

《通信与网络》 实验五

BSC信道编码实验

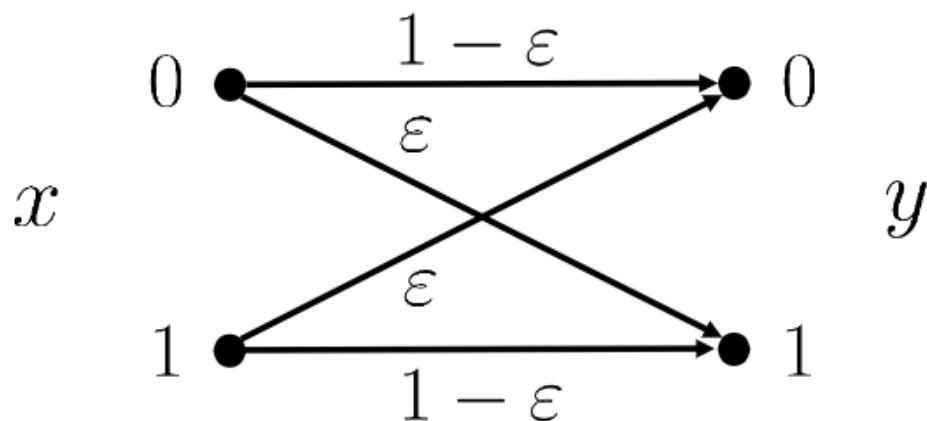
2022年11月

目录

- 信道编码回顾
- Simulink及基本操作方法介绍
- 实验内容和流程

一、信道编码回顾

对称二元信道 (BSC)



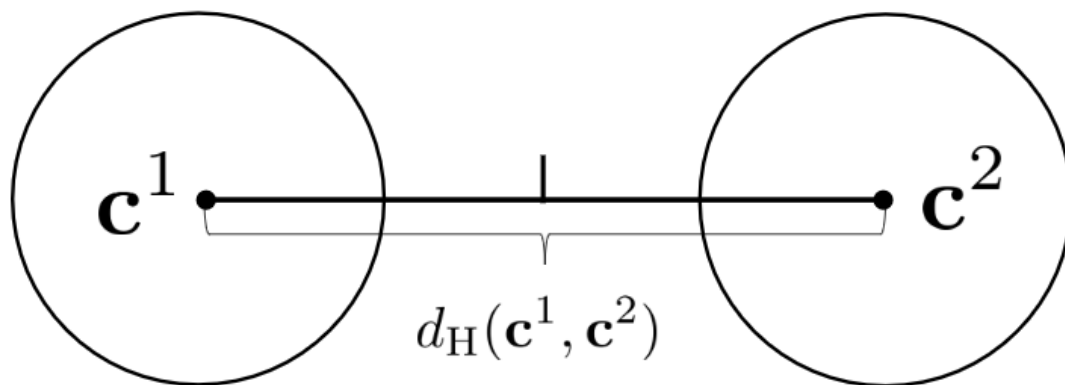
二元：输入、输出均为二元量（符号, 0或1）

对称：“0”错成“1”的概率=“1”错成“0”的概率

差错概率： $\varepsilon = \Pr\{y = 1|x = 0\}$
 $= \Pr\{y = 0|x = 1\}$

前向纠错编码(FEC)

- 适用于无反馈的信道，具有高实时性
- 最小Hamming距离判决 $\arg \min_{\hat{\mathbf{c}} \in \mathcal{X}} d_H(\mathbf{r}, \hat{\mathbf{c}})$



显然，只要不超过 k 个差错（差错位 $\leq k$ ）就一定不会错判，此时纠错能力为 $k = \frac{d_H(\mathbf{c}^1, \mathbf{c}^2) - 1}{2}$ 个bit

前向纠错编码(FEC)

- 最小Hamming距离判决 $\arg \min_{\hat{\mathbf{c}} \in \mathcal{X}} d_H(\mathbf{r}, \hat{\mathbf{c}})$

- 码字集合的最小Hamming码距

$$d_H^{\min} = \min_{\mathbf{c}^i, \mathbf{c}^j \in \mathcal{X}} d_H(\mathbf{c}^i, \mathbf{c}^j)$$

对任给的码字 $\mathbf{c}^i \in \mathcal{X}$ 只要差错不超过 $\left\lfloor \frac{d_H^{\min} - 1}{2} \right\rfloor$ 位, 则一定能纠错 (正确判决)

- 最小Hamming码距=3, 4 \rightarrow 可纠错1位
- 最小Hamming码距=5, 6 \rightarrow 可纠错2位

重复码 (Repetition Code)

- $(n, 1)$ 重复码：重复消息 n 次 $d_H^{\min} = n$

- 消息 ... 0 1 0 ...

- 编码结果 ... 0 ... 0 1 ... 1 0 ... 0 ...
 $\underbrace{\hspace{1.5cm}}_{n\uparrow}$ $\underbrace{\hspace{1.5cm}}_{n\uparrow}$ $\underbrace{\hspace{1.5cm}}_{n\uparrow}$

- 译码

- 最小Hamming距离判决：比较接收到0和1的个数

- 易于实现，但编码效率很低

Hamming码 (Hamming Code)

- (n, k) Hamming码 $d_H^{\min} = 3$
 - 满足 $k = 2^m - m - 1, n = 2^m - 1, m \geq 3$

m	k	n	$R = k/n$
3	4	7	0.57
4	11	15	0.73
5	26	31	0.84
6	57	63	0.90
7	120	127	0.94
8	247	255	0.97
9	502	511	0.98
10	1013	1023	0.99

- 有效利用了最小码距，最大化纠错能力，效率高

Hamming码 (Hamming Code)

- 例：(7, 4) Hamming码

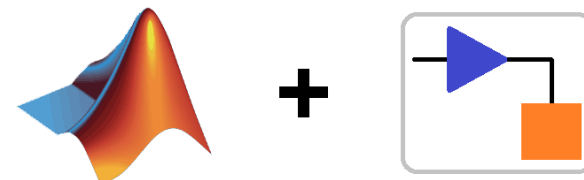
$$\mathbf{H} = \begin{bmatrix} 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 & 1 \end{bmatrix} \quad \mathbf{G} = [\mathbf{I}; \tilde{\mathbf{H}}^T] = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 \end{bmatrix}$$

- 消息： $\mathbf{d} = [0, 0, 1, 0]$
- 编码： $\mathbf{c} = \mathbf{d}\mathbf{G} = [0, 0, 1, 0, 1, 0, 1]$
- 信道： $\mathbf{e} = [1, 0, 0, 0, 0, 0, 0]$
- 接收： $\mathbf{r} = \mathbf{c} + \mathbf{e}$
 $\mathbf{r} = [1, 0, 1, 0, 1, 0, 1]$
- 解码： $\mathbf{s} = \mathbf{r}\mathbf{H}^T$
 $\mathbf{s} = [1, 1, 1] = \mathbf{h}_1^T \rightarrow$ 标识第一位错

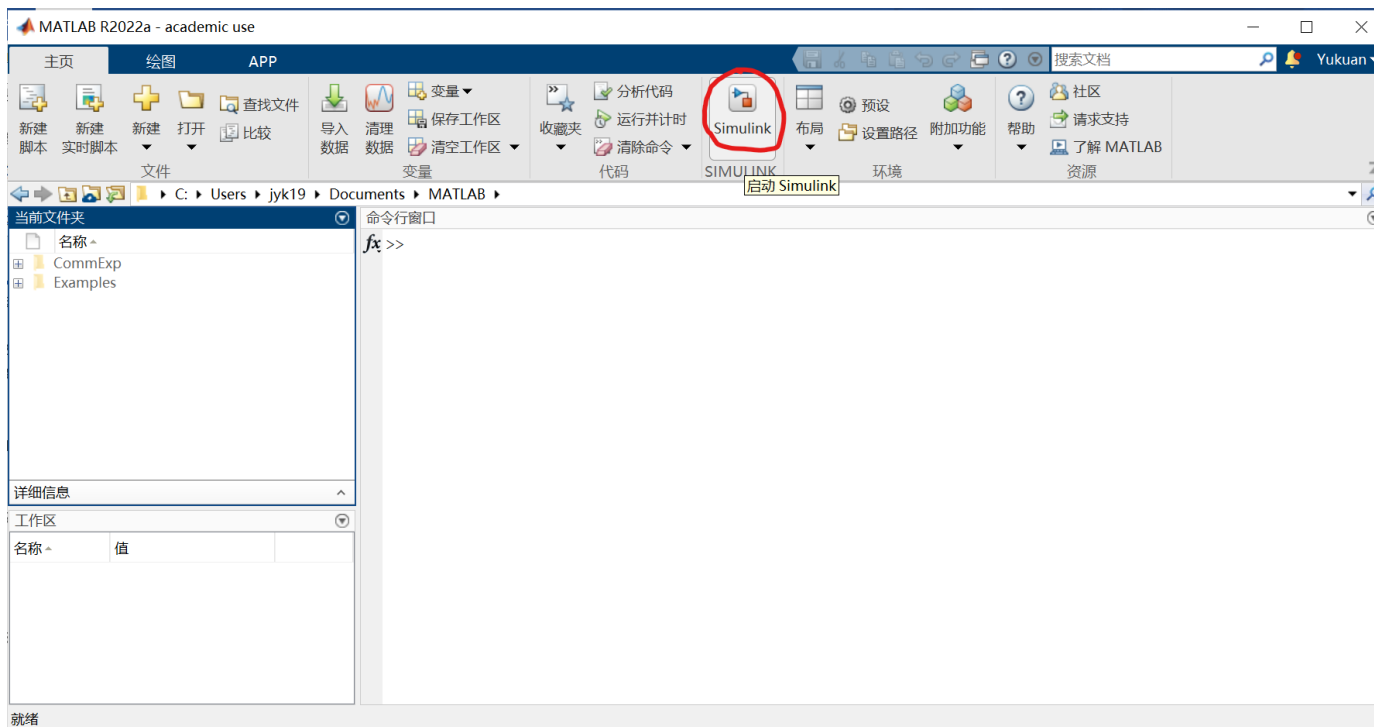
二、Simulink及基本 操作方法介绍

实验环境

- Matlab 2022a + Simulink
 - Communications Toolbox
 - DSP System Toolbox



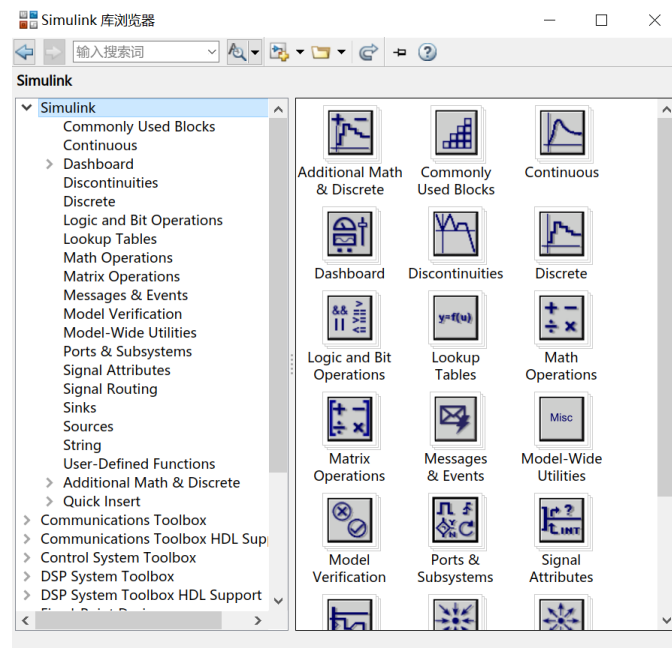
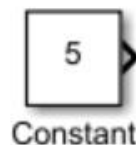
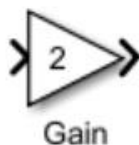
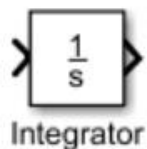
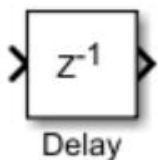
MATLAB
SIMULINK®



Simulink基本介绍

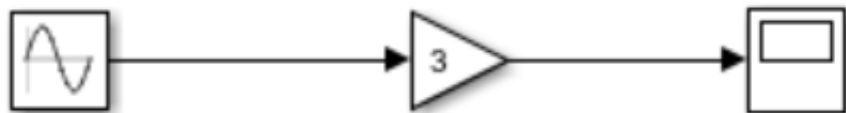
- **模块：基本建模结构**

- 可以从内置的 Simulink 库中添加模块以执行特定操作，也可以创建自定义模块。
- 模块之间的连接接口称为端口。
- 模块从输入端口接受信号，执行运算，并在输出端口输出信号。



Simulink基本介绍

- **信号线：连接模块，使数据在模块间传输**
 - 信号：是随时间变化的量，在所有时间点（连续）或指定的时间点（离散）都有对应的值。
 - 信号线连接模块端口，信号从模块的输出端口流向另一个模块的输入端口。常见的信号是数值或矢量。



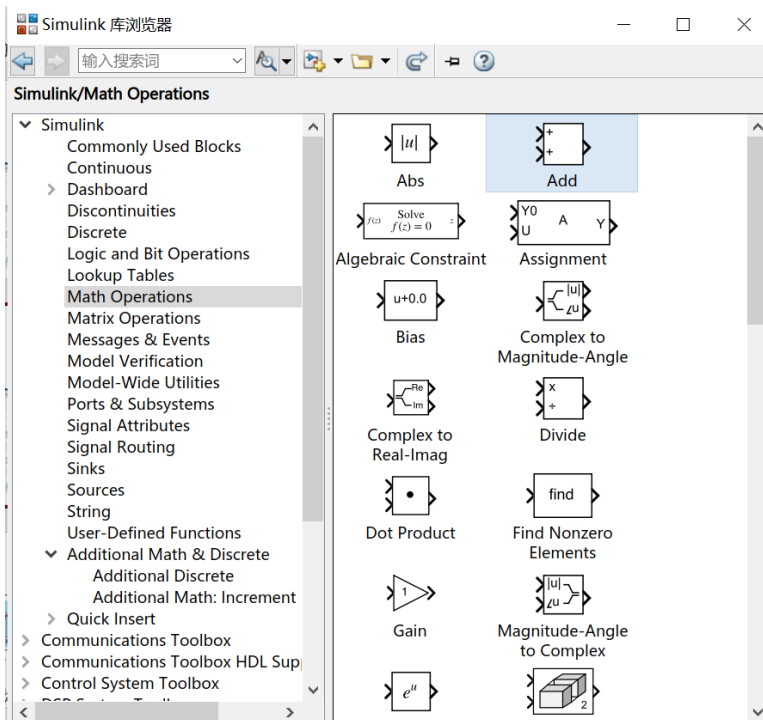
Simulink操作方法

• 新建模型

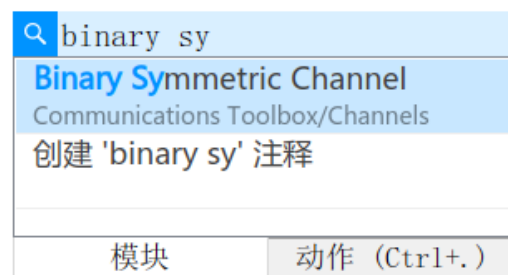


Simulink操作方法

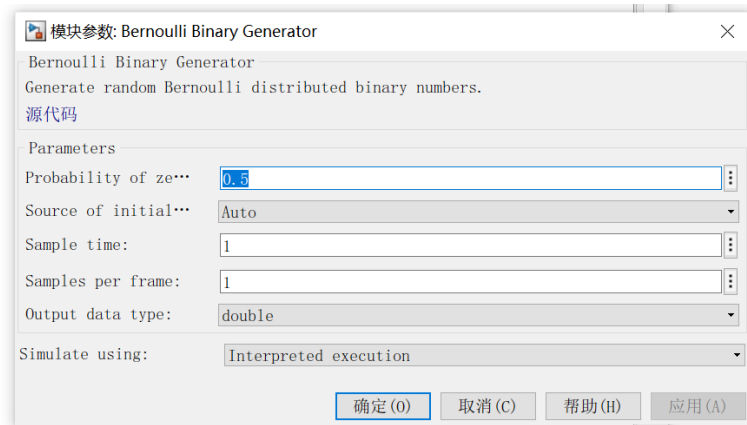
- 添加模块
- 从库浏览器拖入



- 双击空白处搜索



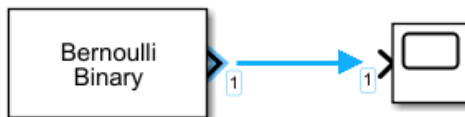
- 双击模块修改其属性



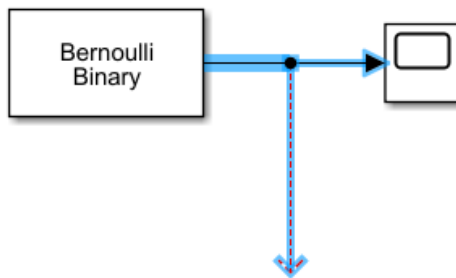
Simulink操作方法

- 添加信号线

- 点击一个端口，所有合适的连接都将突出显示，点击第二个端口以创建连接



- 另一种方式是点击端口并拖动到第二个端口
- 按住Ctrl可以从信号线上创建连接分支



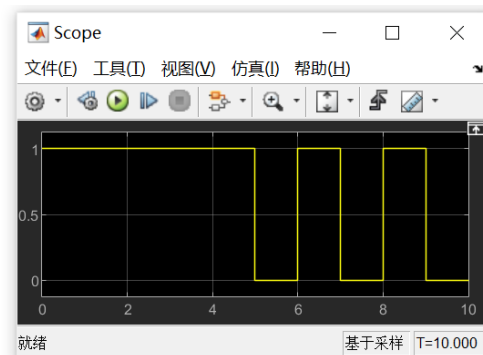
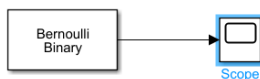
Simulink操作方法

- 运行仿真



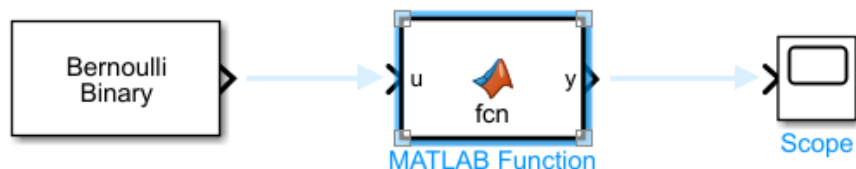
- 观察信号

- 在模型中加入Simulink-Sinks-Scope模块（示波器），连接到想要观察的信号，运行仿真后双击Scope即可观察信号。



Simulink操作方法

- 嵌入自定义Matlab函数
 - 在模型中加入Simulink-User Defined Functions-Matlab Function，双击模块即可编辑编写Matlab函数来自定义模块。



Simulink操作方法

- 模型资源管理器



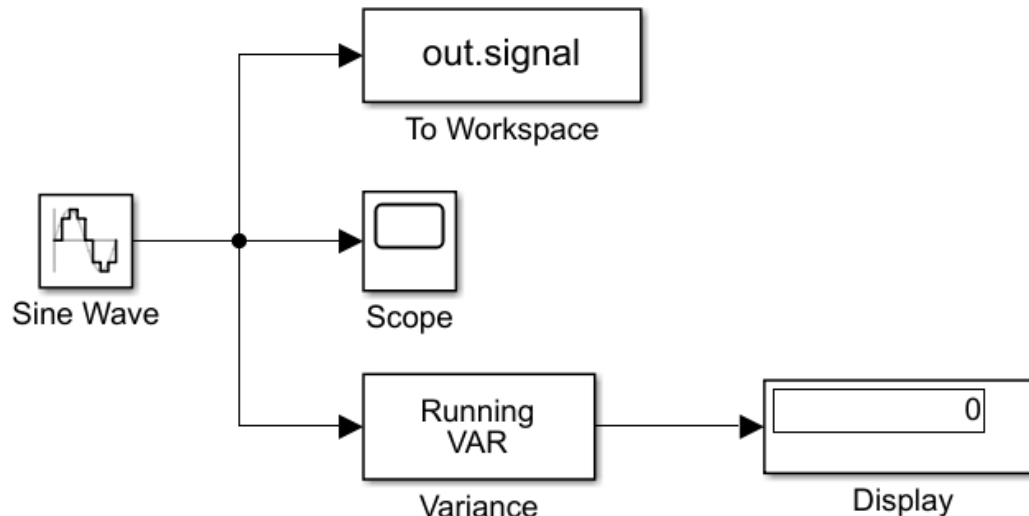
- 可集中修改模型中模块的参数和属性设置



- 对自定义的Matlab函数模块也可以修改变量的属性
 - 本次实验中需将某一变量设为不可变参数

Simulink操作方法

- 从Matlab运行Simulink模型（预习指导书）
 - 例：观察并导出正弦波信号
 - 在模型中添加To Workspace模块，连接到想要导出的信号并在其设置中命名为signal，保存格式为“数组”。

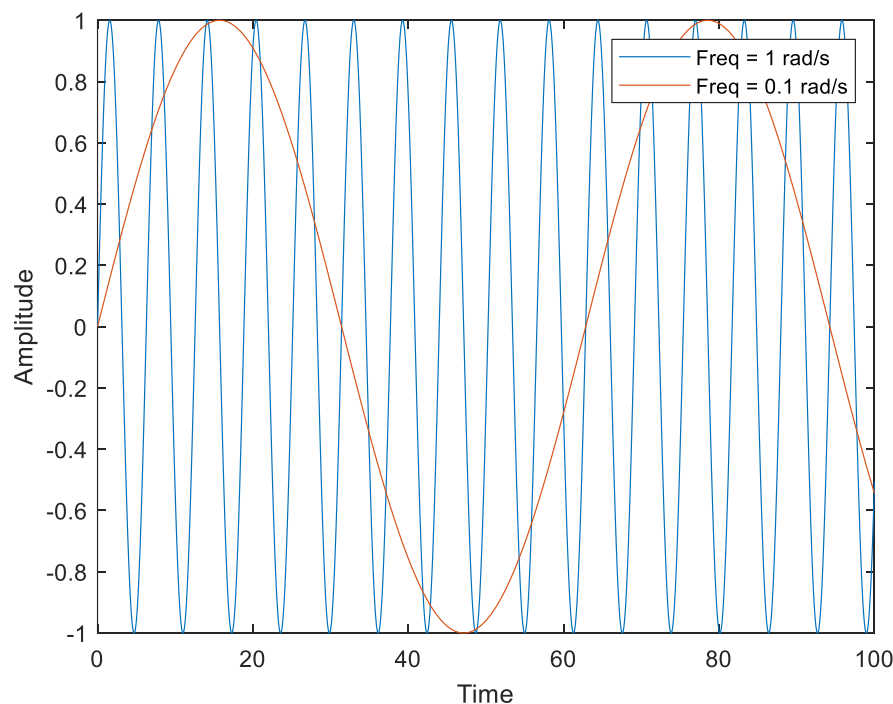


Simulink操作方法

- 从Matlab运行Simulink模型（预习指导书）

- 在Matlab中调用模型仿真，从输出的结构体simOut中提取默认的tout（时间）数组和To Workspace模块指定的signal数组

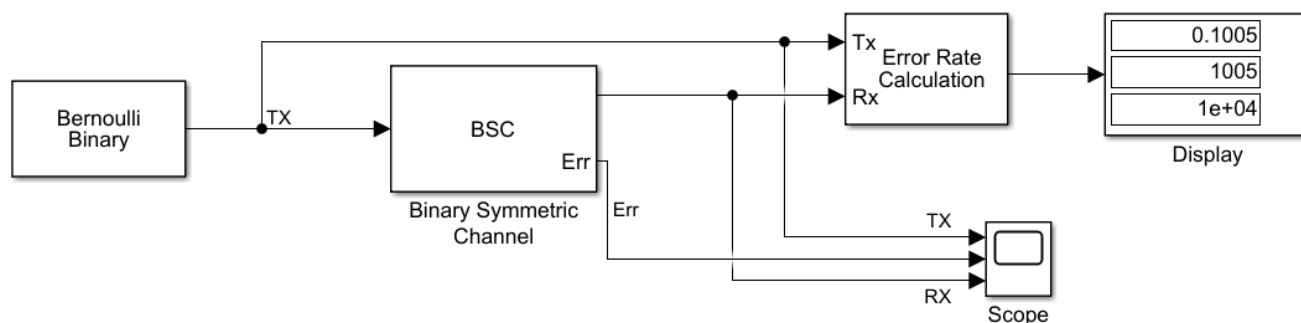
```
1  for f = [1 0.1]
2      open_system('sinewave.slx');
3      simOut = sim('sinewave');
4      save_system;
5      close_system;
6      t = simOut.tout;
7      signal = simOut.signal;
8      plot(t,signal); hold on;
9  end
```



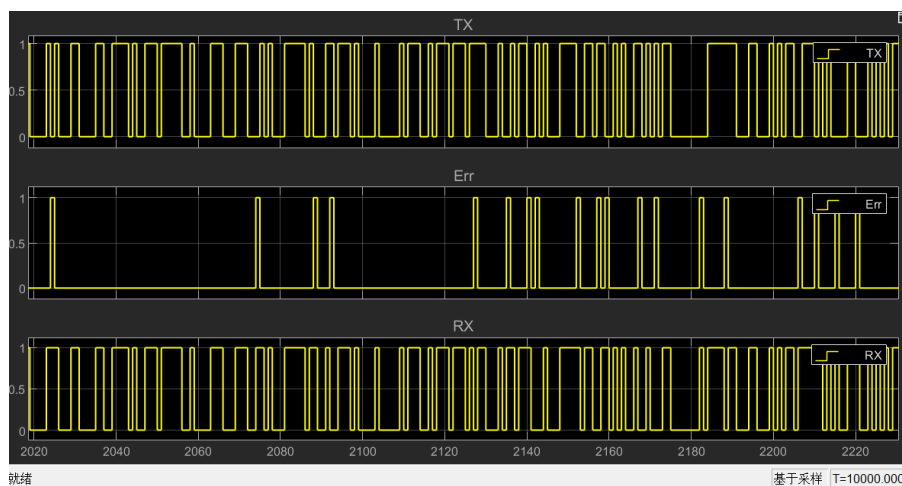
三、实验内容介绍

1. BSC差错信道

- 搭建模型

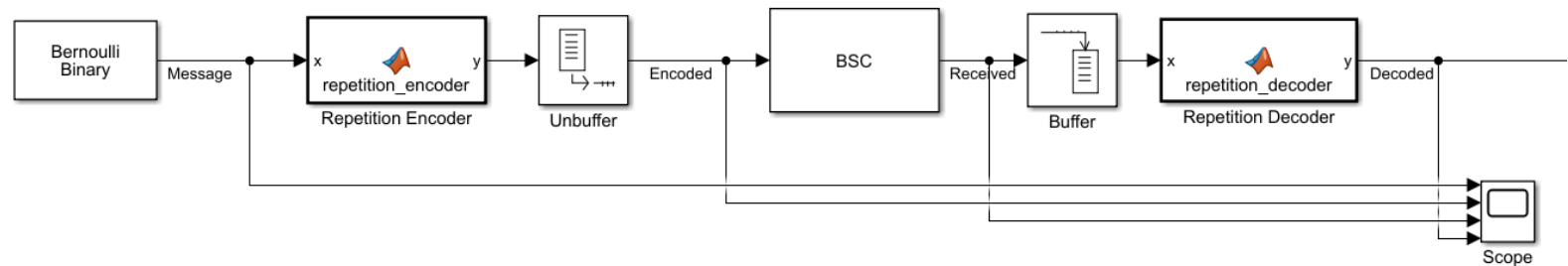


- 观察BSC信道对二元数字信号的影响



2. 重复码

- 搭建模型



- Buffer/Unbuffer用于数据并行/串行转换

- 编写自定义Matlab函数实现 $(n, 1)$ 重复码

```
function y = repetition_encoder(x, n)
```

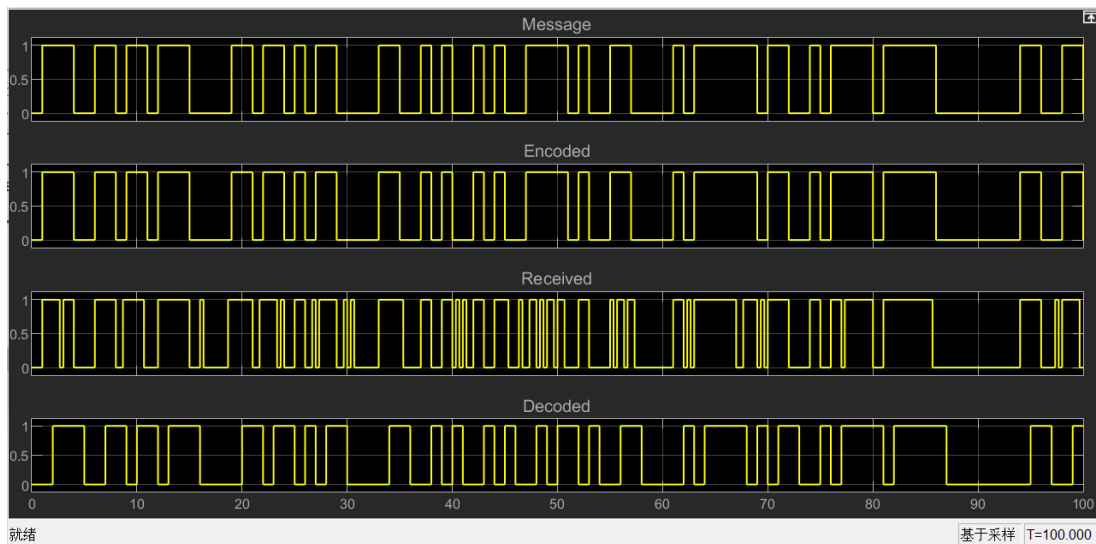
```
% TODO: y = ?
```

```
function y = repetition_decoder(x)
```

```
% TODO: y = ?
```


2. 重复码

- 观察延时并在误比特率计算模块设置补偿

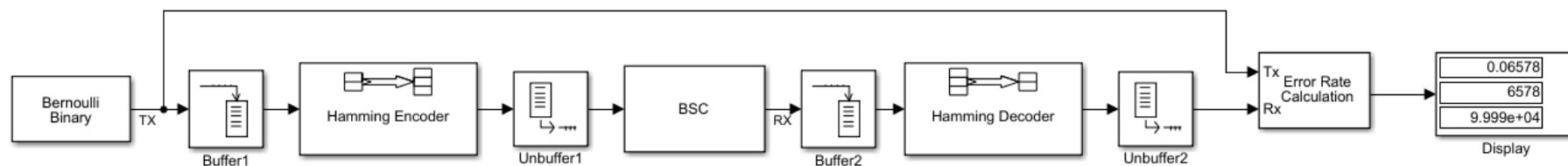


- 改变信道错误率，记录重复码的误比特率
 - (3,1) 重复编码、(7,1) 重复编码

信道错误率	0.3	0.1	0.03	0.01	0.003
误比特率					

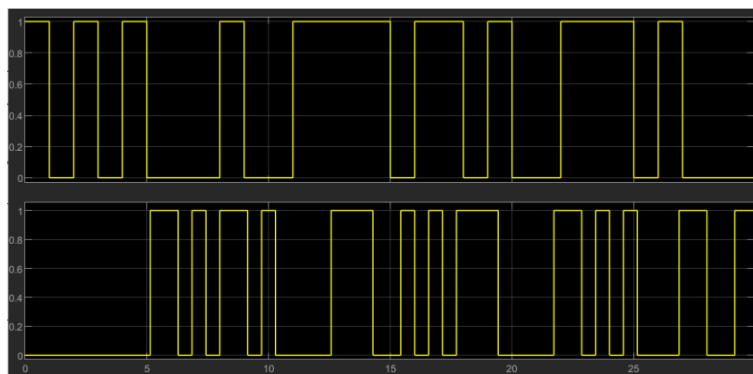
3. Hamming码

- 搭建模型



- 观察延时并设置补偿

- 观察信号并推断生成矩阵 G 和校验矩阵 H^T



3. Hamming码

- 改变信道错误率，记录Hamming码的误比特率
 - (7,4) Hamming码、(31,26) Hamming码
- (选做) 记录Hamming码的误块率
 - 每个块即一个分组，有k个比特
 - 误块率定义为解码错误的块占传输的总块数的比例

信道错误率	0.3	0.1	0.03	0.01	0.003
误比特率					
误块率					

4. 绘制误比特率曲线

- 样例：exp5.m（重复码）

- 根据理论计算并填写 $(n, 1)$ 重复码的错误概率估计值
- 样例代码遍历仿真了不同重复位数 $n=\{3, 4, 5\}$ 和不同的信道错误概率 e_{prob} 下的误比特率并绘制曲线
- 绘制实验数据和理论值的图像并分析其对应关系

```
1  %% Repetition Code
2  open_system('repetition.slx');
3  n_list = [3 4 5];
4  eprob_list = logspace(-3,-0.5,10);
5  ber_rep = zeros(length(n_list), length(eprob_list));
6  ber_rep_th = zeros(length(n_list), length(eprob_list));
7
8  for i = 1:length(n_list)
9      for j = 1:length(eprob_list)
10         n = n_list(i);
11         eprob = eprob_list(j);
12         simOut = sim('repetition');
13         ber_rep(i,j) = simOut.ErrorStat(1);
14         ber_rep_th(i,j) = % TODO: Theoretical BER
15     end
16 end
17 save_system;
18 close_system;
```

4. 绘制误比特率曲线

- **Hamming码**

- 根据理论计算并填写Hamming码的错误概率估计值
- 另存修改exp5.m以绘制(7, 4), (15, 11), (31, 26) Hamming码的仿真和理论差错概率曲线
- 分析理论和仿真的对应关系

注意事项

- 注意查阅指导书中的操作方法
- 提交实验报告至网络学堂
- 实验报告需包括代码、实验流程记录、思考题回答