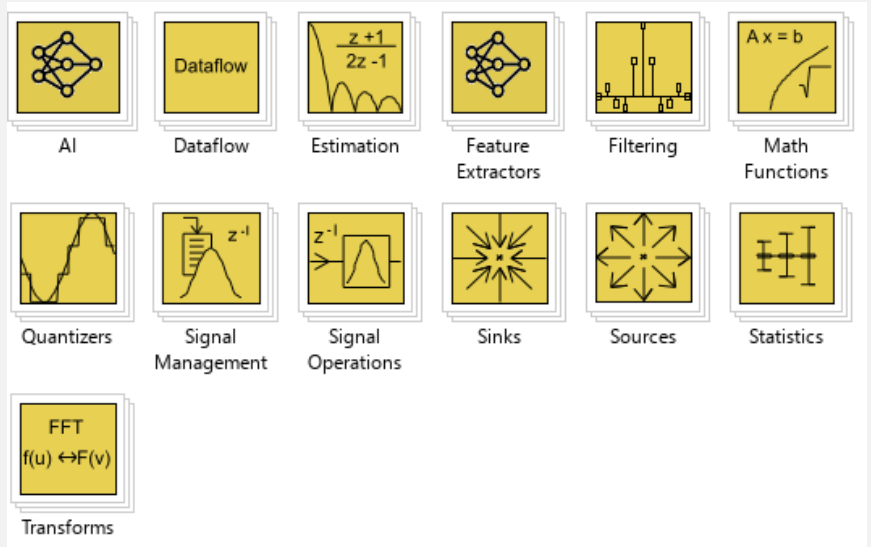
Simulink

Simulink features

- Graphical block based modelling tool
- Both continuous and discrete time simulations can be built
- Also code generation features
- Dozens of toolboxes for various modelling cases
  - Basic blocks
  - Audio, communications, computer vision, deep learning, robotics, …
  - HDL support: Coding, verification, communications, computer vision, DSP
  - Stateflow for control modelling
- In the last design task, quick look
  - Basic blocks and DSP System Toolbox used

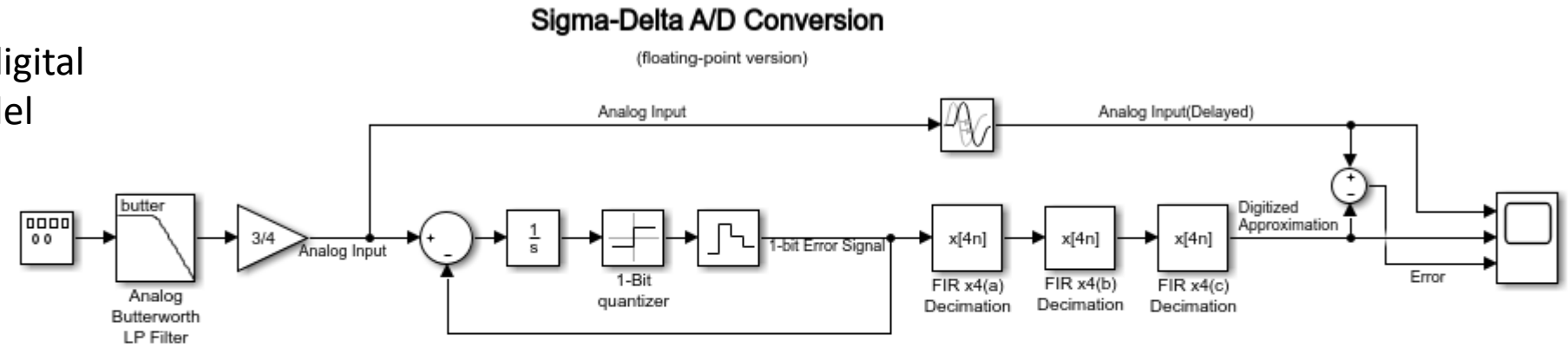


Subsets of basic Simulink blocks

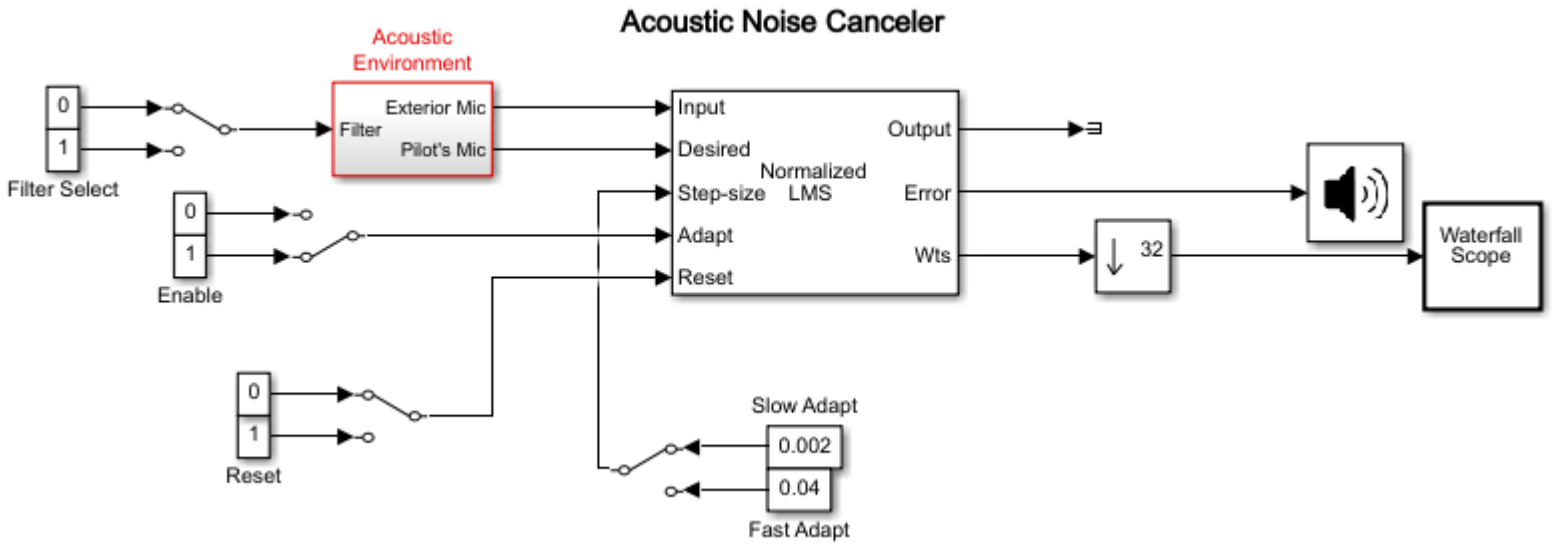


Block subsets in DSP System Toolbox

Example of a mixed analog/digital simulation model

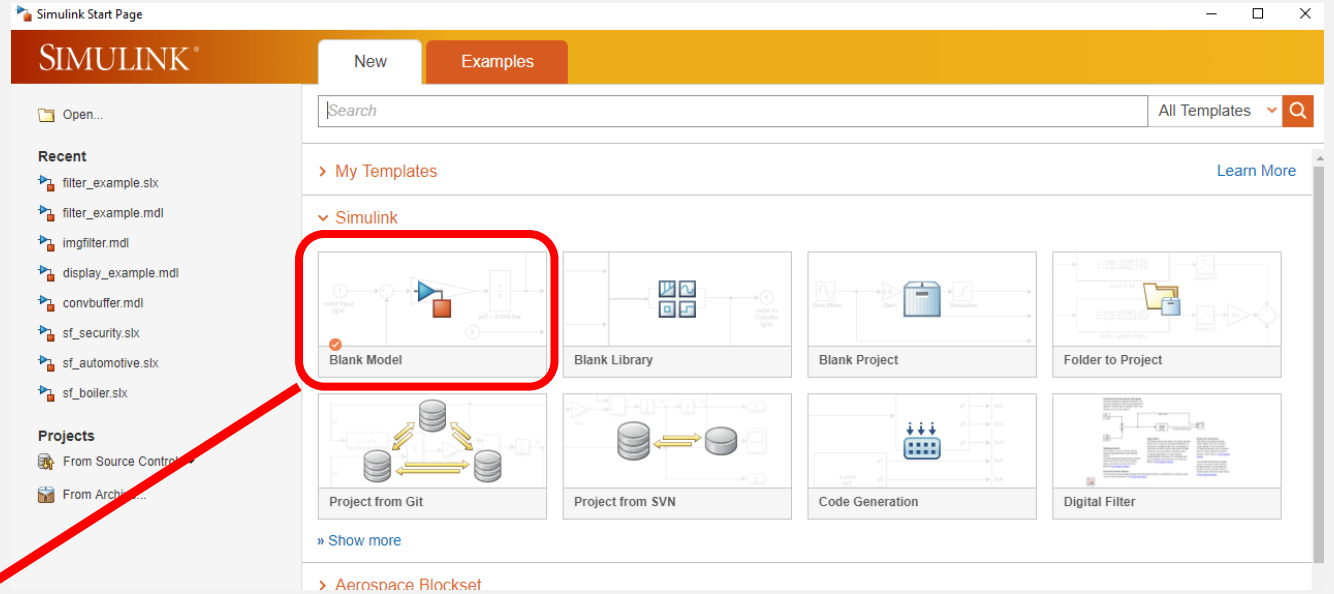


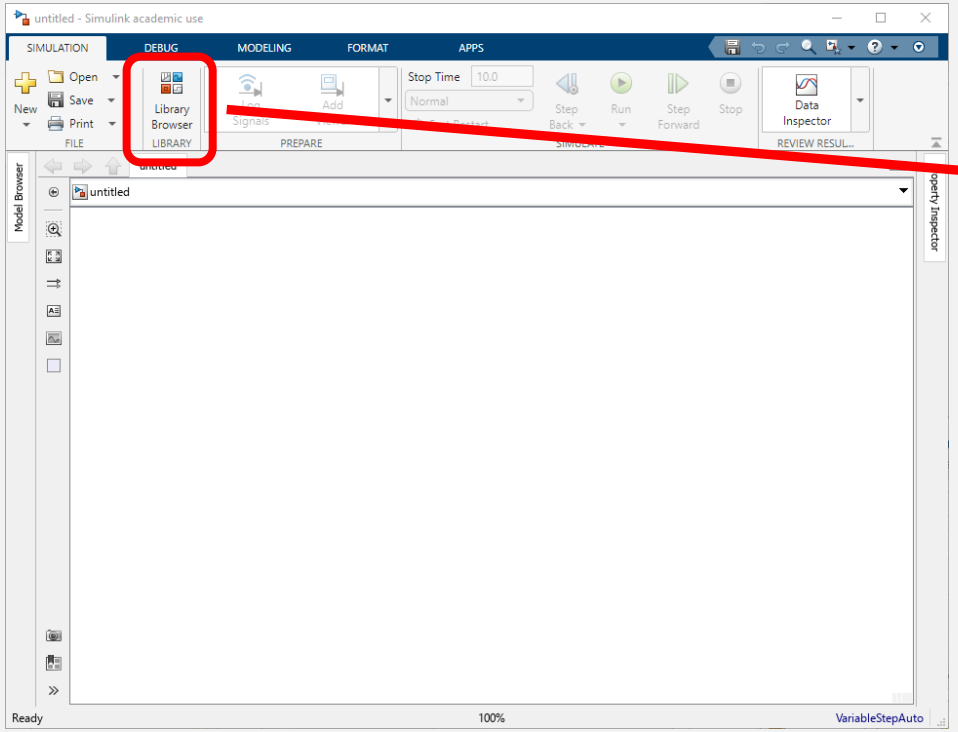
Example of an adaptive filtering test setup

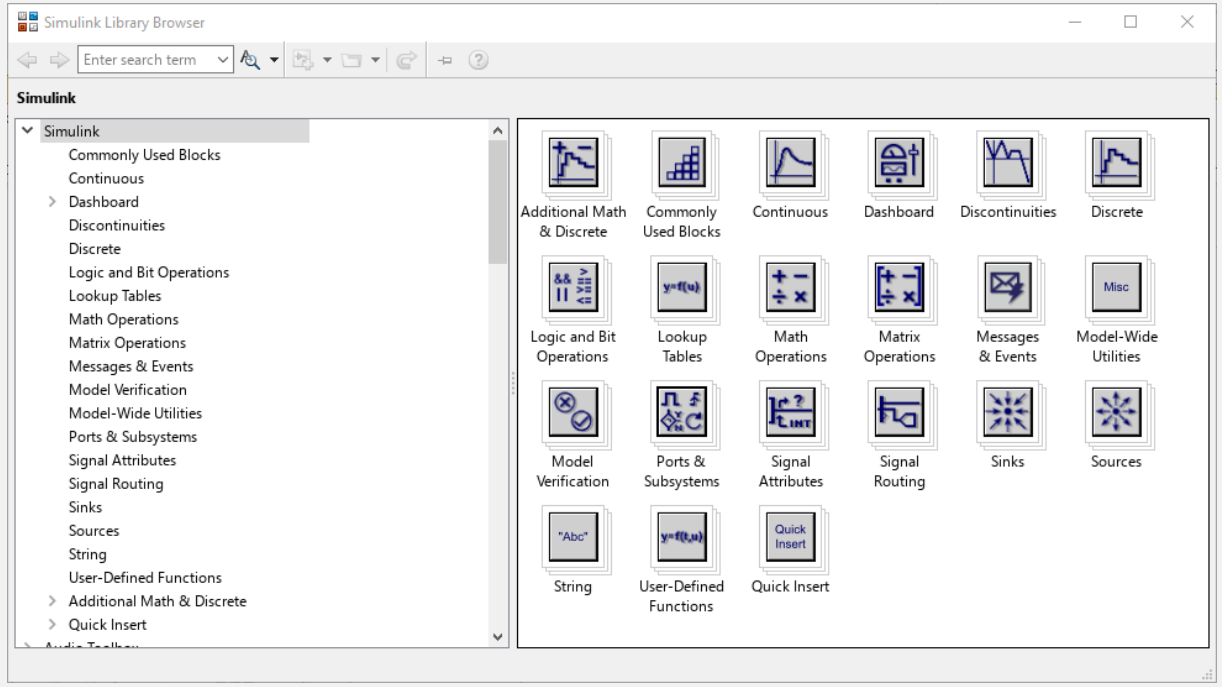


Startup

- Matlab command: **simulink**
  - opens Simulink Start Page
- Button: ”Blank Model”
- > Library > Library Browser
  - Access to toolboxes / blocksets





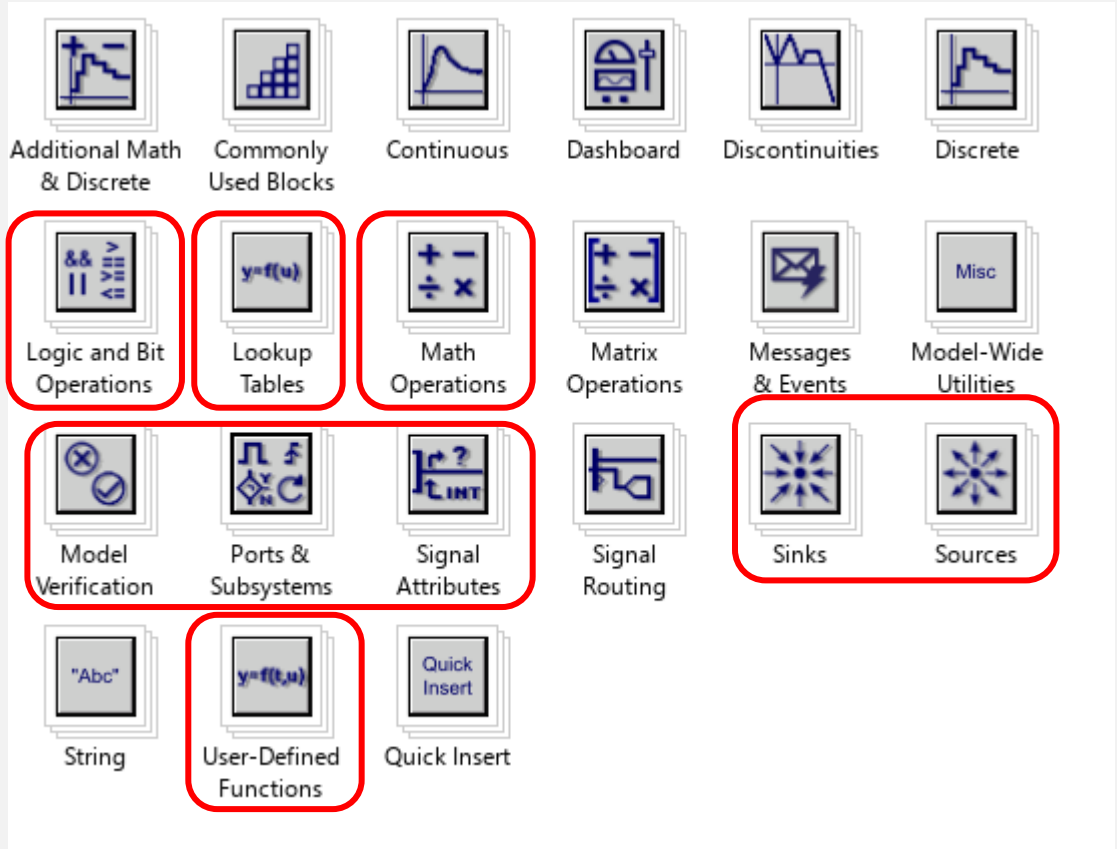


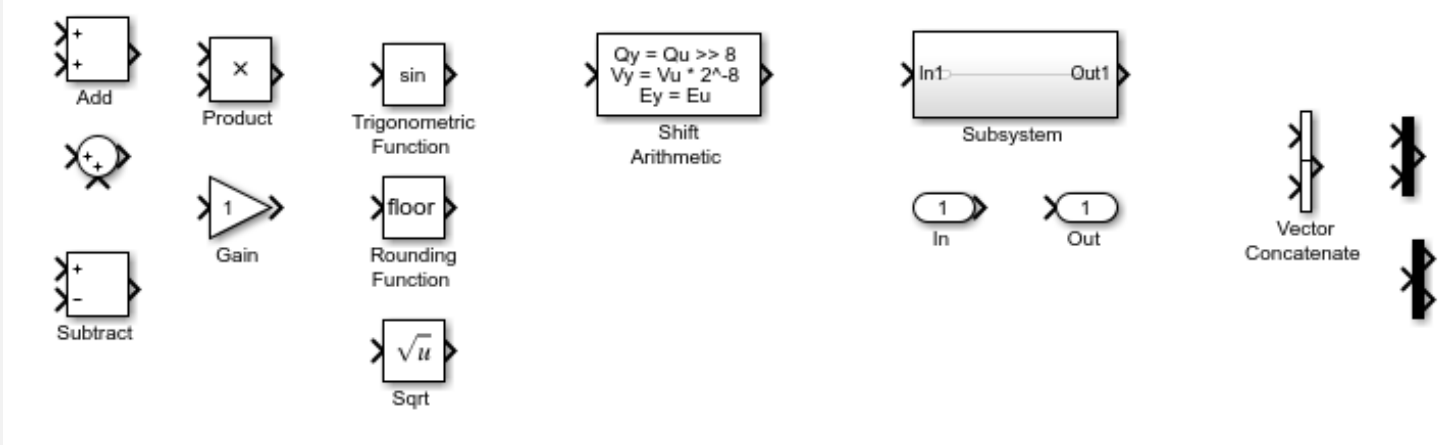
Basic Simulink blocks

Basic I/O
Basic math
Bit operations (e.g. arithmetic shift)
Model organization
User-defined functions

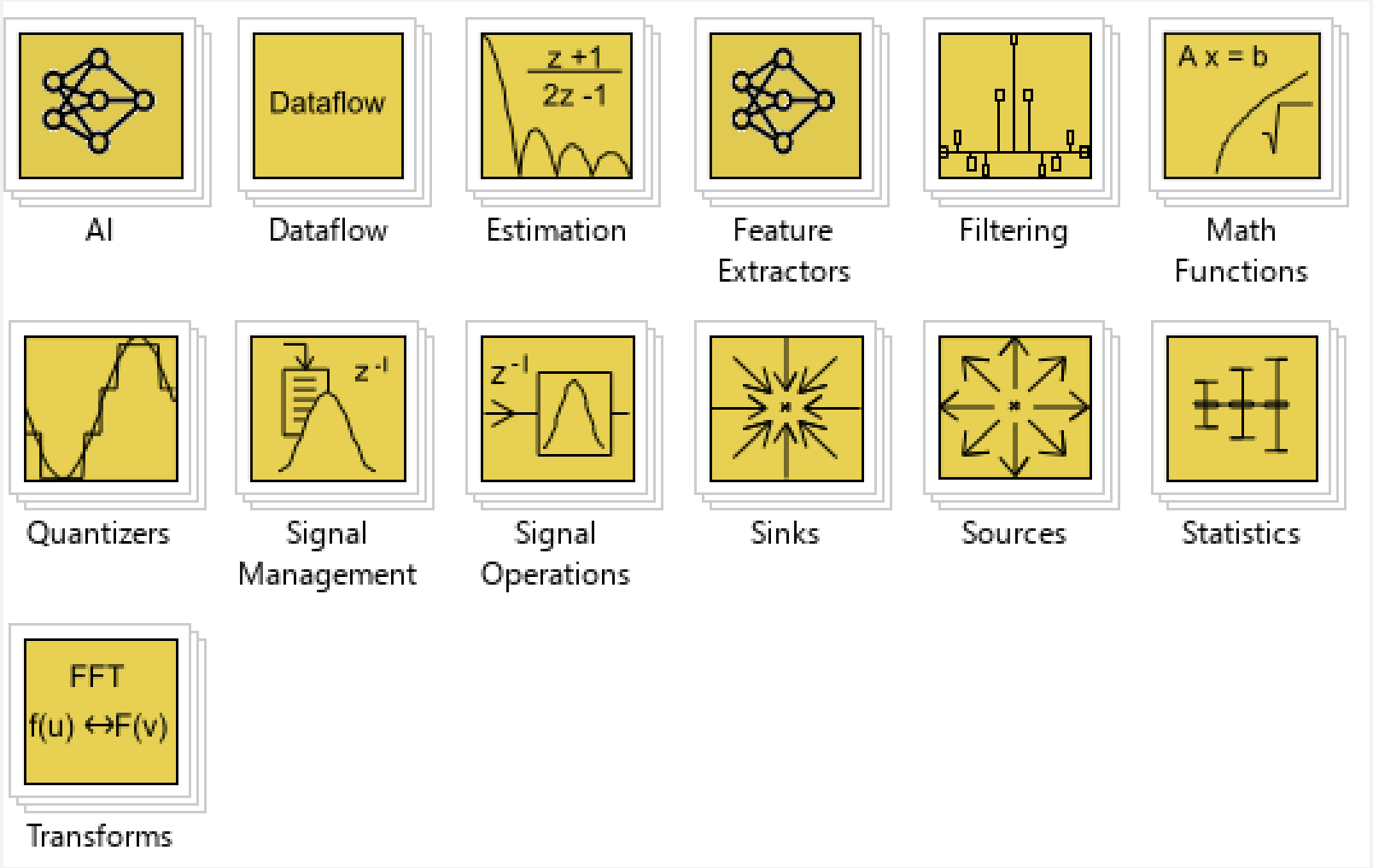
Note: There are blocks which are intended for continuous time simulations.

Sources and Sinks: Prefer blocks from DSP System Toolbox

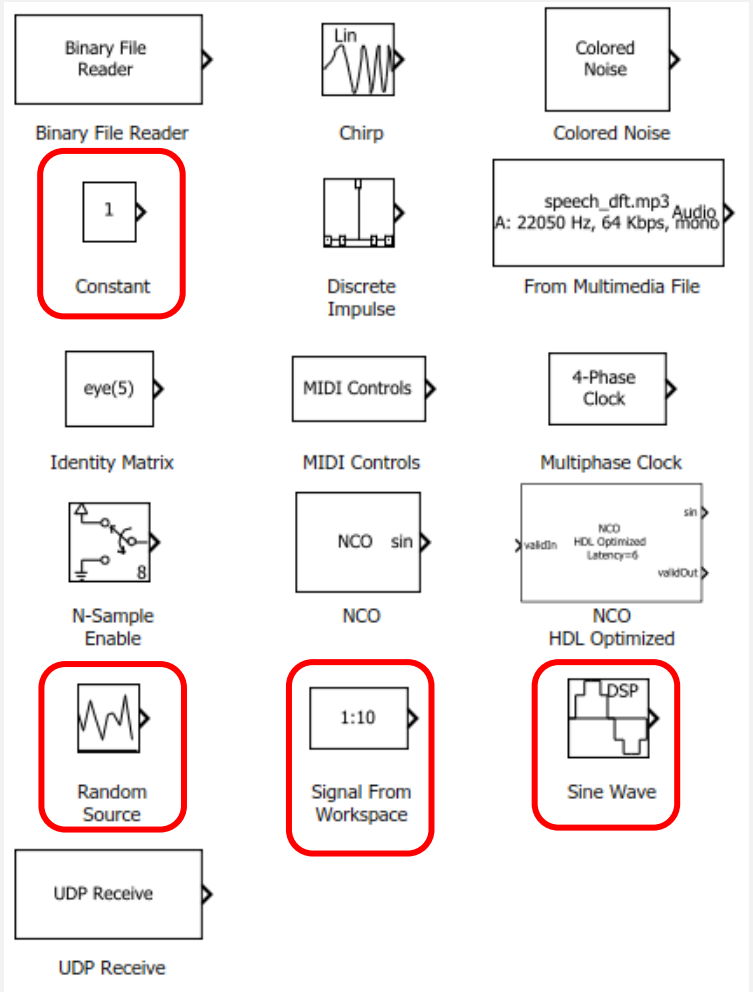




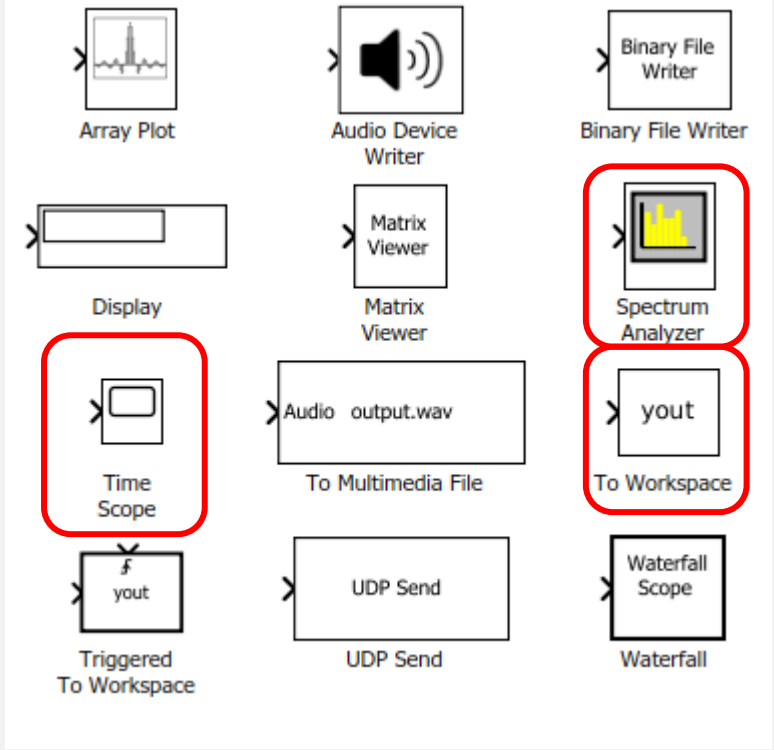
DSP System Toolbox



DSP System Toolbox / Sources and Sinks



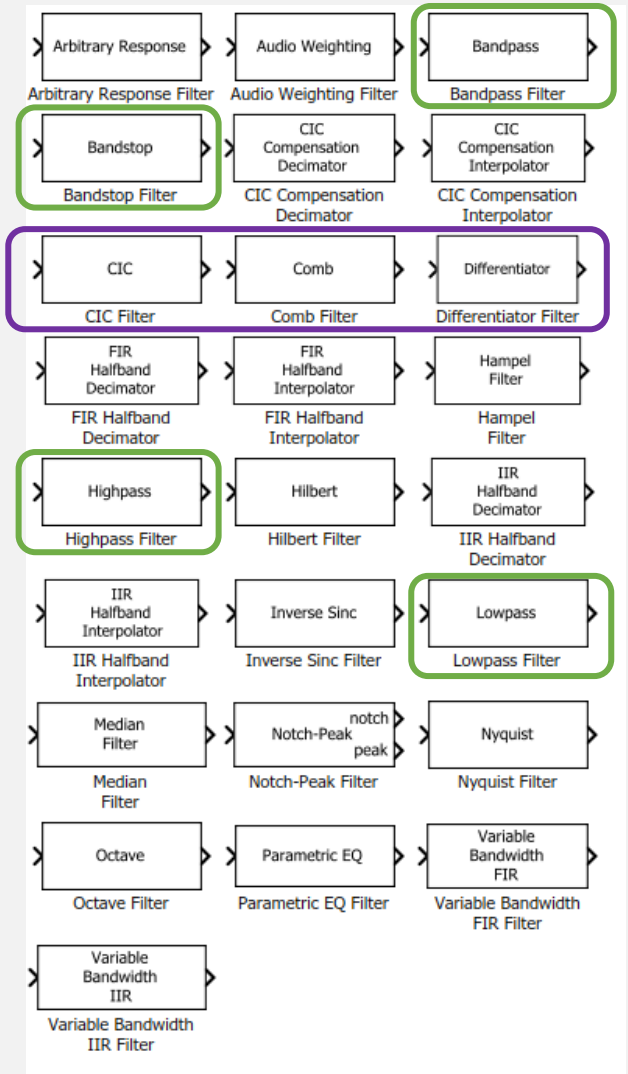
Sources

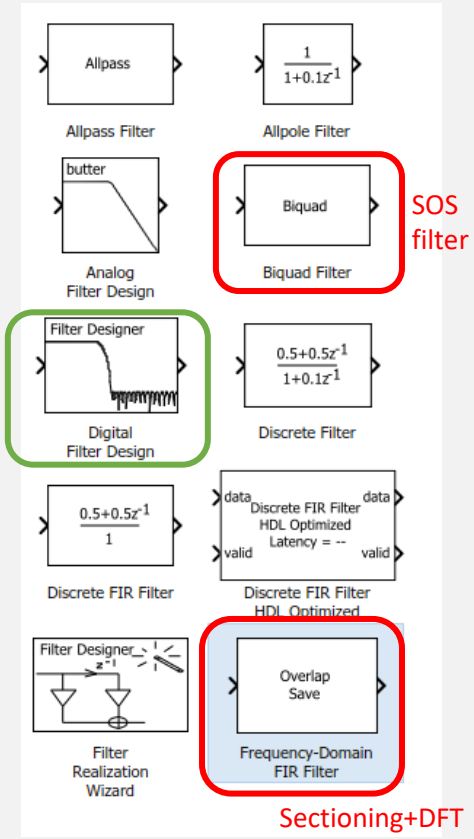


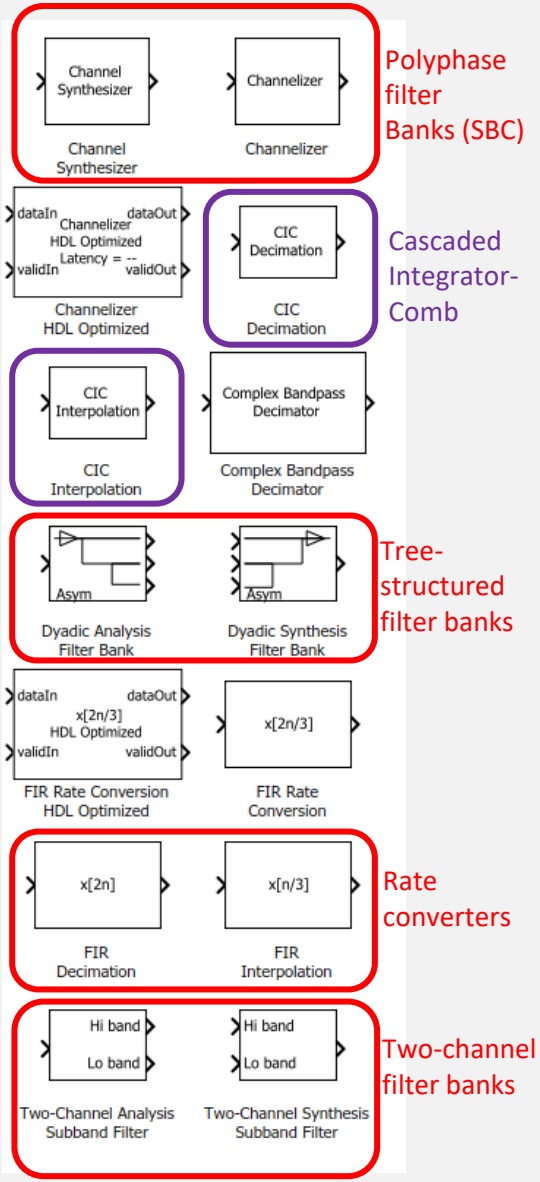
Sinks

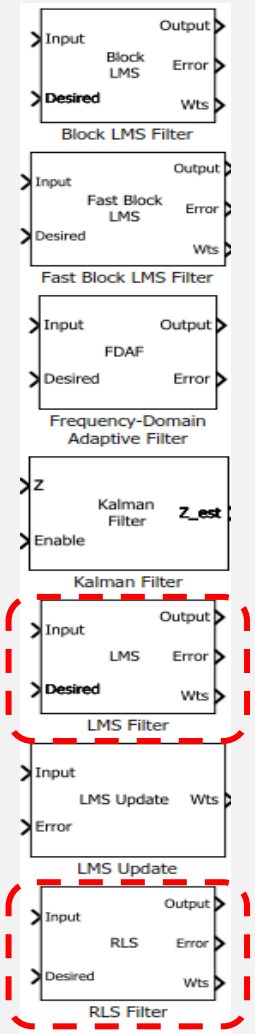
DSP System Toolbox / Filtering

Many techniques discussed in this course in four subsets









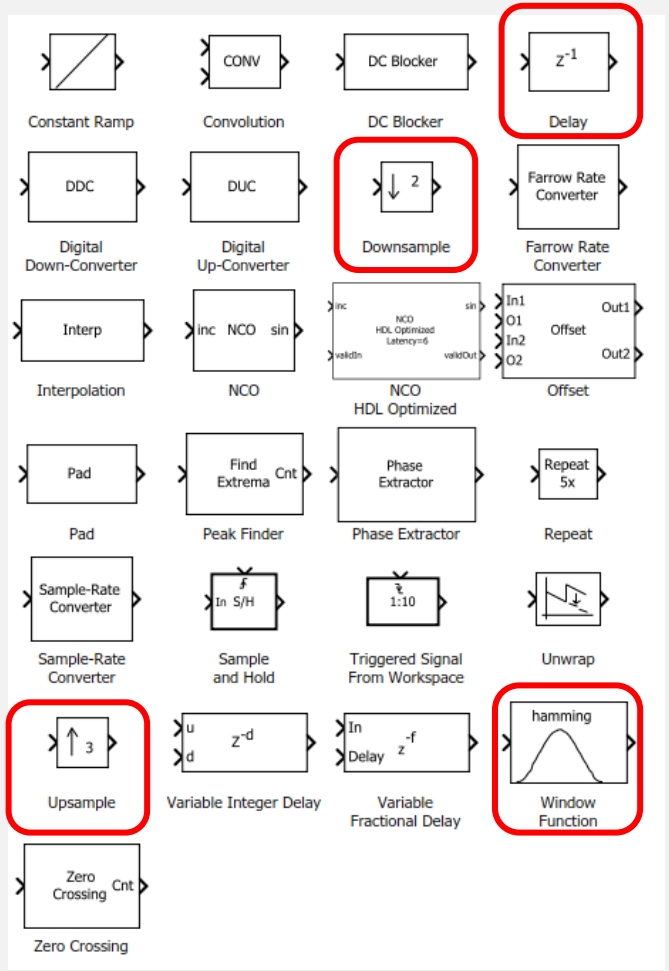
Filter designs

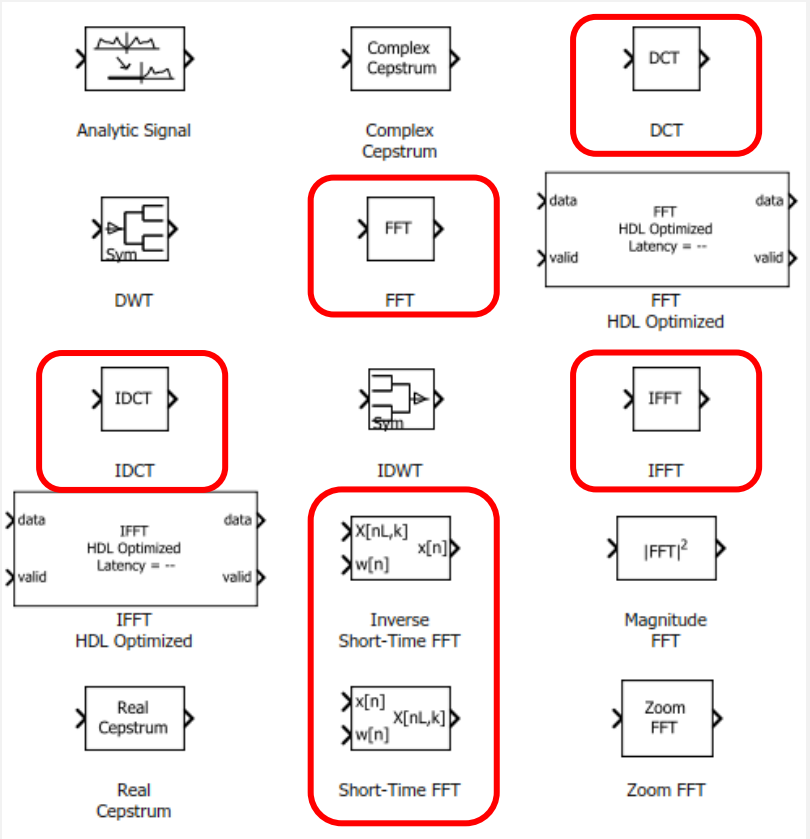
Filter implementations

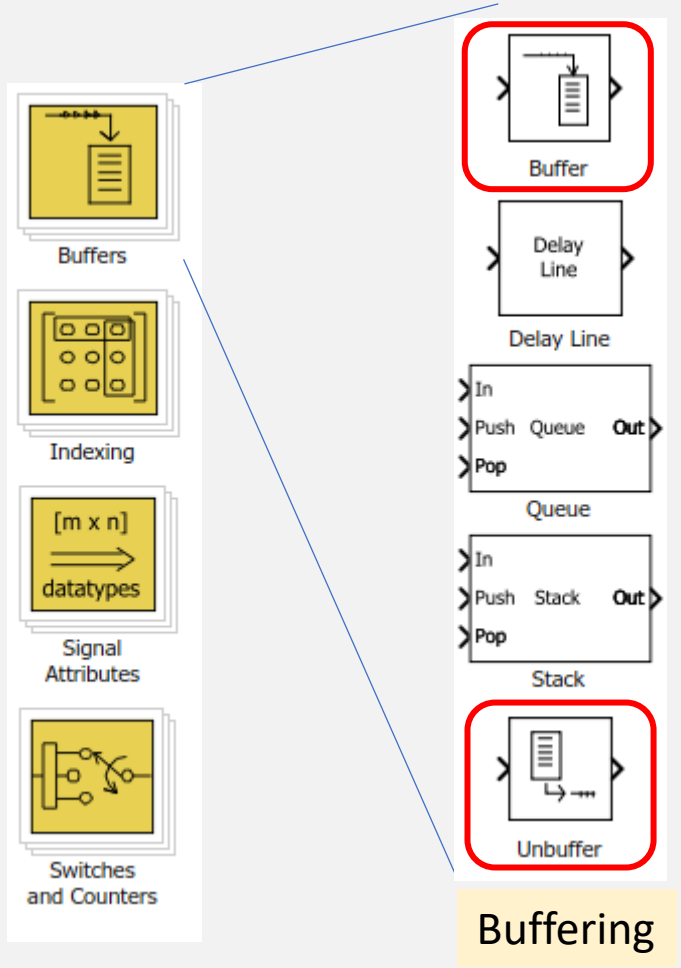
Multirate filters

Adaptive filters

DSP System Toolbox / Some other sets







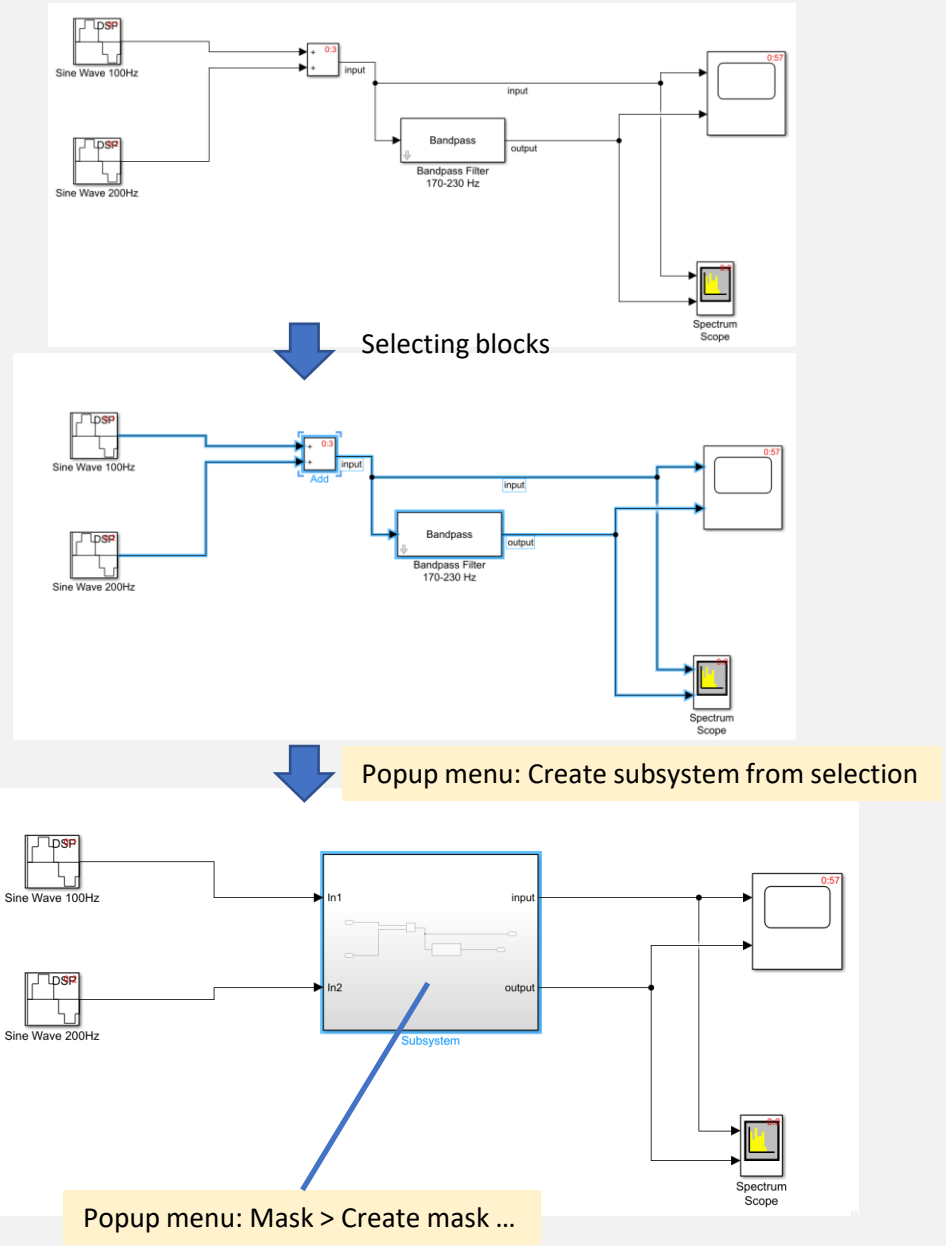
Signal operations

Transforms

Signal management

Organizing models

- Subsystems
  - Hiding details of implementation
- Masking
  - Parameterizing a subsystem
  - Can have initialization code
  - Can have documented interface like toolbox blocks have
- Libraries
  - Collection of masked subsystems for general use. Own libraries can be created

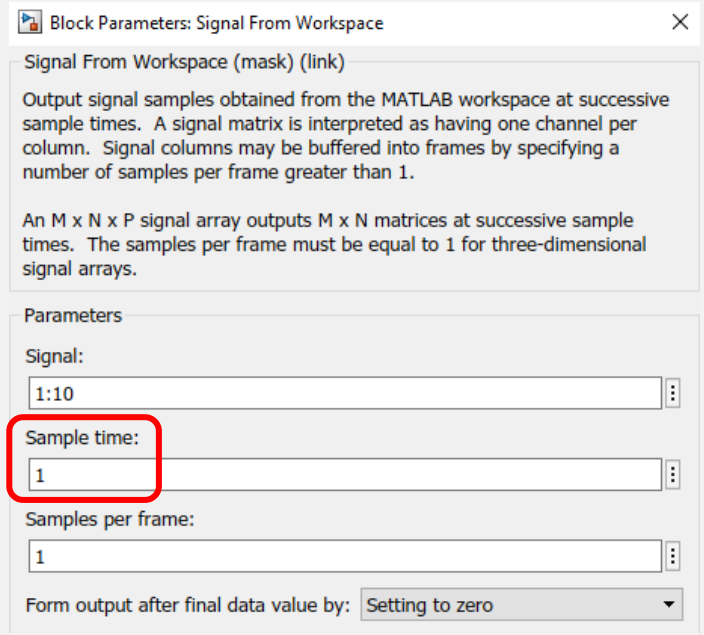


Execution of models

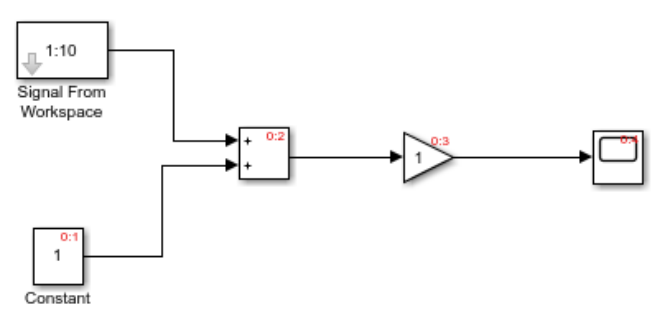
- Simulink models are dataflow models
- The model of computation (MoC) is based on simulation time (*)
  - **Sample Time parameter** specifies the update rate of the block's state and output (may be inherited from input)
  - Simulink puts blocks into some **execution order** based on their connections
  - Simulink determines clock ticks when it performs updating
    - At each tick, zero or more blocks are updated
  - Two phases in update:
    1. Outputs of all scheduled blocks are updated
    2. Possible internal states of those blocks are updated

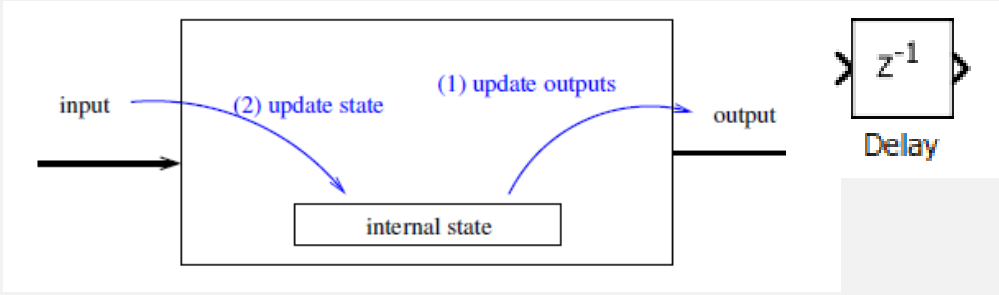
(*) DSP tools for dataflow modelling use different kinds of MoCs

- For example, Synchronous DataFlow (SDF) and its relatives
- Firm semantic basis for creating block-based dataflow models, analyzing their properties and translating them to final implementations (e.g. scheduling of operations)



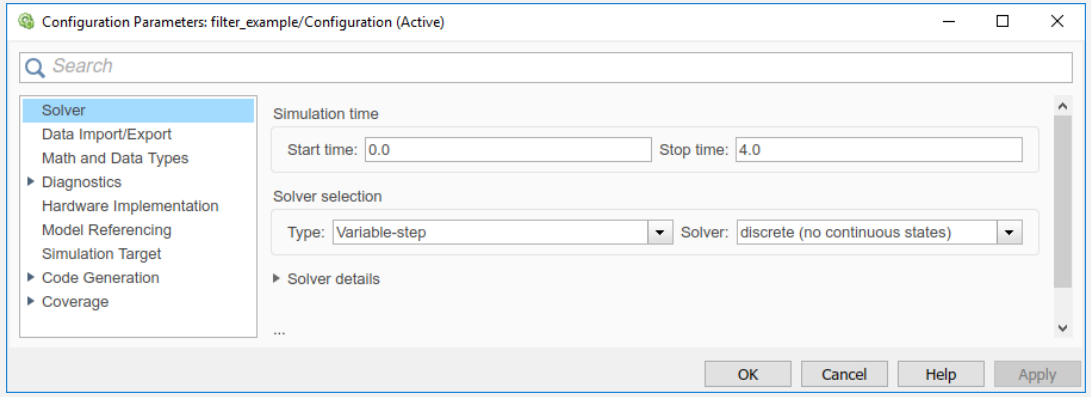
Display > Blocks > Sorted Execution Order

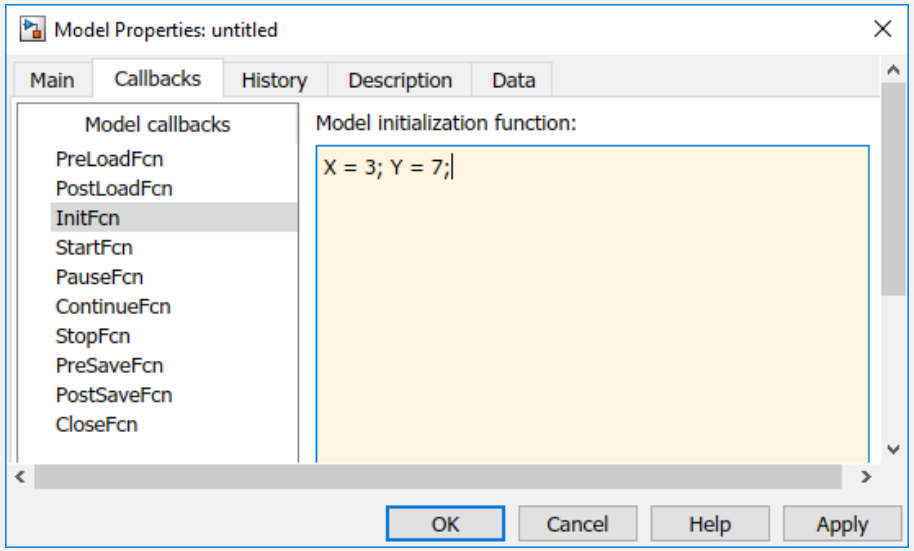




Execution of models ... Matlab R2024b

- Configuration
  - Modeling (tab) > Model settings
  - Start & stop times
  - Solver selection
    - e.g. selecting discrete simulation !!
- Access to a Matlab workspace
  - Some Matlab workspace used: workspace variables can be used as block parameters
  - Simulation can write new variables to the same workspace
- Initialization alternatives
  1. Prepare global workspace before simulation
  2. Model settings > Model Properties > **Callbacks**
     - Making the model self-contained
     - Note the availability of callbacks for postprocessing
  3. Writing a Matlab function for execution
     - **simset** – setting the workspace to function's workspace
     - **sim** – loading and executing the model
     - Recommended: good way for testing various parameter combinations
- Alternatives for running a simulation
  - Simulation > Run
  - sim command





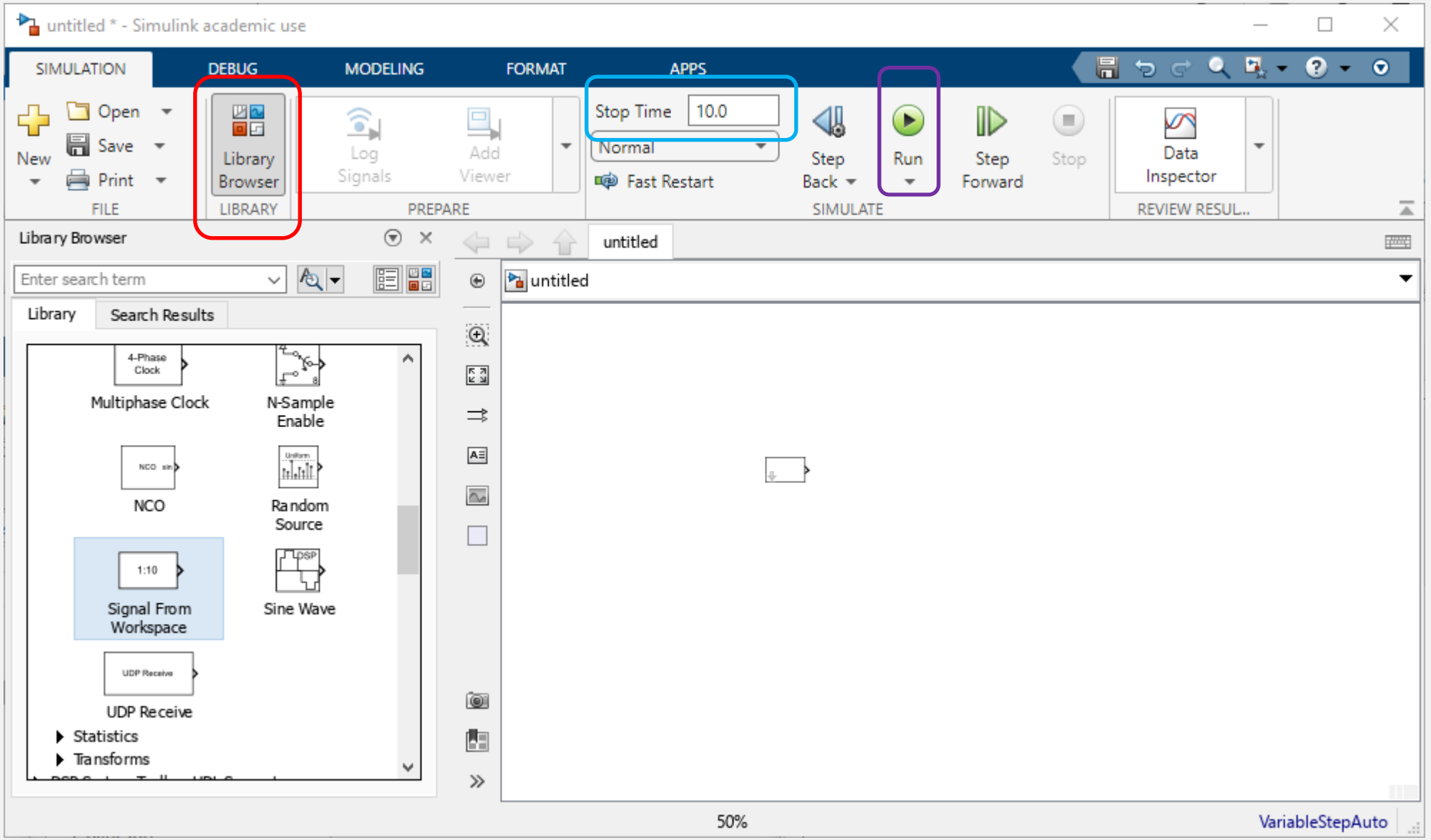

```
function exsim_run(L,D,step_size)                % Simulation input variables

    stop_time = 1.0;
    options = simset('SrcWorkspace','current');  % Workspace setup
    sim('exsim',stop_time,options);              % Simulation
    analyze_result_func(bit_error,errdiff);      % Analyzing simulation output variables

end
```

Simulink in Matlab R2024b

Appearance of Simulink depends on Matlab version

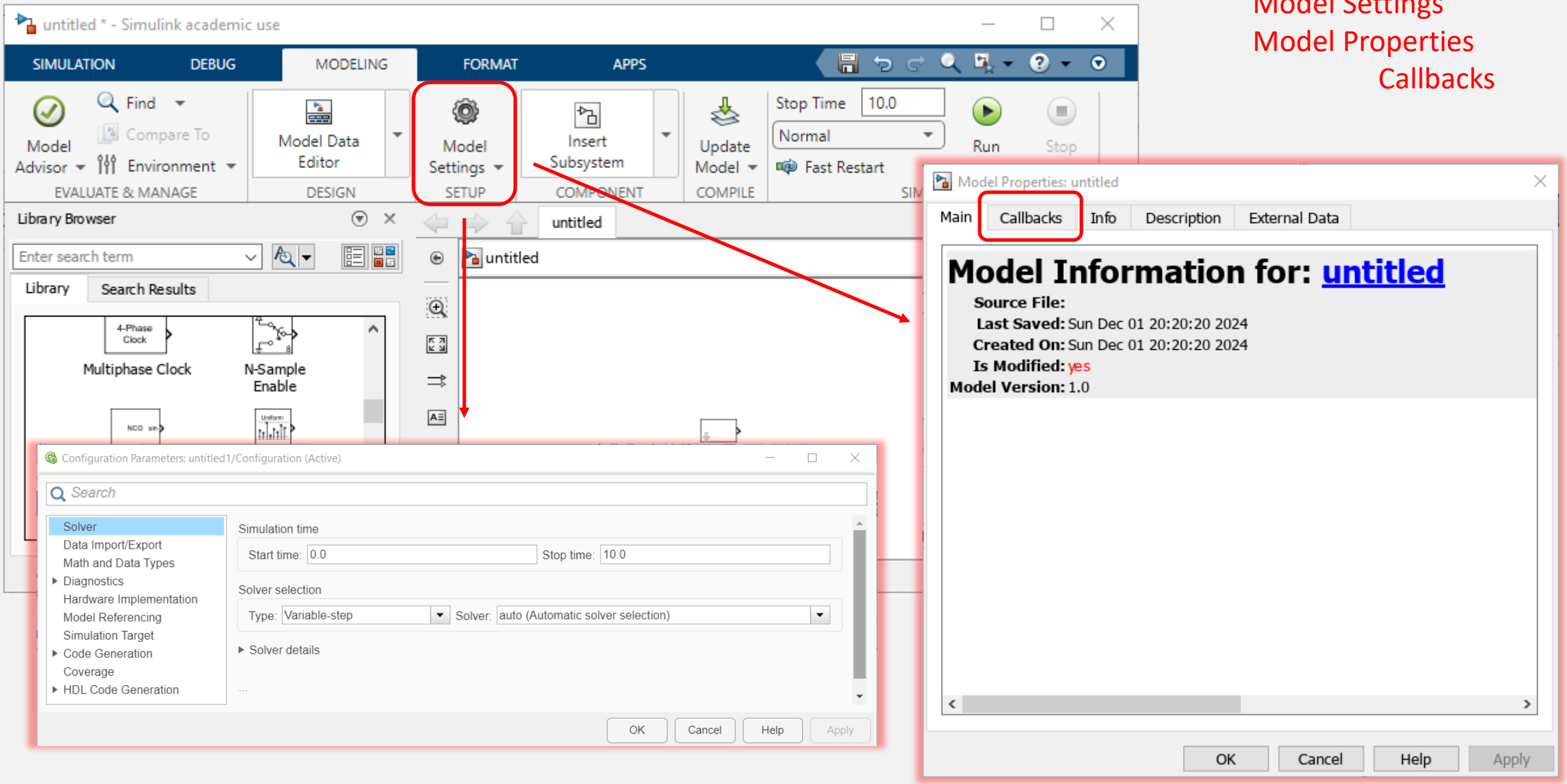


Library Browser
Stop Time
Run

Simulink in Matlab R2024b

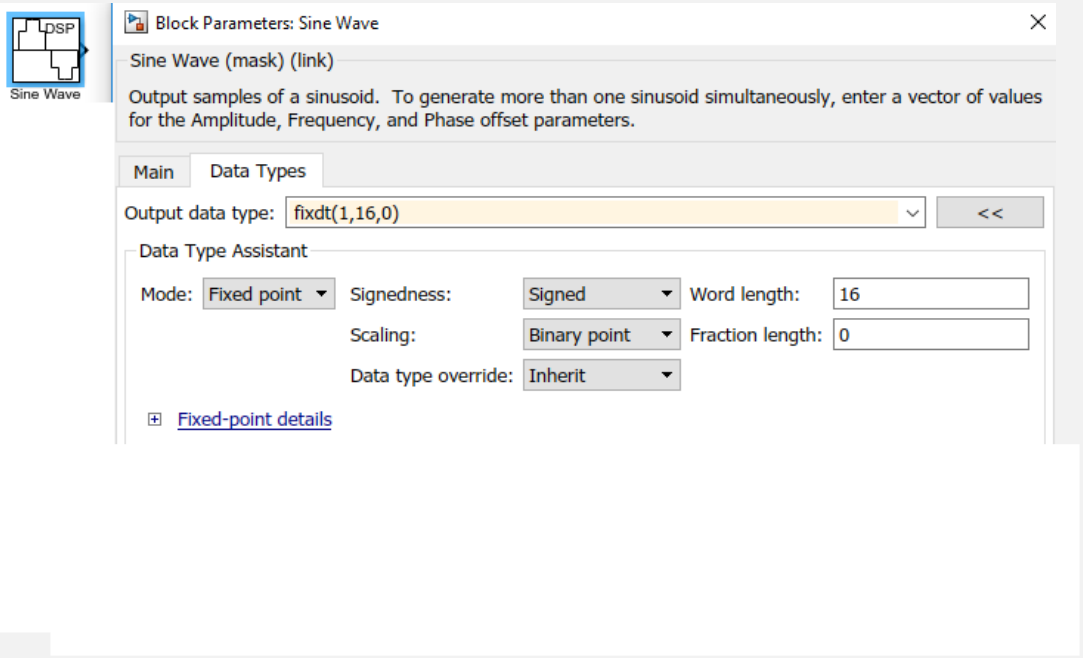
SETUP

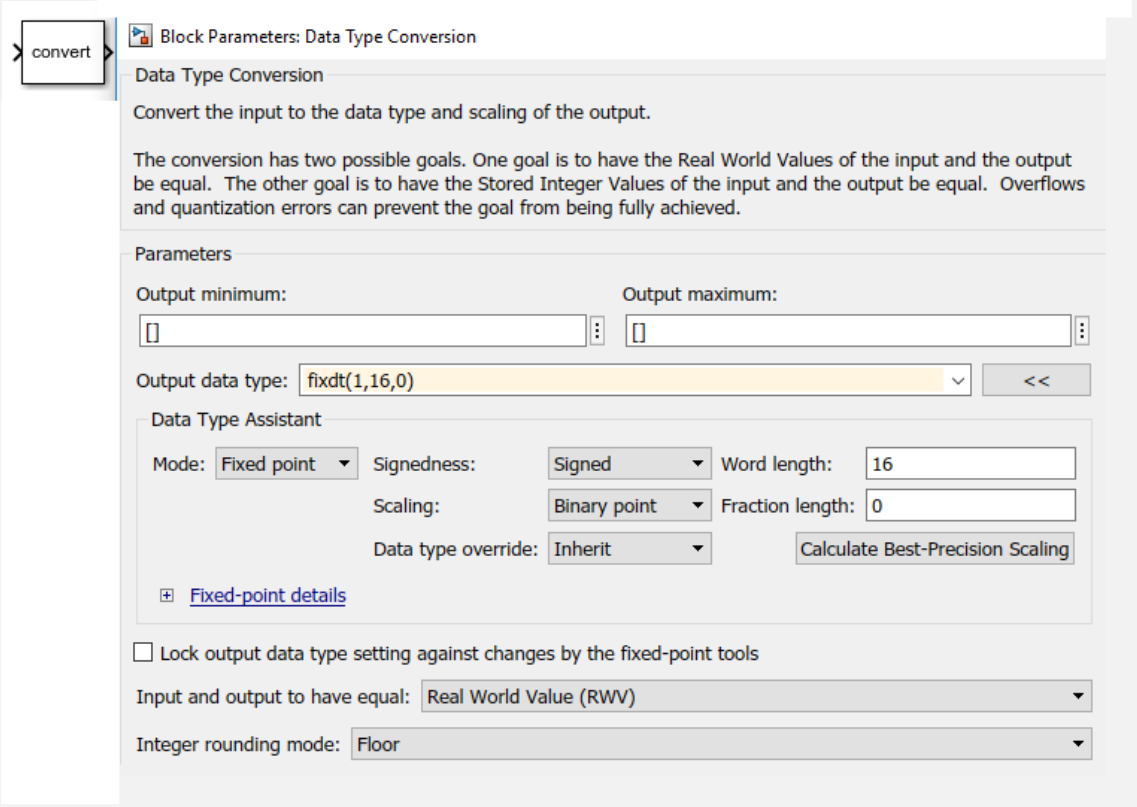
Model Settings
Model Properties
Callbacks



Note: Fixed-point simulation

Blocks from DSP System Toolbox: some have fixed-type simulation parameters like Sine Wave block.
Signal Management contains Data Type Conversion block for converting signal types.





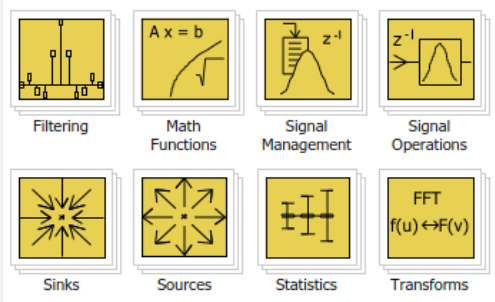
Testing: Data type override mode can be used to temporarily switch the data types in the model.

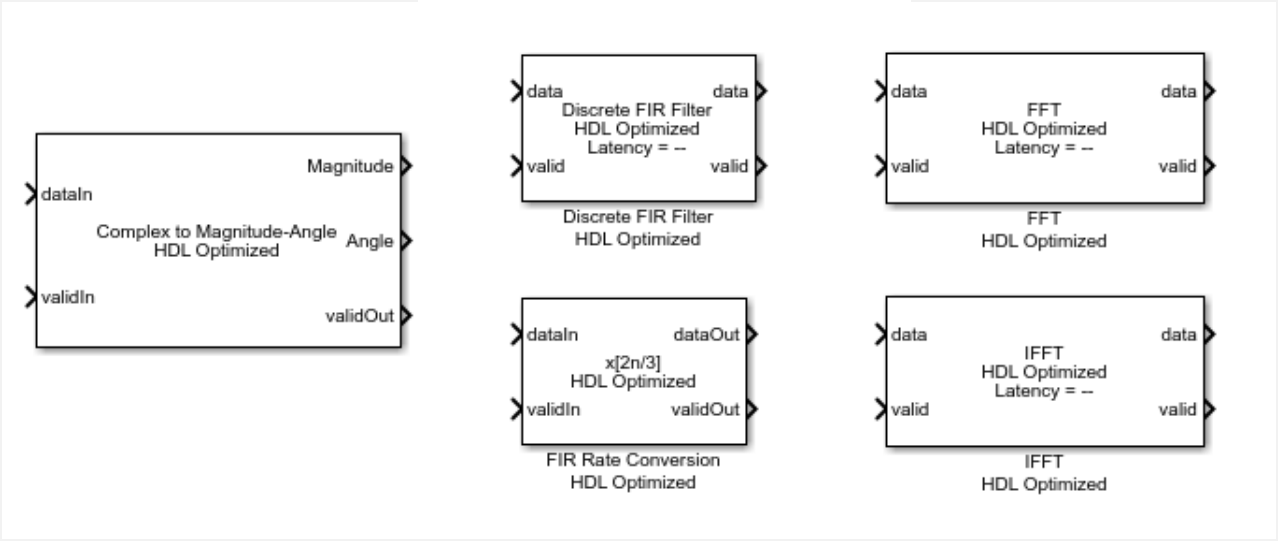
```
>> set_param('MyModel', 'DataTypeOverride', 'Double')
```

Note: DSP development support

- Beyond basic modelling
- Simulink Coder
  - Generating C and C++ code
- Matlab's HDL Coder
  - (HDL = hardware description language)
  - Generates portable, synthesizable Verilog and VHDL code for FPGA programming or ASIC prototyping
  - Workflows for Xilinx, Microsemi, and Intel FPGAs
  - Special Simulink blocks with HDL code generation support used

DSP System Toolbox HDL Support

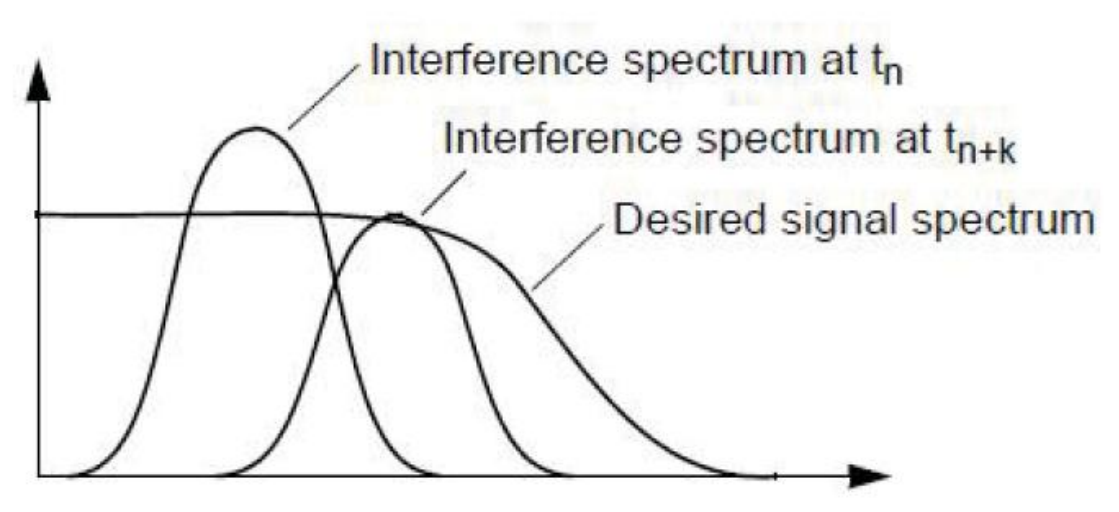




Adaptive filters

A1. Need for adaptive filters

- Fixed frequency selective filters
  - Mostly useless, when the signal of interest and the interferences are in the **same frequency band**
  - Also problematic, when characteristics of the interference are **changing over time**



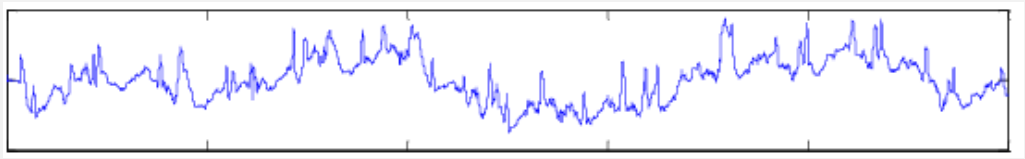
- Solution in adaptive filtering
  - one uses **another input** signal, which provides information about the disturbance source

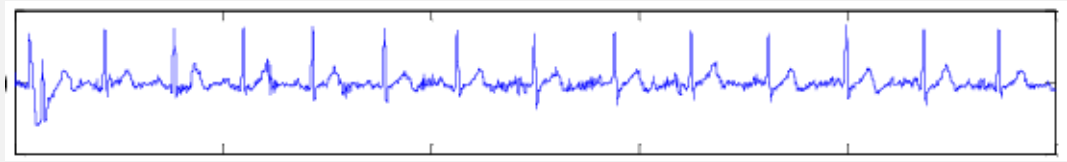
We focus to this class of two-input signal processing.

Note that the term ”adaptive” is used also in other contexts, where processing has to be adapted to some temporally / spatially local conditions.

Some examples of adaptive filtering

- Analysis of biosignals such as ECG and EEG. For example, the measurement of the ECG (and heart) rate is complicated by varying interference from the muscle and breathing signals.
  - Reference measurements for those interferences used to improve biosignal of interest



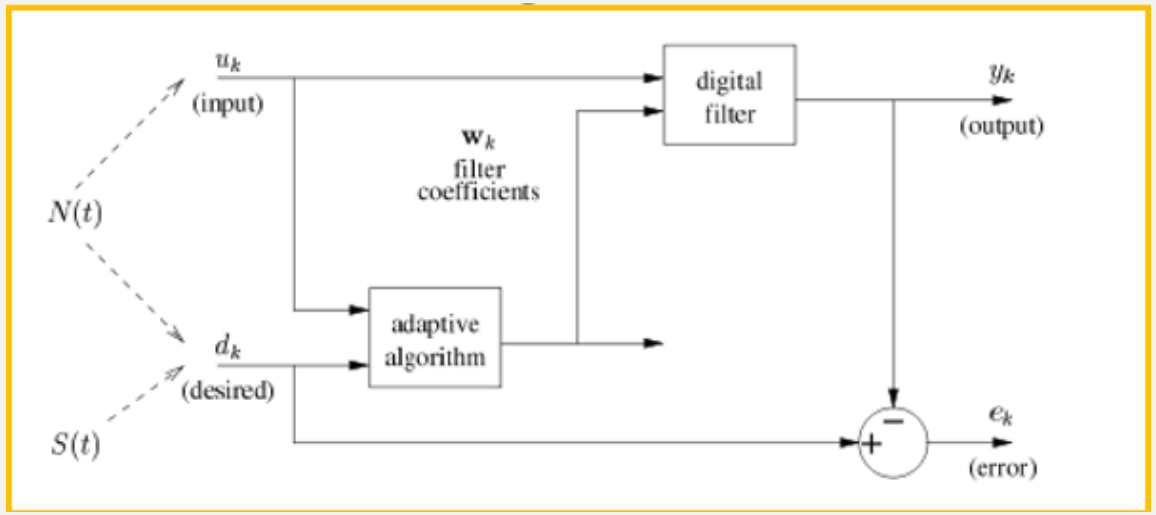


- Receivers in telecommunication must adapt to the changes in the channel conditions.
  - Pilot signals utilized for learning the channel

- Active noise cancellation for headphones requires generation of an antiphase signal that depends on the current noise conditions.
  - Microphones used to measure conditions and noise cancellation quality.

A2. General structure

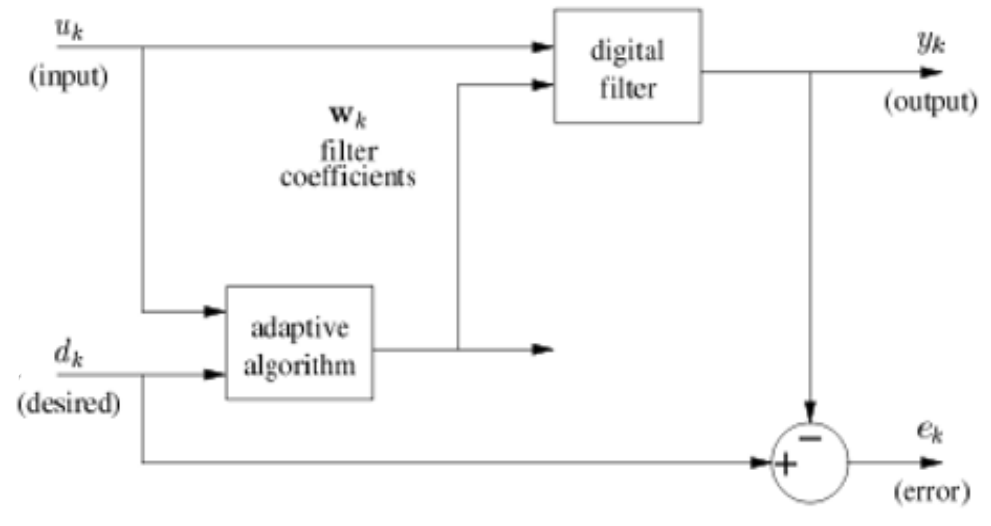
- An adaptive filter consists of two components, a digital (FIR) filter and an adaptive algorithm
- Adaptive algorithm computes coefficients for the filter, updating them continuously.
- The goal is to produce a signal at the filter's output, which corresponds to the effect of source $N(t)$ on the signal d_k.
- Computing the difference of d_k and the filter output y_k reveals then the content in d_k that depends on the source $S(t)$



k is the discrete time index

If there is no source $S(t)$ then e_k is 0 in optimal situation

Terminology & notation



k is the time index

L	The number of filter coefficients
d_k	The *primary* input signal (also called the desired signal)
e_k	The error signal, the difference between d_k and y_k
u_k	The *reference* input signal (input of the digital filter)
$\mathbf{u}_k$	Vector of input signal values, $[\, u_k,\ u_{k-1},\ \ldots,\ u_{k-L+1} \,]^T$
$w_k(i)$	Filter coefficient i (weight i)
$\mathbf{w}_k$	The vector of filter coefficients, $[\, w_k(0),\ w_k(1),\ \ldots,\ w_k(L-1) \,]^T$
y_k	The filter output signal, equal to $\mathbf{w}_k^T \mathbf{u}_k$

FIR: $y_k = \sum_{n=0}^{L-1} w_k(n) u_{k-n} = \mathbf{w}_k^T \mathbf{u}_k$

When the adaptation works?

- Consider a case, where d_k is equal to the sum of the delayed u_k and some signal s_k. Assume that $s_k = 0$.
  - The goal of adaptation is to minimize the error e_k
  - Optimally, it is zero
  - What is the digital filter response that achieves this?

- **Answer**: the impulse response of the filter, $\mathbf{w}_k$, should be such that

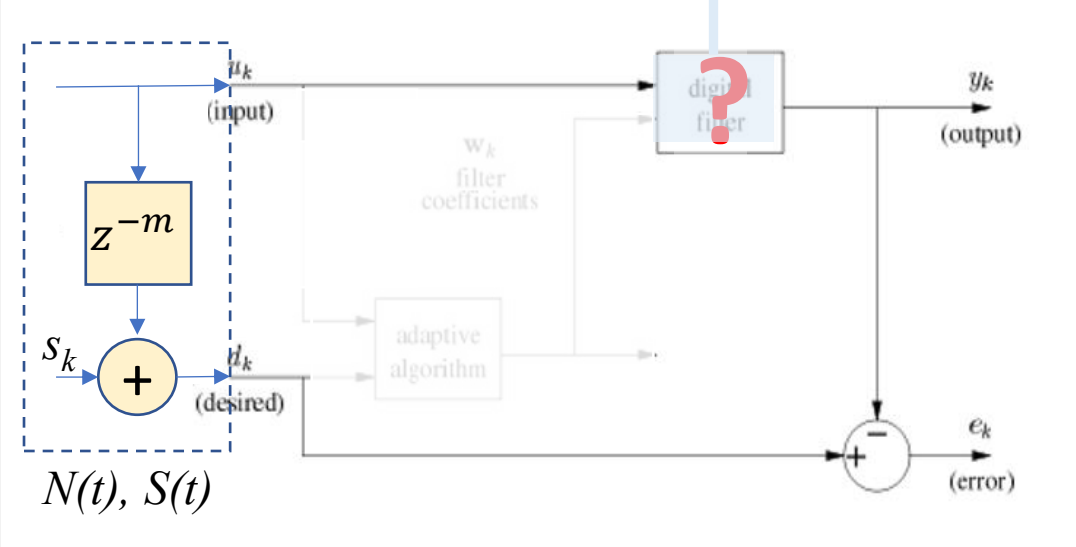
$$w_k(n) = \begin{cases} 1 & \text{for } n = m \\ 0 & \text{otherwise} \end{cases}$$

  - That is, it matches with the delay element shown in the figure. Due to delay, $d_k = u_{k-m}$
  - The goal of the adaptive algorithm is to find out this
  - Done by processing the sequence of u_k and d_k values

- **Lesson:** Temporal order is important
  - Effect of interfering process *N(t)* is earlier for u_k than for d_k
  - Only such a causal relationship between u_k and d_k can be modelled by the filter, it cannot be vice versa!

k is the time index

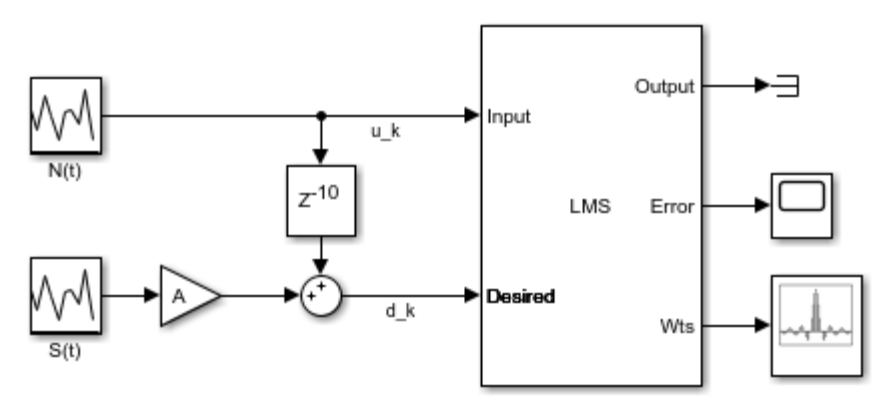
FIR: $y_k = \sum_{n=0}^{L-1} w_k(n) u_{k-n} = \mathbf{w}_k^T \mathbf{u}_k$



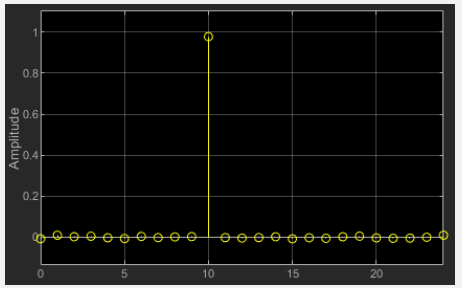
Demonstration in Simulink

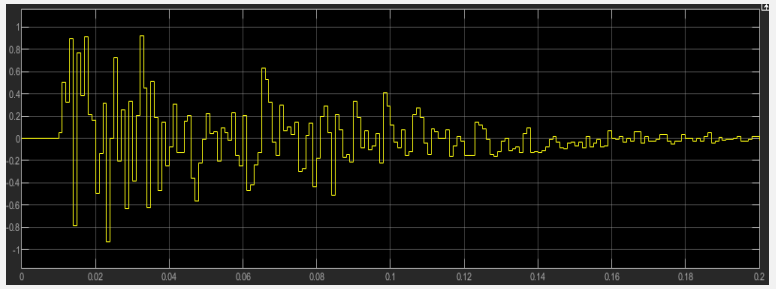
simple_example.slx in Moodle (set gain A before simulation)

Independent random sources (uniform pdf)

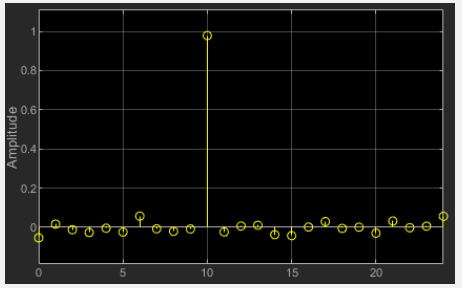


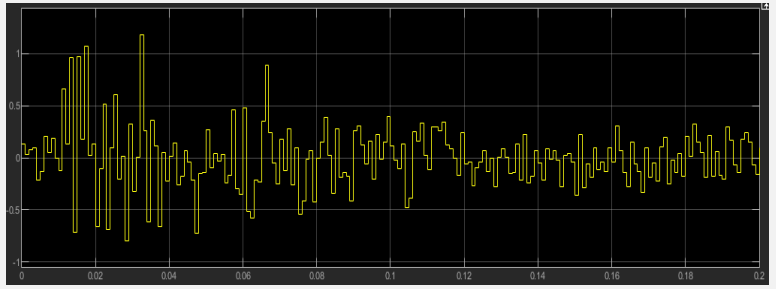
A = 0





A = 0.25

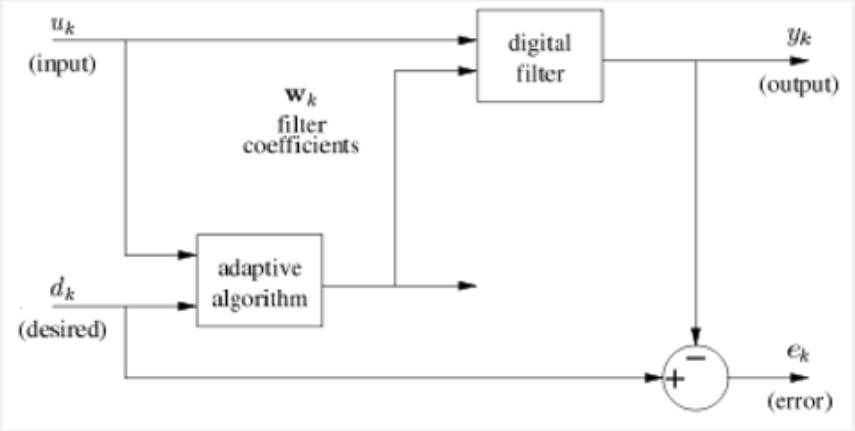


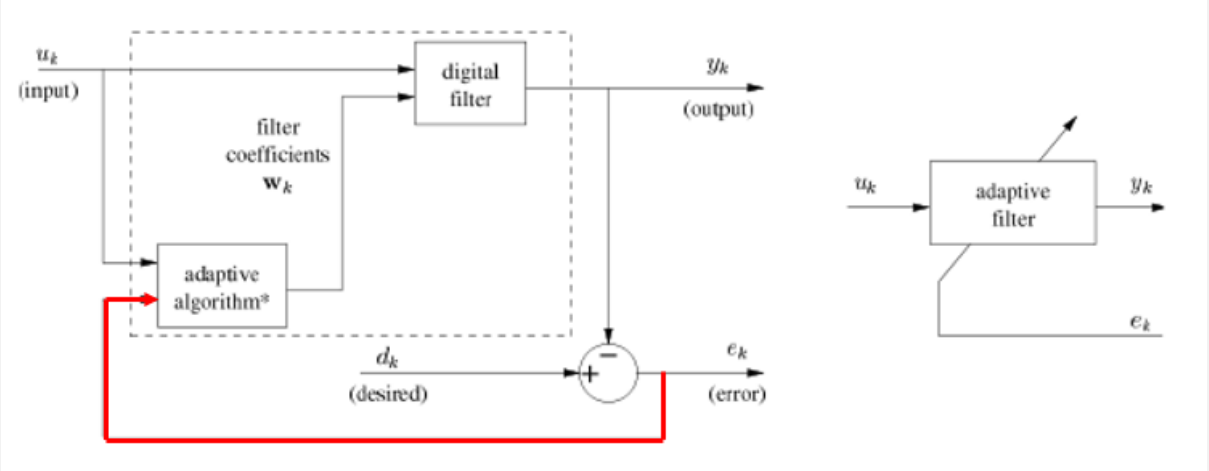


LMS block is from DSP System Toolbox, contains FIR filter & adaptive algorithm (Least Mean Squares).

A3. Error-based adaptation

- In the general structure, the digital filter and adaptive algorithm are shown as separate components
- However, many adaptive algorithms, like LMS and RLS, use a feedback connection from the error signal
  - In some sense, the digital filter becomes a part of the adaptive algorithm
  - The error signal is typically a **computed** value, but it can also be a **measured** value

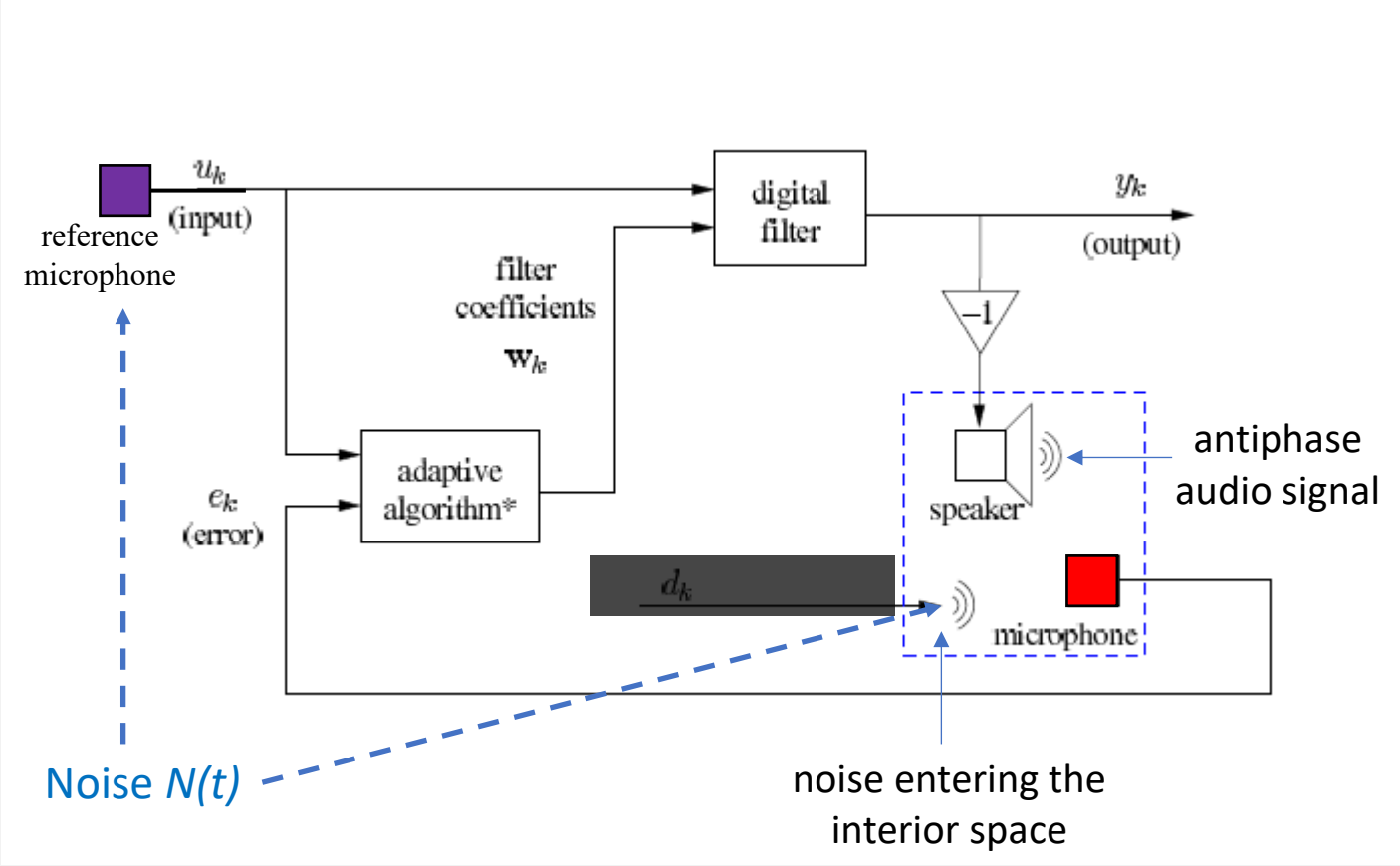




Error-based adaptation with computed error signal

Example. Active noise cancellation

- An example of error-based adaptation, where the used error signal is measured
- Two microphones:
  - The microphone outside provides reference information about the noise
  - The microphone inside measures the quality of noise cancellation. d_k is not available
  - Spiky noise should enter the headphone interior space later, which provides time to produce the antiphase signal
  - With periodic noise, this is not so critical

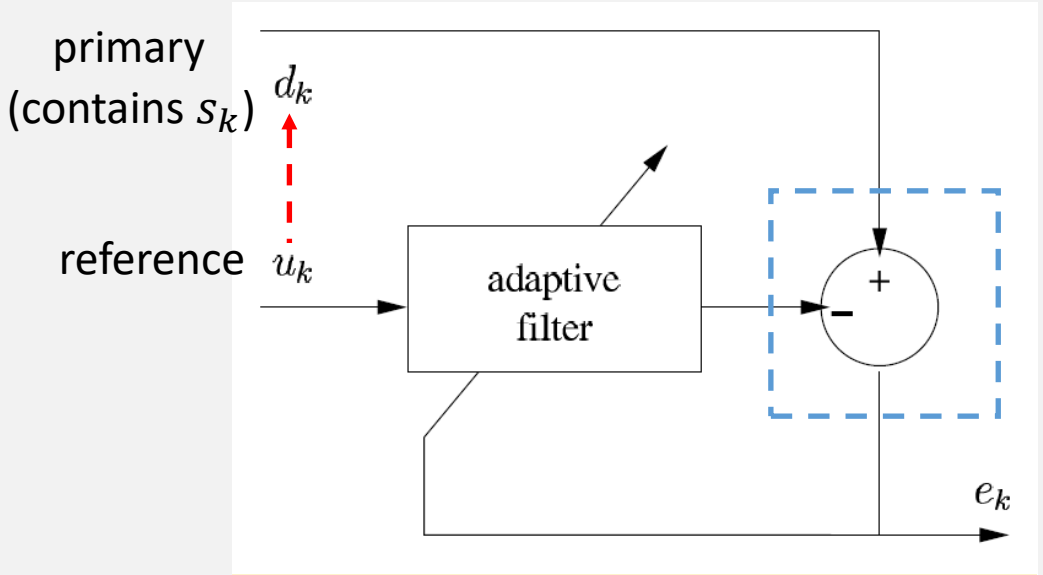


A4. Application schemes

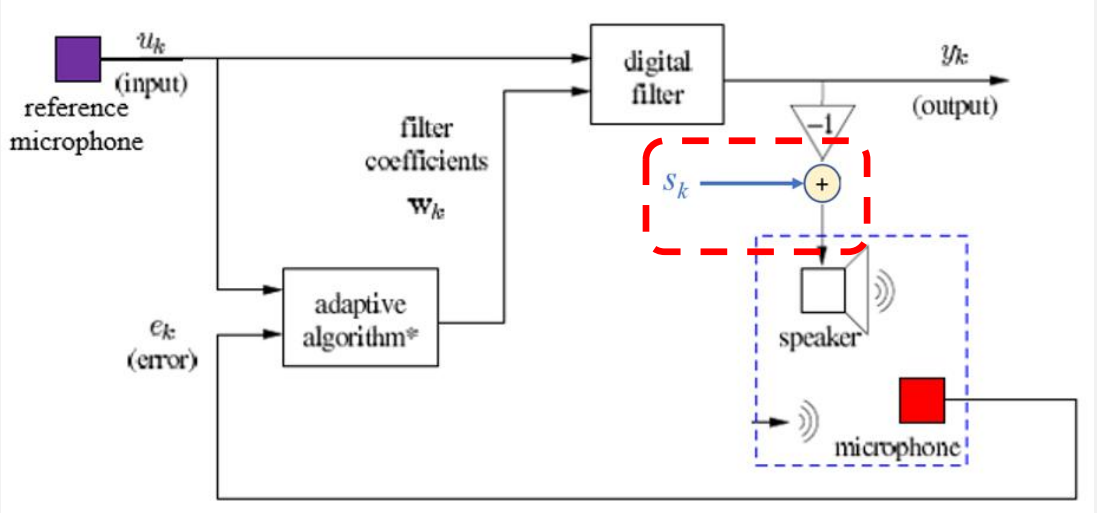
- Adaptive filtering is used in many kinds of applications with different input/output configurations:
  - Noise cancellation
  - System identification
  - Inverse system modelling (equalization)
  - Narrowband signal enhancement

Configurations (1)

- Reference input provides information about the noise
- Output of the adaptive filter should provide a signal that corresponds to the noise within the primary input signal
- As a result, the error signal provides information about the signal of interest s_k
- Also measurement based configuration can be sketched



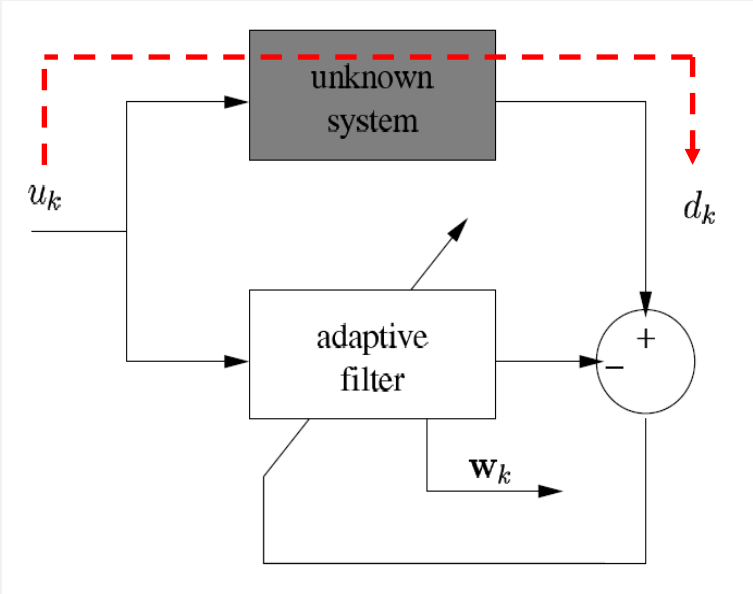
Noise cancellation (by computation)



Noise cancellation (by measurement)

Configurations (2)

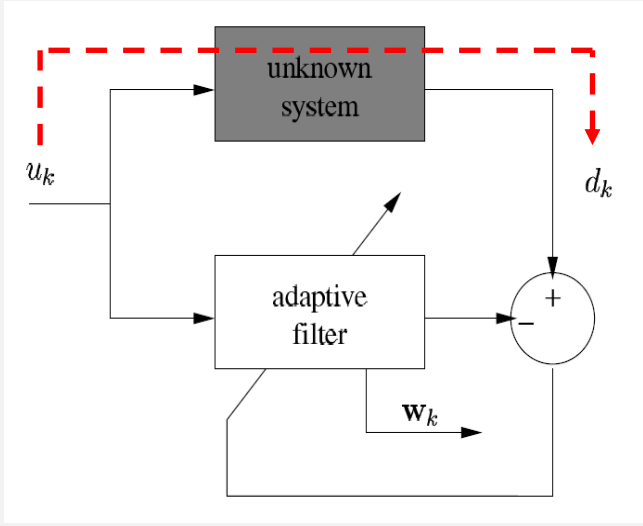
- In system identification, a wideband signal (e.g. a sinc pulse) is used for excitation of both the unknown system and the adaptive filter.
- After convergence, the adaptive filter coefficients correspond to the **system's impulse response**.



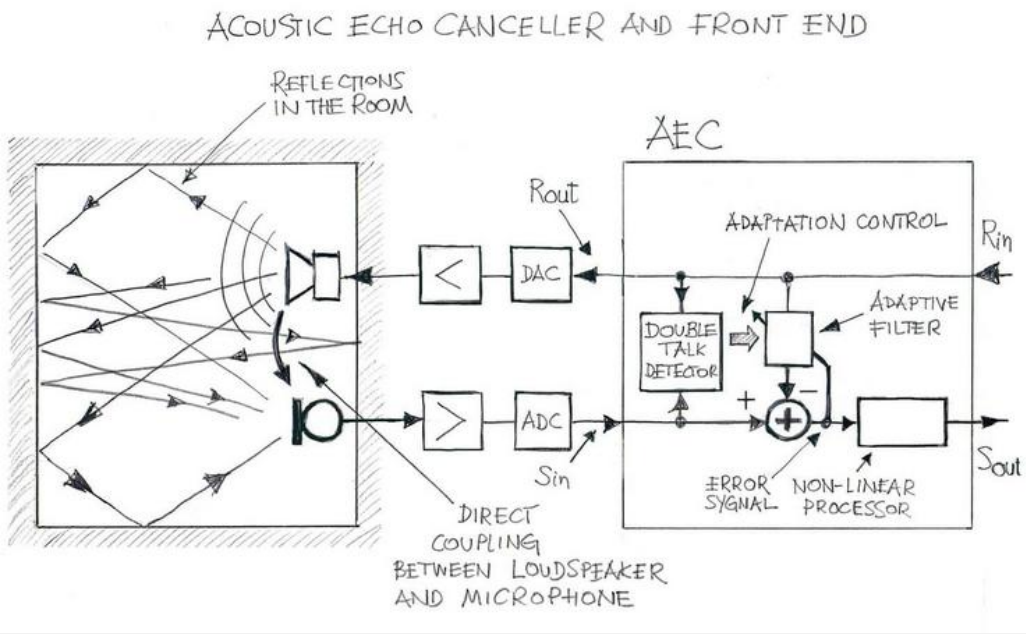
System identification

Example

- An application that fits to this scheme is the echo cancellation
  - u_k = received speech signal (Rin)
  - d_k = signal containing echo (Sin)
  - unknown system = reflections generated in the room
  - adaptive filter predicts echo
  - e_k = signal without echo (Sout)



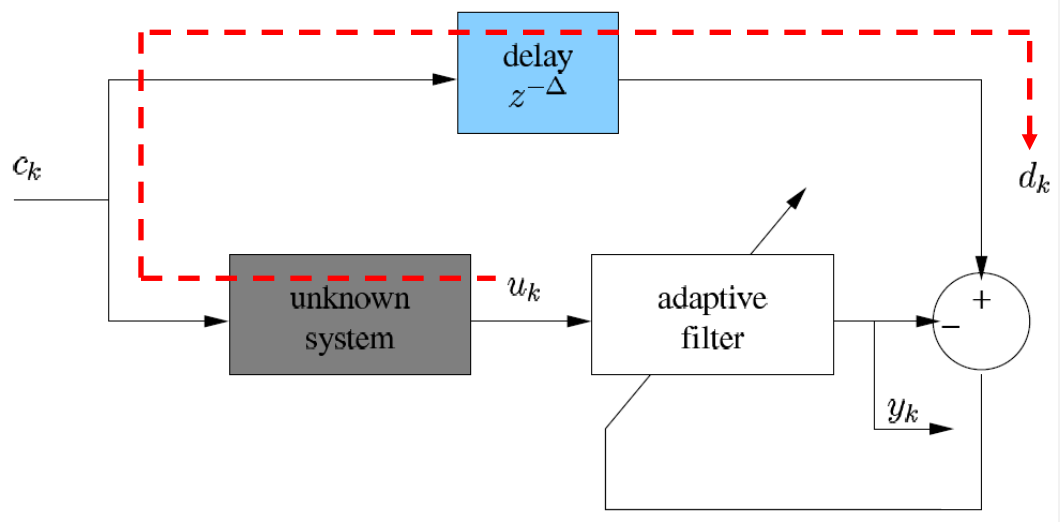
System identification



https://vocal.com/echo-cancellation/acoustic-echo-canceller/

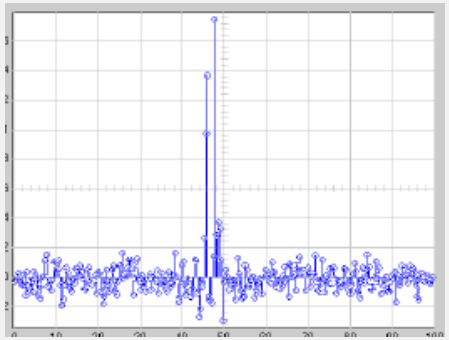
Configurations (3)

- In inverse modelling, the reference input u_k is the output of an unknown system
- The primary input d_k corresponds to the system input: it is delayed
- The adaptive filter should converge to a system model of the path from u_k to d_k. The estimated system is $z^{-\Delta}H^{-1}(z)$
  - $H^{-1}(z)$ is noncausal: the delay element can then be understood as a tool for providing causality so that filter modelling is possible



Inverse modelling (equalization)

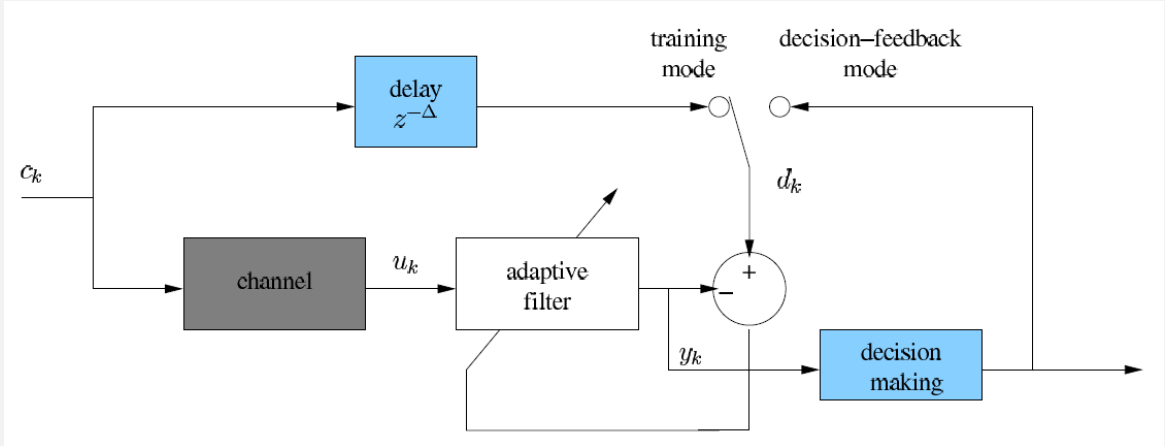
An example of filter's impulse response: change in the delay moves the peak.



Case in Design Task 6

An example of inverse modelling application is the adaptive channel equalization, which is used in modems. The equalizer compensates for the distortion caused by the transmission channel (e.g. multipath propagation), whose properties are largely unknown and can change during transmission.

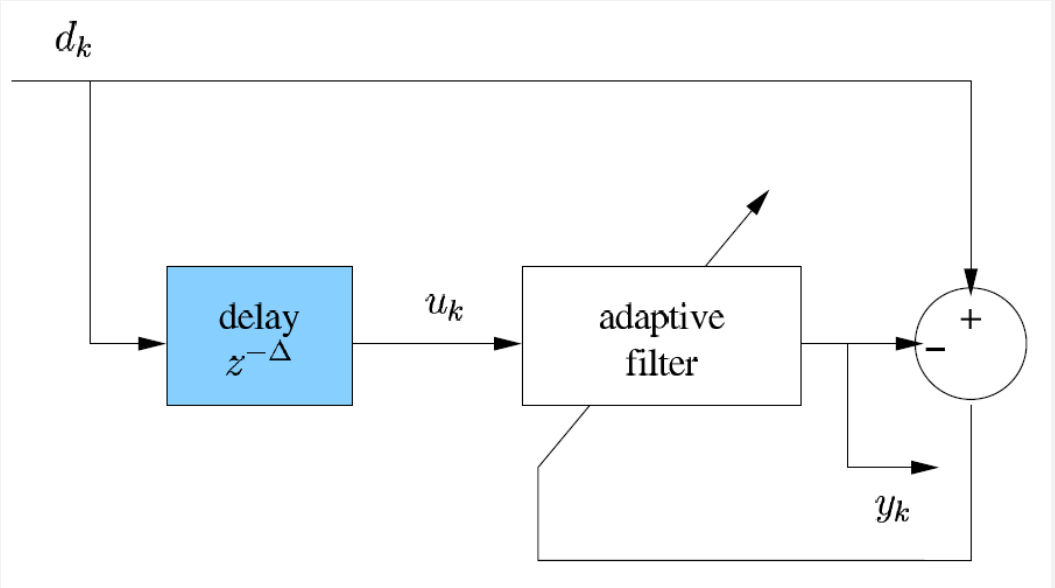
Initially, the channel can be estimated using a training bit pattern (pilot signal). When the system has converged to a working model, the system can switch to the decision-feedback mode. In that mode, adaptation can continue, but now the bit detection results provide the primary input.



Channel equalization

Configurations (4)

- In **adaptive line enhancement**, there is no separate reference input u_k
- The primary input d_k is delayed in order to obtain that filter input
- The requirement is that the signal of interest within d_k is periodic
- The goal is that the filter adapts to this periodicity – the periodic component is enhanced at the output
- Analysis of either the output or filter coefficients (impulse response) by DFT provides information about the periodic signal
- The configuration is used also for **linear prediction** of the input, which is used for example in speech coding

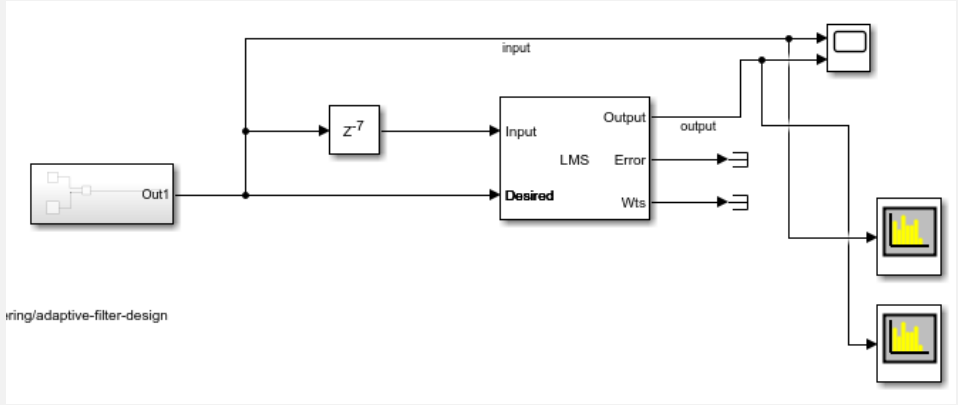


Narrowband signal enhancement
(adaptive line enhancement)

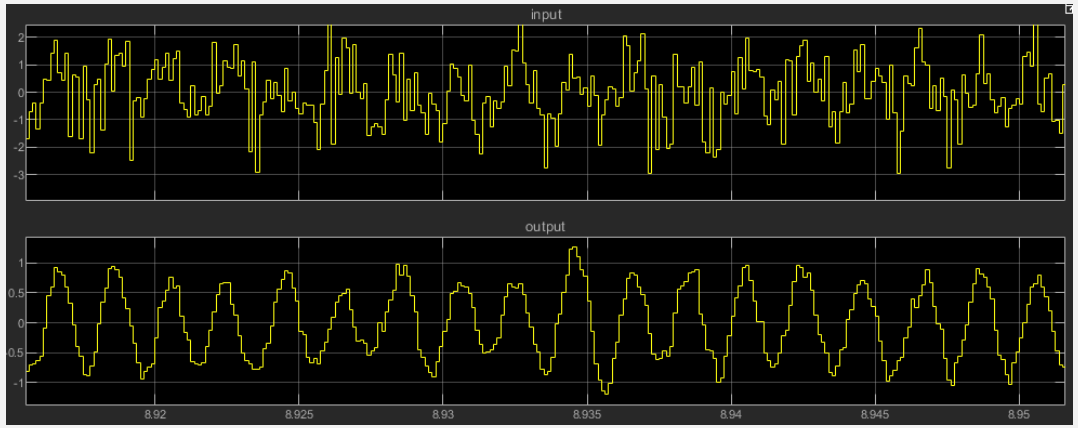
Example. Narrowband signal enhancement

narrowband_signal_enhancement.slx in Moodle

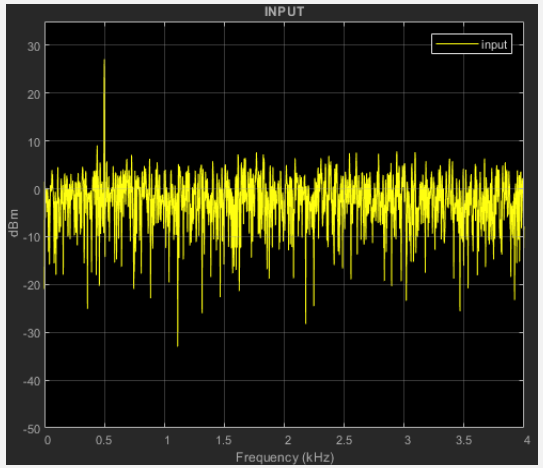
Simulation parameters from
https://www.sciencedirect.com/topics/engineering/adaptive-filter-design

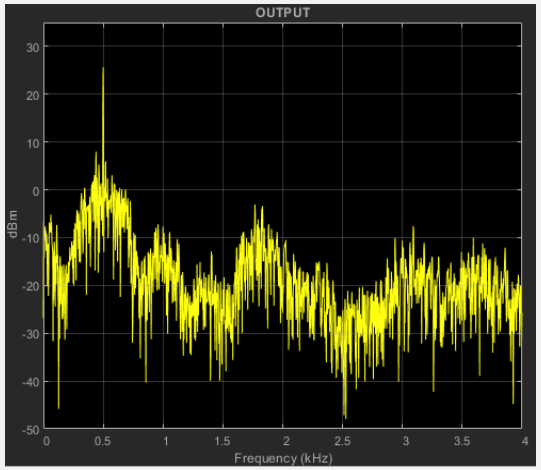


Input: 500 Hz sine wave + noise



Output





reduction in noise level

In next lecture …

- Adaptive algorithms
  - Wiener filtering
  - Least Mean Squares (LMS)
  - Recursive Least Squares (RLS)