Matlab, simulink

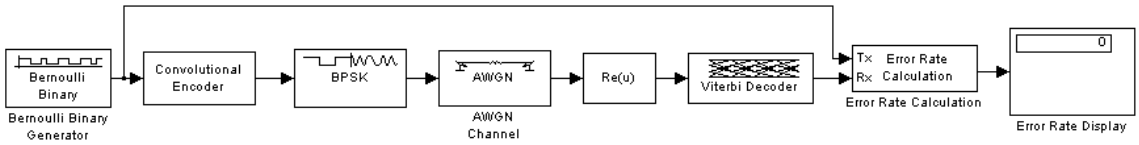
Convolutional Code Model

Satellite Communications
CE74027-3
Tutor: A. El-HELW
Email: elhelw@staffs.ac.uk

Convolutional Code Model

The following model simulates the use of convolutional coding to send a signal through a channel with noise.

Convolutional Code Model



The topics in this section are as follows:

- Building the Convolutional Code Model
- Understanding the Blocks in the Model
- Setting Parameters in the Convolutional Code Model
- Running the Convolutional Code Model

To open a completed version of the model, enter `convdoc` at the MATLAB prompt.

Building the Convolutional Code Model

You can build the convolutional code model by adding blocks to the model shown in the figure Channel Noise Model.

To build the model, follow these steps:

1. Enter `channeldoc`at the MATLAB Help browser to open the channel noise model. Then save the model as `my_conv` in the directory where you keep your work files.
2. Delete the Binary Symmetric Channel block.
3. Drag the following blocks from the Simulink Library Browser into the model window, and connect them as shown in the figure Convolutional Code Model:
    - Convolutional Encoder, from the Convolutional sublibrary of the Error Detection and Correction library
    - BPSK Modulator Baseband, from PM in the Digital Baseband Modulation sublibrary of the Modulation library
    - Complex to Real-Imag, from the Simulink Math Operations library
    - Viterbi Decoder, from the Convolutional sublibrary of the Error Detection and Correction library

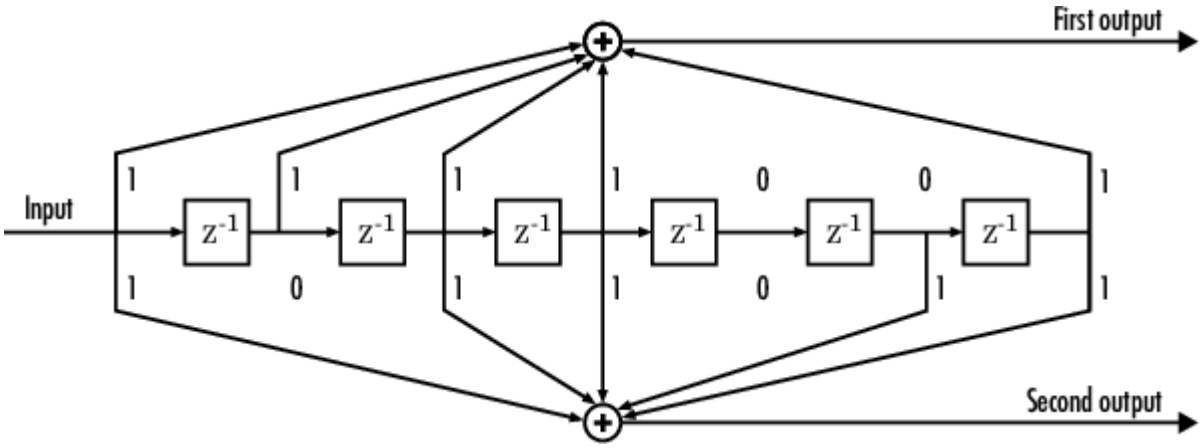
Understanding the Blocks in the Model

The model contains the following blocks.

Convolutional Encoder

The Convolutional Encoder block encodes the signal from the Bernoulli Binary Generator. The example uses the industry standard rate 1/2 convolutional code, with constraint length 7, defined by the following diagram.

Convolutional Encoder Schematic Block Diagram



The encoder structure is described by a pair of binary numbers, having the same length as the code's constraint length, that specify the connections from the delay cells to modulo-2 addition nodes. The binary number for the upper addition node is 1111001. A 1 indicates that the bit in the corresponding delay cell (reading from left to right) is sent to the addition node, and a 0 indicates that the bit is not sent. The binary number for the lower addition node is 1011011. Converting these two binary numbers to octal gives the pair [171,133]. You can enter this pair into the block's dialog by typing `poly2trellis(7, [171 133])` in the field for **Trellis Structure**.

To learn more about the convolutional coding features of the Communications Blockset, see Convolutional Coding in the online Communications Blockset documentation.

Complex to Real-Imag

The Complex to Real-Imag block, labeled Re(u), receives the complex signal and outputs its real part. Since the output BPSK Modulator Baseband block has zero complex part, all of the signal is carried by the real part. You can set this option by selecting `Real` in the **Output** parameter field in the dialog. It is not necessary to demodulate the signal, because the Viterbi Decoder block can accept unquantized inputs.

Viterbi Decoder

The Viterbi Decoder block decodes the signal using the Viterbi algorithm. The **Decision Type** parameter is set to `Unquantized` so that the block can accept real numbers from the Complex to Real-Imag block. The **Traceback depth** parameter, which is set to `96`, is the number of branches in the trellis that the block uses to construct each traceback path. This produces a delay of `96` between the input and output of the block. For more information on delays, see Finding the Delay in a Model.

For an example of a convolutional coding model that uses soft-decision decoding, see Example: Soft-Decision Decoding in the online Communications Blockset documentation.

Setting Parameters in the Convolutional Code Model

To set parameters in the convolutional code model, do the following:

1. Double-click the Bernoulli Binary Generator block and check the box next to **Frame-based outputs** in the block's dialog.

2. Double-click the AWGN Channel block and make the following changes to the default parameters in the block's dialog:
    - Set **Es/No** to `-1`.
    - Set **Symbol period** to `1/2`. Since the code rate is 1/2, this setting causes the block to produce the same amount of noise per channel symbol as it would without channel coding. For more information, see Verifying the Symbol Period.
3. Double-click the Error Rate Calculation block and make the following changes to the default parameters in the block's dialog:
    - Set **Receive delay** to `96`. The Viterbi Decoder block creates a delay of 96, due to its **Traceback depth** setting.
    - Check the box next to **Stop simulation**.
    - Set **Target number of errors** to `100`.

Running the Convolutional Code Model

When you run the model, you observe an error rate of approximately .003.