```
using Pkg
```

```
using Plots
```

```
using Soss
```

Using Soss.jl to find the optimal price

We are goint to model the relationship between quantity and price using the equation: Q = aP^c. The priors of **a** and **c** are modeled as a *cauchy* distribution and Q is modeled as a *poisson* distribution.

```
md"### Using Soss.jl to find the optimal price
We are goint to model the relationship between quantity and price using the equation:
Q = a$$P^c$$.
The priors of **a** and **c** are modeled as a *cauchy* distribution and Q is modeled as a
*poisson* distribution."
```

```
m = @model begin
            loga ~ Cauchy()
            c ~ Cauchy()
            logμ0 = loga .+ c * log.(p0)
            μ0 = exp.(logμ0)
            qval ~ For(eachindex(μ0)) do j
                    Poisson(μ0[j])
                end
        end
```

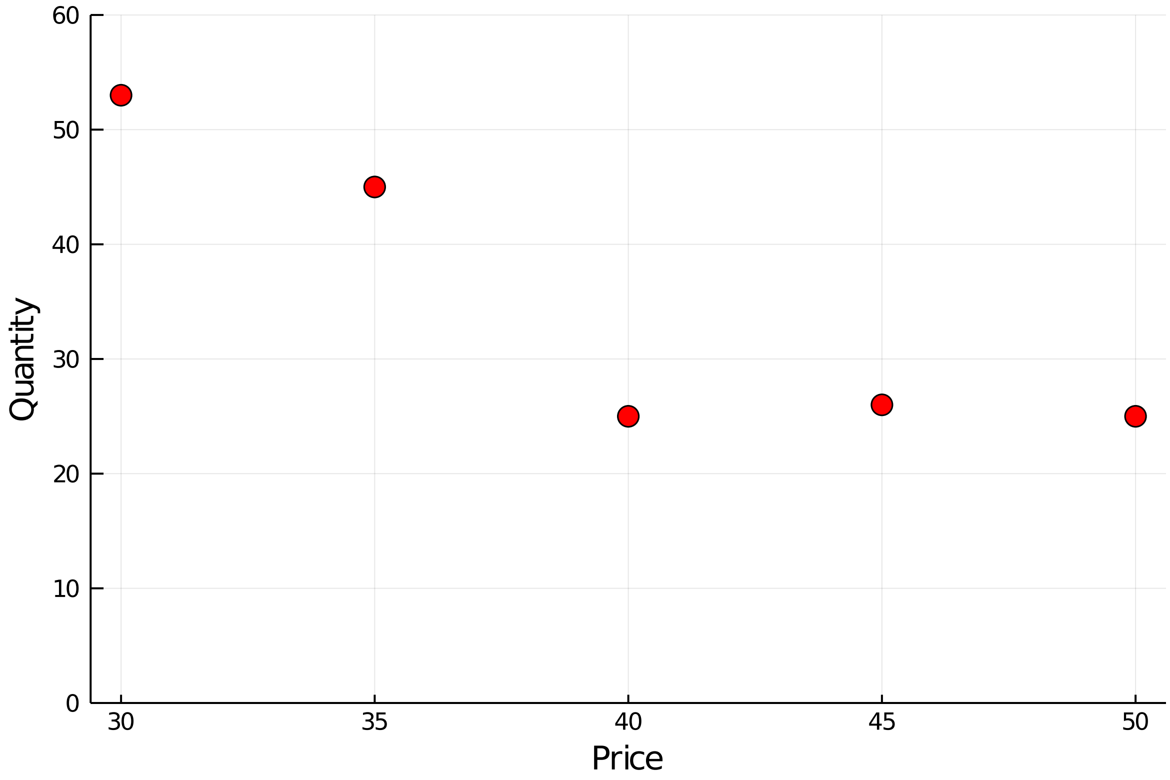
```
m = @model begin
    loga ~ Cauchy()
    c ~ Cauchy()
    logμ0 = loga .+ c*log.(p0)
    μ0 = exp.(logμ0)
    qval ~ For(eachindex(μ0)) do j
        Poisson(μ0[j])
    end
end
```

Now we plot the data

```
md"Now we plot the data"
```

```
Int64[30,  35,  40,  45,  50]
```

```
begin
    Quantity = [53, 45, 25, 26, 25]
    Price = [30, 35, 40, 45, 50]
end
```




```
begin
scatter(Price, Quantity, markersize=6, color="red", ylim=(0,60), legend=false)
xlabel!("Price")
ylabel!("Quantity")
end
```

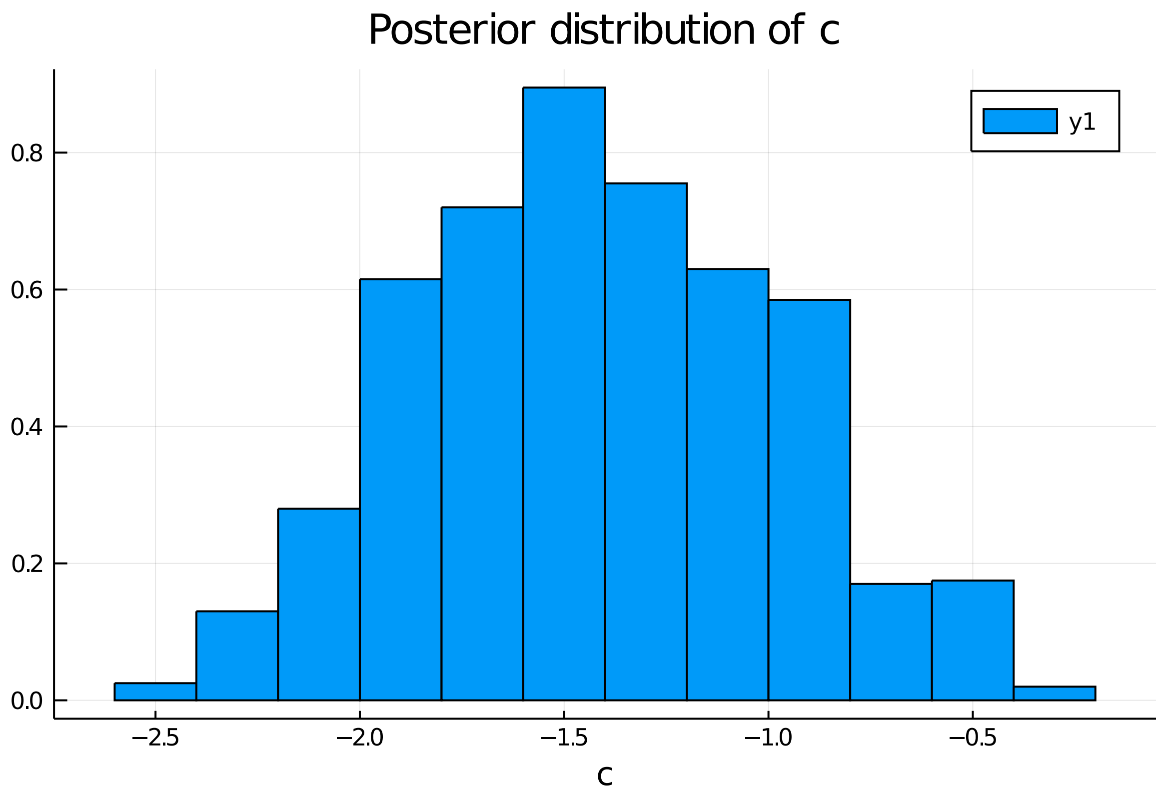
Now, we estimate the posterior and sample from it.

```
md"Now, we estimate the posterior and sample from it."
```

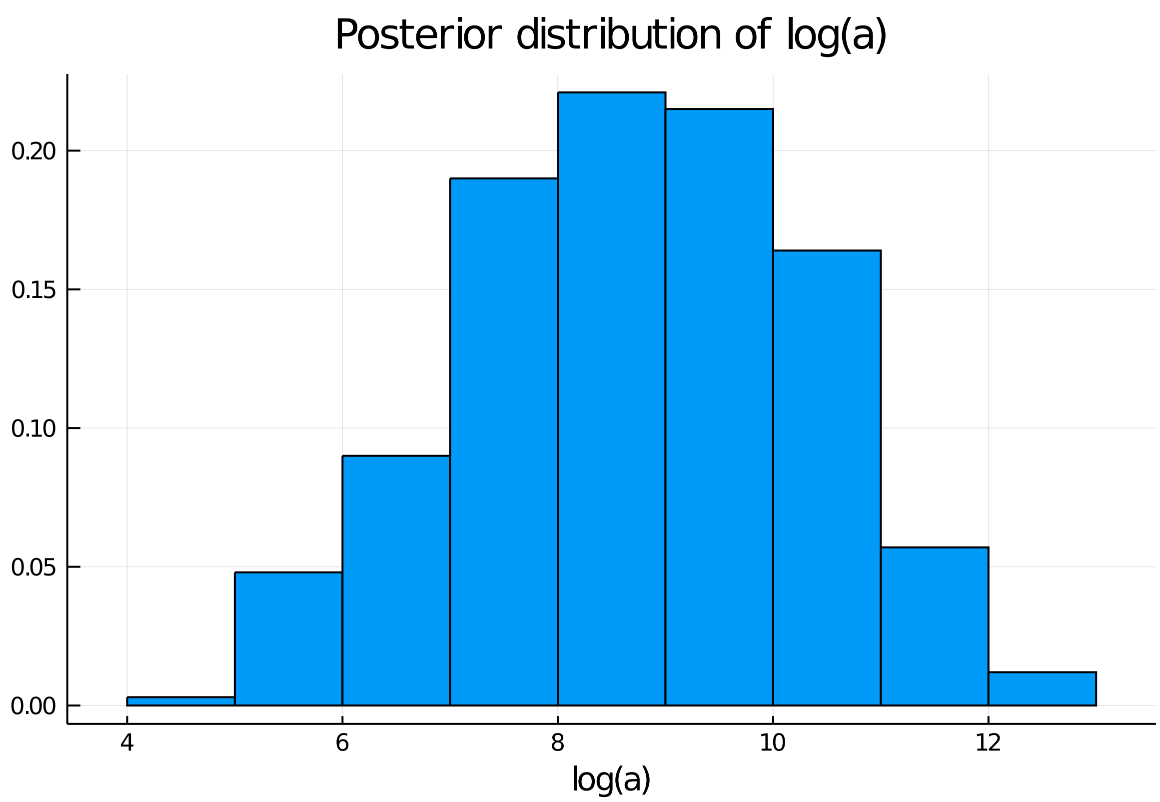
```
post = dynamicHMC(m(), (p0=Price,qval=Quantity,));
```

```
Float64[-1.45943, -1.04506, -1.54466, -1.52799, -1.52749, -1.53325, -1.53
```

```
begin
    loga_ = [vec.loga for vec in post]
    c_ = [vec.c for vec in post]
end
```




```
begin
histogram(c_, normed=true, bins=15)
title!("Posterior distribution of c")
xlabel!("c")
end
```

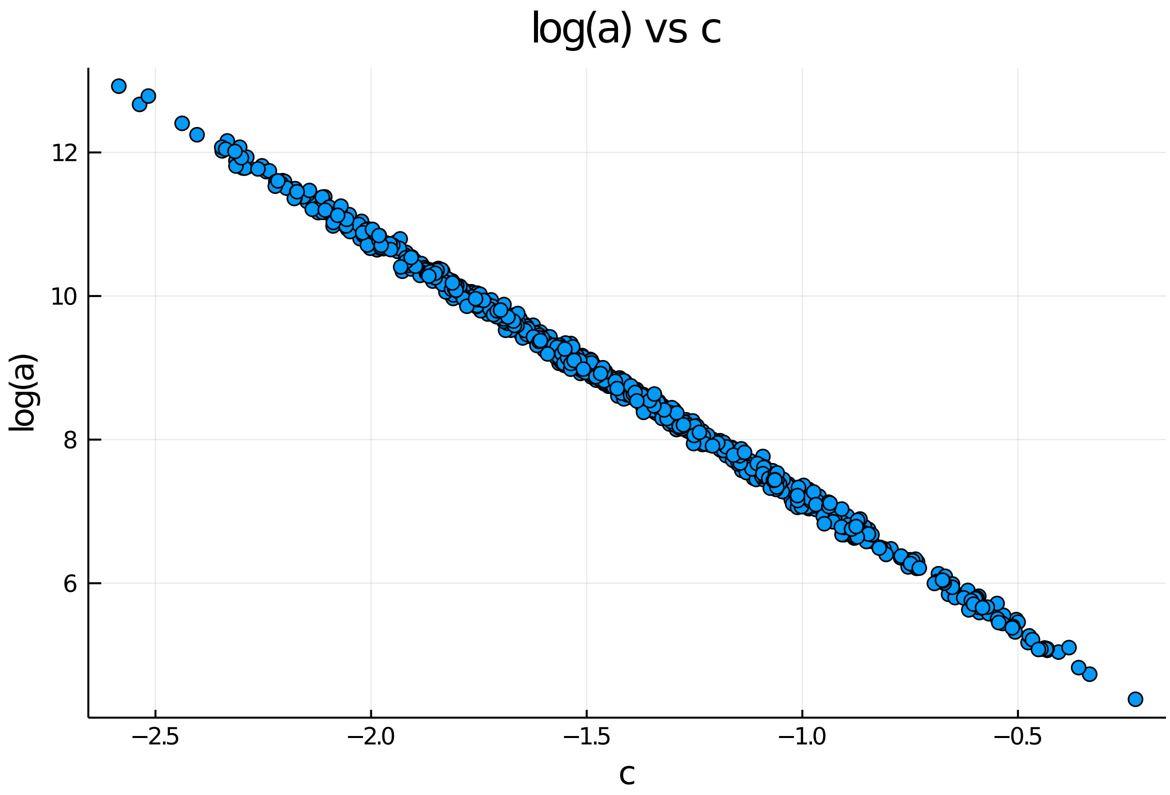



```
begin
```

```
histogram(loga_, normed=true, legend=false, bins=15)
xlabel!("log(a)")
title!("Posterior distribution of log(a)")
end
```

If we plot c vs log(a), we notice that the two variables have multicollinearity.

```
md"If we plot c vs log(a), we notice that the two variables have multicollinearity."
```




```
begin
scatter(c_, loga_, legend=false)
title!("log(a) vs c")
xlabel!("c")
ylabel!("log(a)")
end
```

Now we reparametrize the model to fix the multicollinearity problem, subtracting the mean of *log(p)*. We rename our model variables as β and α.

```
md" Now we reparametrize the model to fix the multicollinearity problem, subtracting the
mean of *log(p)*. We rename our model variables as β and α."
```

```
m2 = @model begin
         β ~ Cauchy()
         α ~ Cauchy()
         logμ0_ = α .+ β * (log.(p0) .- mean(log.(p0)))
         μ0_ = exp.(logμ0_)
         qval ~ For(eachindex(μ0_)) do j
                 Poisson(μ0_[j])
             end
     end
```

```
m2 = @model begin
    α ~ Cauchy()
    β ~ Cauchy()
    logμ0_ = α .+ β*(log.(p0) .- mean(log.(p0)))
    μ0_ = exp.(logμ0_)
    qval ~ For(eachindex(μ0_)) do j
            Poisson(μ0_[j])
    end
end
```

```
post_m2 = NamedTuple[(β = -1.97545, α = 3.5347), (β = -2.14622, α = 3.4771
```

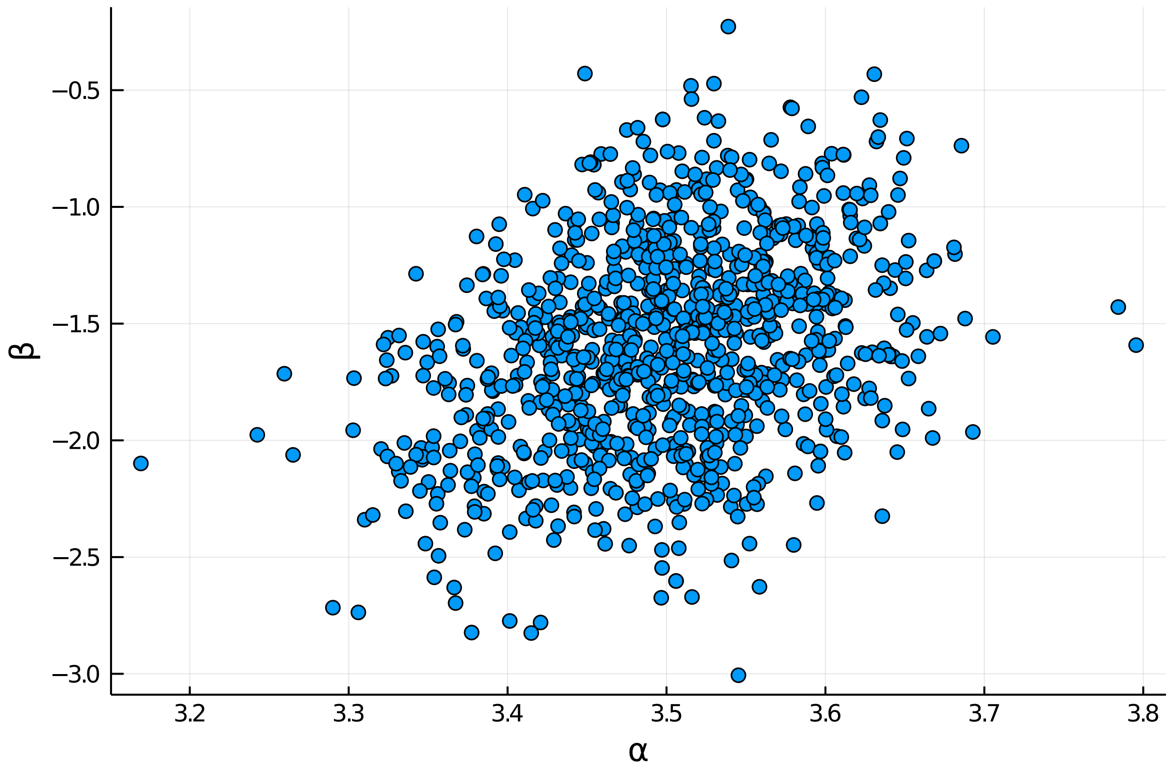
```
post_m2 = dynamicHMC(m2(), (p0=Price,qval=Quantity,))
```

```
Float64[-1.97545, -2.14622, -1.76546, -2.29369, -1.88791, -2.32478, -1.7
```

```
begin
    α_m2 = [vec.α for vec in post_m2]
    β_m2 = [vec.β for vec in post_m2]
end
```

If we plot β vs α, we see that they are not correlated.

```
md"If we plot β vs α, we see that they are not correlated."
```




```
begin
scatter(α_m2, β_m2, legend=false)
xlabel!("α")
ylabel!("β")
end
```

Now we want to plot different samples of α and β

```
md"Now we want to plot different samples of α and β"
```

```
p = range(25,65,step = 1);
```

```
t = sample(post_m2,1000);
```

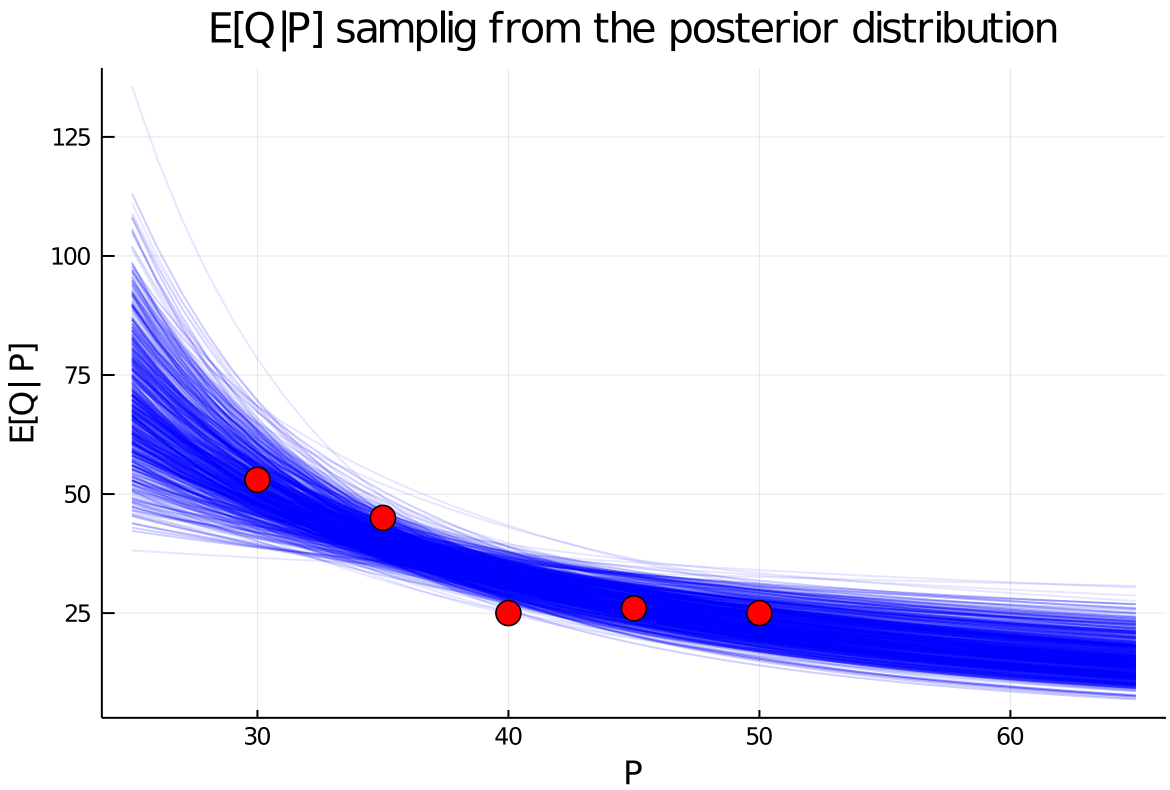
```
sample_α = [i.α for i in t];
```

```
sample_β = [i.β for i in t];
```

```
begin
    μ = zeros(length(p),length(sample_β))
    for i in collect(1:length(sample_β))
        μ[:,i] = exp.(sample_α[i] .+ sample_β[i] .* (log.(p) .- mean(log.(Price))))
    end
end
```

```
begin
```

```
.     plot(p,μ[:,1])
.     for i in collect(1:length(sample_β))
.     end
. end
```




```
. begin
. gr()
. plot(p,μ[:,1])
.     for i in collect(1:length(sample_β))
.             plot!(p,μ[:,i], color="blue", legend=false, alpha = 0.1)
.     end
. scatter!(Price, Quantity, color="red", markersize=7)
. title!("E[Q|P] samplig from the posterior distribution")
. ylabel!("E[Q| P]")
. xlabel!("P")
. current()
. end
```

Taking into account the unit cost of k=$20.

```
. md"Taking into account the unit cost of k=\$20."
```

k = 20

```
. k = 20
```

We compute now the profit π:

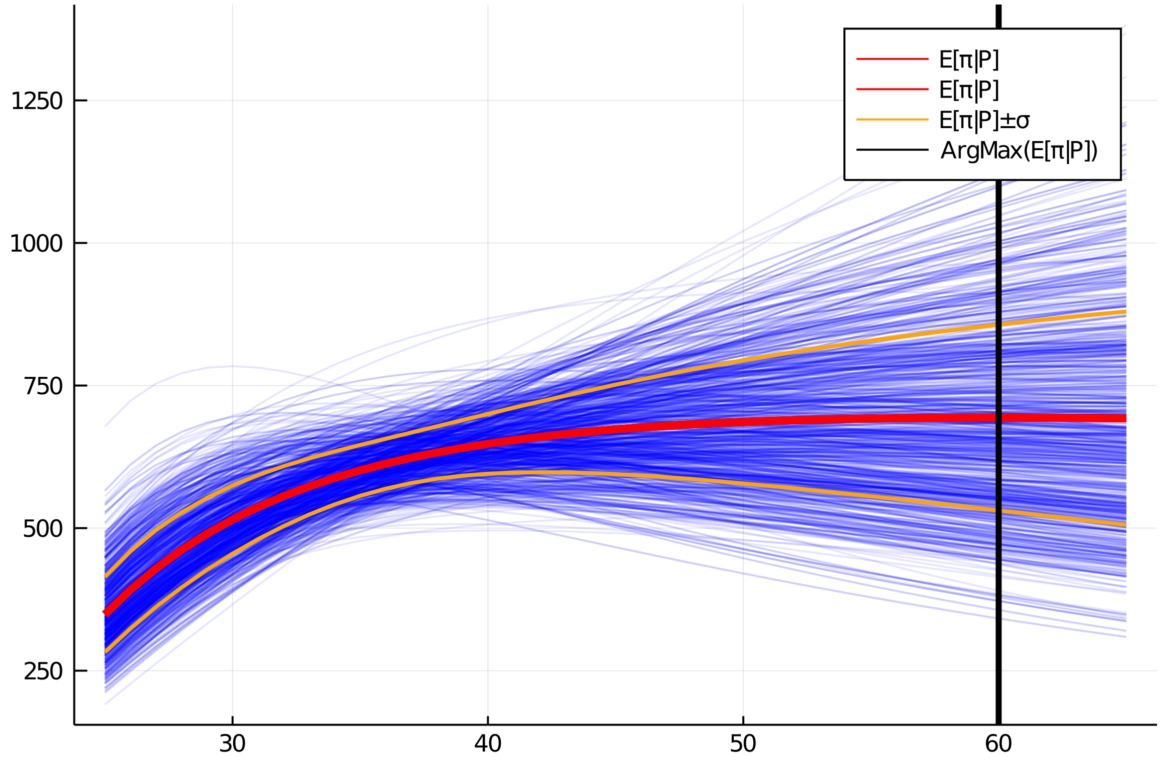
```
md"We compute now the profit π:"
```

```
π = (p .- k).*μ;
```

Now we find the maximum value and plot:

```
md"Now we find the maximum value and plot:"
```

```
mxval, mxindx = findmax(mean(π, dims=2); dims=1);
```




```
begin
plot(p,mean(π, dims=2), color = "red", lw=4, label="E[π|P]")
for i in collect(1:length(sample_β))
            plot!(p,π[:,i], color="blue", label=false, alpha = 0.1)
    end
plot!(p,mean(π, dims=2), color = "red", lw=4, label="E[π|P]")
plot!(p,mean(π, dims=2) + std(π, dims=2),  color = "orange", lw=2, label ="E[π|P]±σ")
plot!(p,mean(π, dims=2) - std(π, dims=2),  color = "orange", lw=2, label="")
vline!(p[mxindx], p[mxindx], line = (:black, 3), label="ArgMax(E[π|P])")
plot!(legend=true)
current()
end
```