⌘ S

```
begin
    using StatsBase
    using Distributions
    using InformationMeasures
    using Plots
    using PlutoUI
end

```

0.27

```
#Creating some sliders so we can change p (noise probability) and N (message length)
@bind p Slider(0:0.01:0.5, default = 0.1, show_value = true)
```

20000

```
@bind N Slider(1000:1000:20000, default = 10000, show_value = true)
```

BitVector: [true, true, false, false, true, true, true, true, false, false, true, false, fals

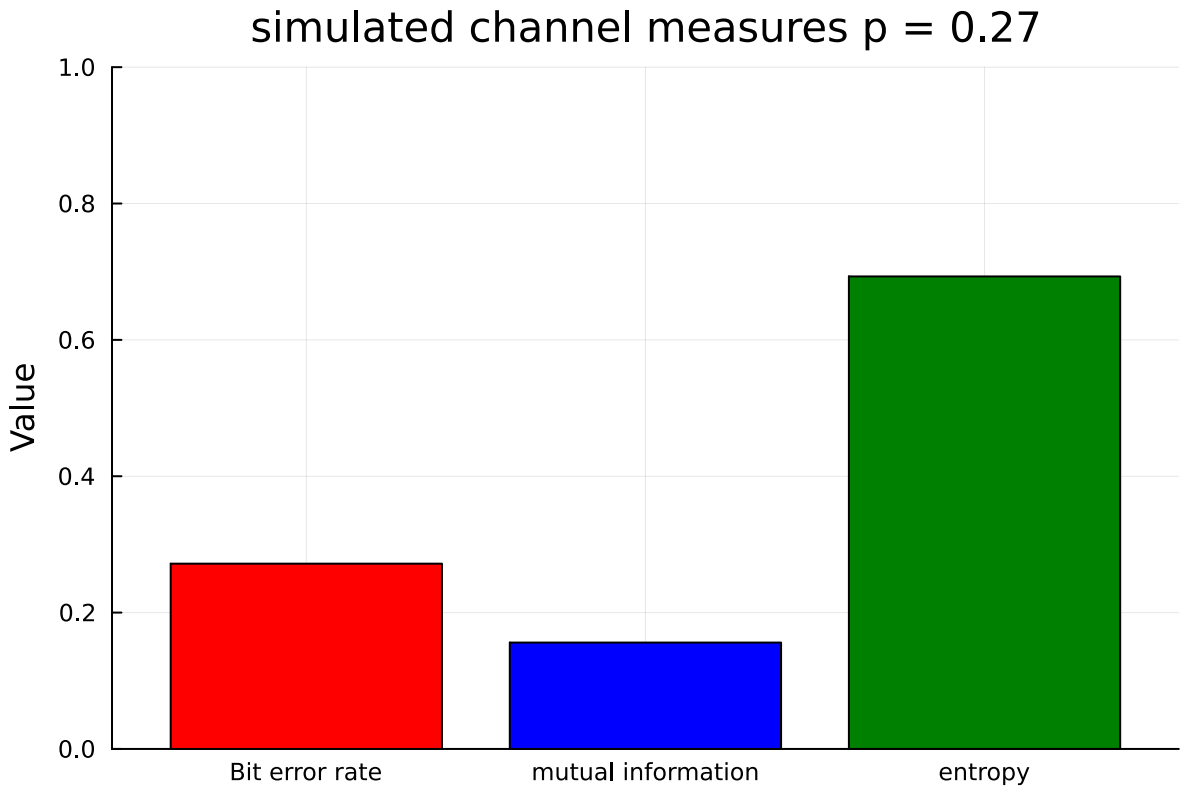
```
begin
    #Generating a random binary message and putting the message through a noisy
    channel
    message = rand(Bool,N)

    function binary_symmetric_channel(bits,p)
        flips = rand(Bernoulli(p), length(bits))
        return xor.(bits, flips)
    end

    received = binary_symmetric_channel(message, p)
end
```

0.6930463571714478

```
begin
    #Some information measures
    bit_error_rate = sum(message .!= received) / N
    mutual_info = get_mutual_information(message,received)
    received_entropy = entropy(Float64[mean(received), 1 - mean(received)])
end
```



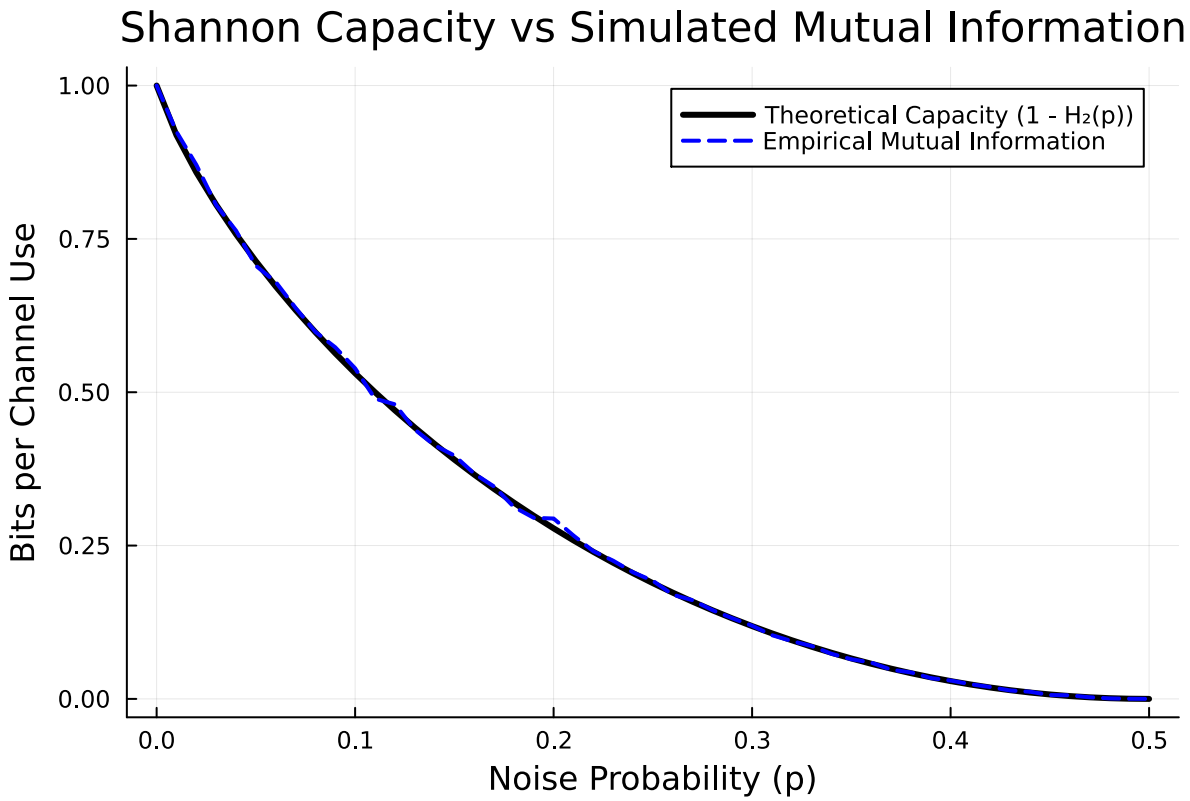

```julia
#Plotting things
bar(
    ["Bit error rate", "mutual information", "entropy"],
    [bit_error_rate, mutual_info, received_entropy],
    legend = false,
    color = [:red, :blue, :green],
    ylim = (0,1),
    title = "simulated channel measures p = $(round(p,digits=2))",
    ylabel = "Value"
)
```

```julia
begin
    #Comparing with Shannon's Thoeretical Channel Capacity

    function binary_entropy(p)
        if p == 0 || p == 1
            return 0.0
        else
            return -p*log2(p) - (1-p)*log2(1-p)
        end
    end

    ps = 0:0.01:0.5
    capacity = [1 - binary_entropy(p) for p in ps]

    mutual_infos = Float64[]
    for prob in ps
        received_temp = binary_symmetric_channel(message,prob)
        push!(mutual_infos, get_mutual_information(message, received_temp))
    end
end
```




```
begin
    #Combining the plots
    plot(
    ps, capacity,
    label="Theoretical Capacity (1 - H₂(p))",
    lw=3, color=:black,
    xlabel="Noise Probability (p)",
    ylabel="Bits per Channel Use",
    title="Shannon Capacity vs Simulated Mutual Information",
    legend=:topright
    )

    plot!(
    ps, mutual_infos,
    label="Empirical Mutual Information",
    lw=2, color=:blue, ls=:dash
    )

end
```