Problem Sheet 5

```julia
using Pkg; Pkg.activate("."); Pkg.add(["Distributions", "Plots", "PyPlot",
"SpecialFunctions"])
```

```julia
begin
    using Distributions
    using LinearAlgebra
    using Plots
    using StatsPlots
    pyplot()
    using SpecialFunctions
    default(lw = 3.0, legendfontsize= 15.0)
end
```

1. Variational inference

Assume we have n observations $D = (x_1, \ldots, x_n)$ generated independently from a Gaussian density $\mathcal{N}(x|\mu, \tau^{-1})$, i.e.

$$p(D|\mu, \tau) = \left(\frac{\tau}{2\pi}\right)^{n/2} \exp\left[-\frac{\tau}{2}\sum_{i=1}^{n}(x_i - \mu)^2\right]$$

We also assume prior densities $p(\mu|\tau) = \mathcal{N}(\mu|\mu_0, (\lambda_0\tau)^{-1})$ and $p(\tau) = \mathrm{Gamma}(\tau|a_0, b_0)$. λ_0 and μ_0 as well as a_0, b_0 are given hyper parameters.

Our goal is to approximate the posterior density $p(\mu, \tau|D)$ by a **factorising density** $q(\mu, \tau) = q_1(\mu)q_2(\tau)$ which minimises the variational free energy

$$F[q] = \int q(\mu, \tau) \ln \frac{q(\mu, \tau)}{p(\mu, \tau, D)}\, d\mu\, d\tau$$

(a) [MATH] Show that the optimal $q_1(\mu)$ is a *Gaussian density* and give expressions for the mean and variance in terms of expectations with respect to q_2.

(b) [MATH] Show that the optimal $q_2(\tau)$ is a *Gamma density* and give expressions for the parameters in terms of expectations with respect to q_1.

> Tip
>
> You can use the following results which follow from the derivations given in the lecture
>
> $$\begin{aligned} q_1(\mu) &\propto \exp\left[E_\tau[\ln p(\mu, \tau, D)]\right] \\ q_2(\tau) &\propto \exp\left[E_\mu[\ln p(\mu, \tau, D)]\right] \end{aligned}$$

Solution

We have the representation of the joint density

$$p(\mu, \tau, D) = p(D|\mu, \tau)p(\mu|\tau)p(\tau)$$

with

$$\begin{aligned} p(\mu|\tau) =& \frac{(\lambda_0\tau)^{1/2}}{2\pi}\exp\left(-\frac{(\mu-\mu_0)^2\lambda_0\tau}{2}\right) \\ p(\tau) \propto& \tau^{a_0-1}e^{-b_o\tau} \end{aligned}$$

a)

Hence

$$\begin{aligned} E_\tau[\ln p(\mu, \tau, D)] =& -\frac{E_\tau[\tau]}{2}\sum_{i=1}^n(x_i-\mu)^2 - \frac{\lambda_0 E_\tau[\tau]}{2}(\mu-\mu_0)^2 + \text{const} = \\ & -\frac{1}{2}(E_\tau[\tau](n+\lambda_0))\mu^2 + \mu E_\tau[\tau]\left(\sum_i x_i + \lambda_0\mu_0\right) + \text{const} \end{aligned}$$

Note, that the second constant differs from the first. We get a Gaussian density for $q_1(\mu)$ with

$$\begin{aligned} E[\mu] &= \frac{\sum_i x_i + \lambda_0\mu_0}{n+\lambda_0} \\ \text{VAR}[\mu] &= \frac{1}{E_\tau[\tau](n+\lambda_0)} \end{aligned}$$

b)

for the density of $q_2(\tau)$, we use

$$E_\tau[\ln p(\mu,\tau,D)] = \ln\left(\tau^{a_0+(n+1)/2-1}e^{-b_0\tau}\right) - \frac{\tau}{2}\sum_{i=1}^{n}E_\mu[(x_i-\mu)^2] - \frac{\lambda_0\tau}{2}E_\mu[(\mu-\mu_0)^2] + \text{co}$$

We get a Gamma density

$$q_2(\tau) \propto \tau^{a_n-1}e^{-b_n\tau}$$

with parameters

$$\begin{aligned} a_n =& a_0 + (n+1)/2 \\ b_n =& b_0 + \frac{1}{2}\sum_{i=1}^{n}E_\mu[(x_i-\mu)^2] + \frac{\lambda_0}{2}E_\mu[(\mu-\mu_0)^2] \end{aligned}$$

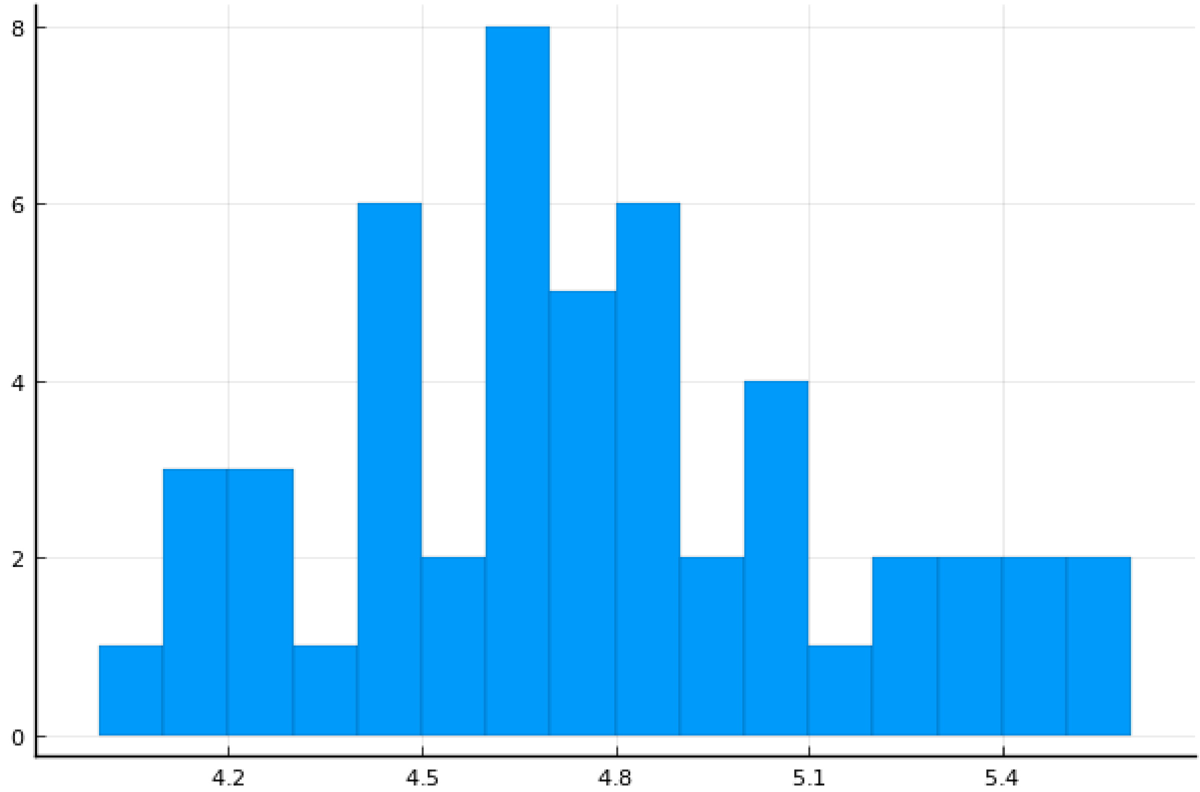
Knowing the form of both variational distributions we can also compute closed form solution of the expectations :

$$\begin{aligned} E_\tau[\tau] =& \frac{a_n}{b_n} \\ E_\mu\left[(x-\mu)^2\right] =& \text{Var}_\mu[\mu] + (x - E_\mu[\mu])^2 \end{aligned}$$

And proceed to coordinate ascent updates to converge to the optimal distribution.

c) [CODE] From the generated dataset implement a coordinate ascent scheme, updating variational parameters of τ and μ in an alternated way.

```
begin
    N = 50
    μ₀ = 5.0
    λ₀ = 2.0
    a₀ = 1.0
    b₀ = 2.0
    τ = rand(Gamma(a₀, b₀))
    μ = rand(Normal(μ₀, inv(sqrt(τ * λ₀))))
    x = rand(Normal(μ, inv(sqrt(τ))), N)
end;
```



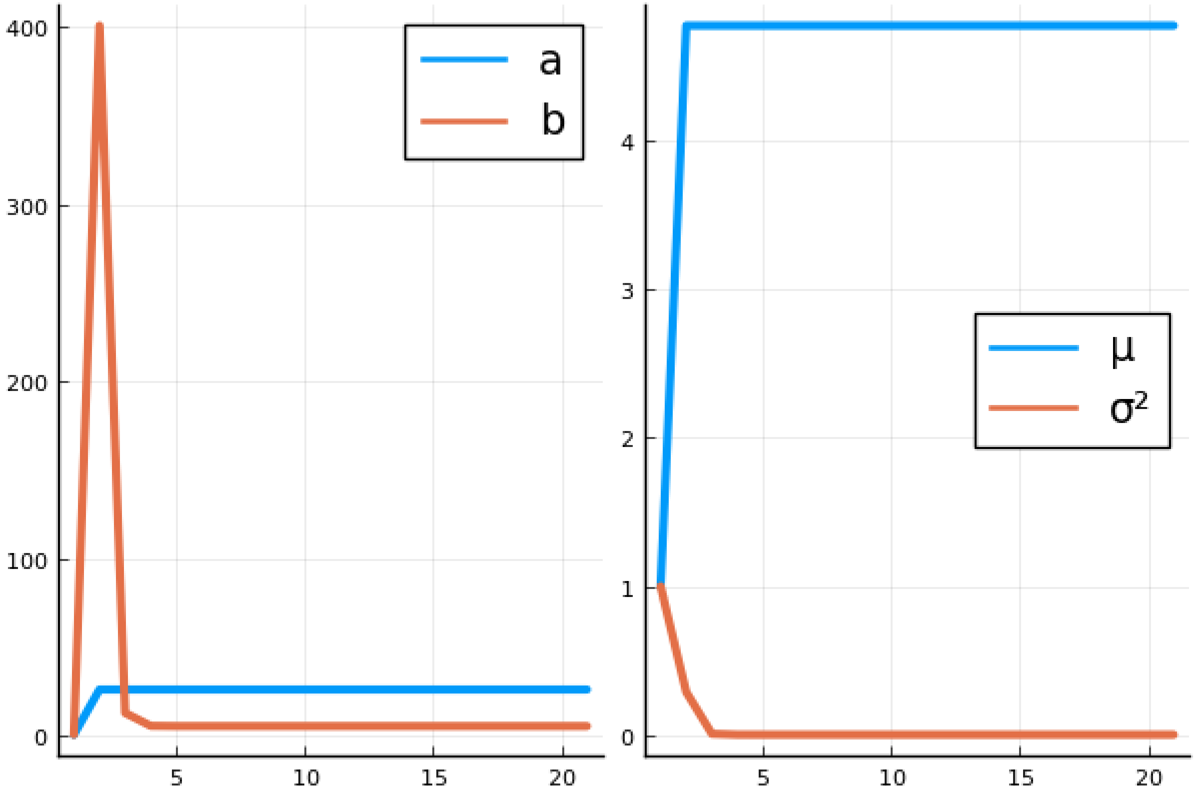

```julia
histogram(x, lab="", bins=20, lw=0.1)
```

```
bₙ (generic function with 1 method)
```

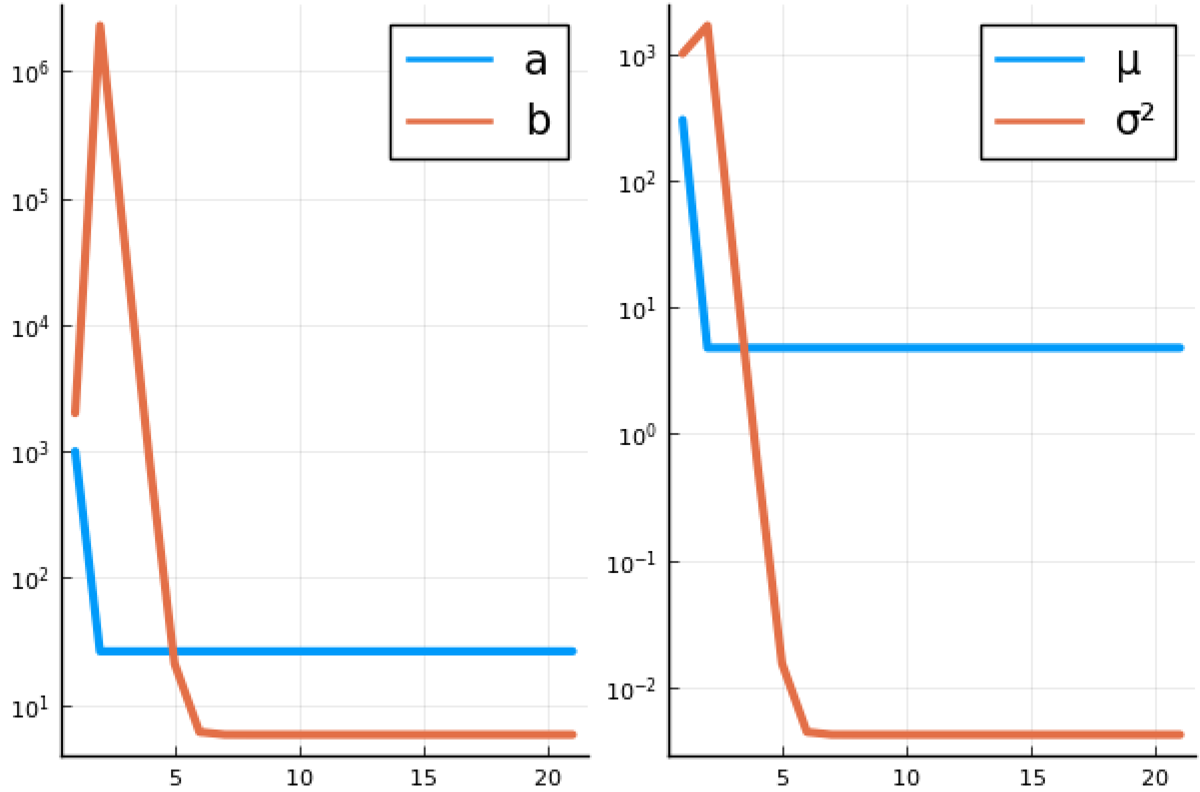
```julia
begin
    expec_μ(x, λ₀, μ₀, n) = (sum(x) + λ₀ * μ₀) / (n + λ₀)
    var_μ(e_τ, λ₀, n) = inv(e_τ * (n + λ₀))
    expec_τ(a, b) = a / b
    shifted_expec(e_mu, var_mu, x) = var_mu + abs2(x - e_mu)
    aₙ(a₀, n) = a₀ + (n + 1)/2
    bₙ(b₀, x, e_mu, var_mu, λ₀, μ₀) = b₀ + 0.5 * (sum(shifted_expec.(e_mu, var_mu,
x)) + λ₀ * shifted_expec(e_mu, var_mu, μ₀))
end
```

```
coordinate_ascent (generic function with 5 methods)
```

```julia
function coordinate_ascent(T, a = 1.0, b = 1.0, μ = 1.0, σ² = 1.0)
    as = vcat(a, zeros(T))
    bs = vcat(b, zeros(T))
    μs = vcat(μ, zeros(T))
    σs = vcat(σ², zeros(T))
    for i in 2:T+1
        a = aₙ(a₀, N); as[i] = a;
        b = bₙ(b₀, x, μ, σ², λ₀, μ₀); bs[i] = b
        e_τ = expec_τ(a, b)
        μ = expec_μ(x, λ₀, μ₀, N); μs[i] = μ
        σ² = var_μ(e_τ, λ₀, N); σs[i] = σ²
    end
    return as, bs, μs, σs
end
```




```
begin
    T = 20
    as, bs, mus, σs = coordinate_ascent(T)
    plt1 = plot([as bs], label = ["a" "b"])
    plt2 = plot([mus σs], label = ["μ" "σ²"])
    plot(plt1, plt2)
end
```




```
begin
    T_hard = 20
    as_hard, bs_hard, mus_hard, σs_hard = coordinate_ascent(T_hard, 1000, 2000, 300,
1e3)
    p1 = plot([as_hard bs_hard], label = ["a" "b"], yaxis = :log)
    p2 = plot([mus_hard σs_hard], label = ["μ" "σ²"], yaxis = :log)
    plot(p1, p2)
end
```

```
struct NormalGamma{T}
    μ::T
    λ::T
    shape::T
    rate::T
end
```

```
function Distributions.logpdf(d::NormalGamma, x::Real, τ::Real)
    C = d.shape * log(d.rate) - lgamma(d.shape) + 0.5 * (log(d.λ) - log(2π))
    return C + (d.shape - 0.5) * log(τ) - 0.5 * τ * (d.λ * (x - d.μ)^2 + 2 * d.rate)
end
```

```
struct VariationalDistribution{T1,T2}
    q1::T1
    q2::T2
end
```

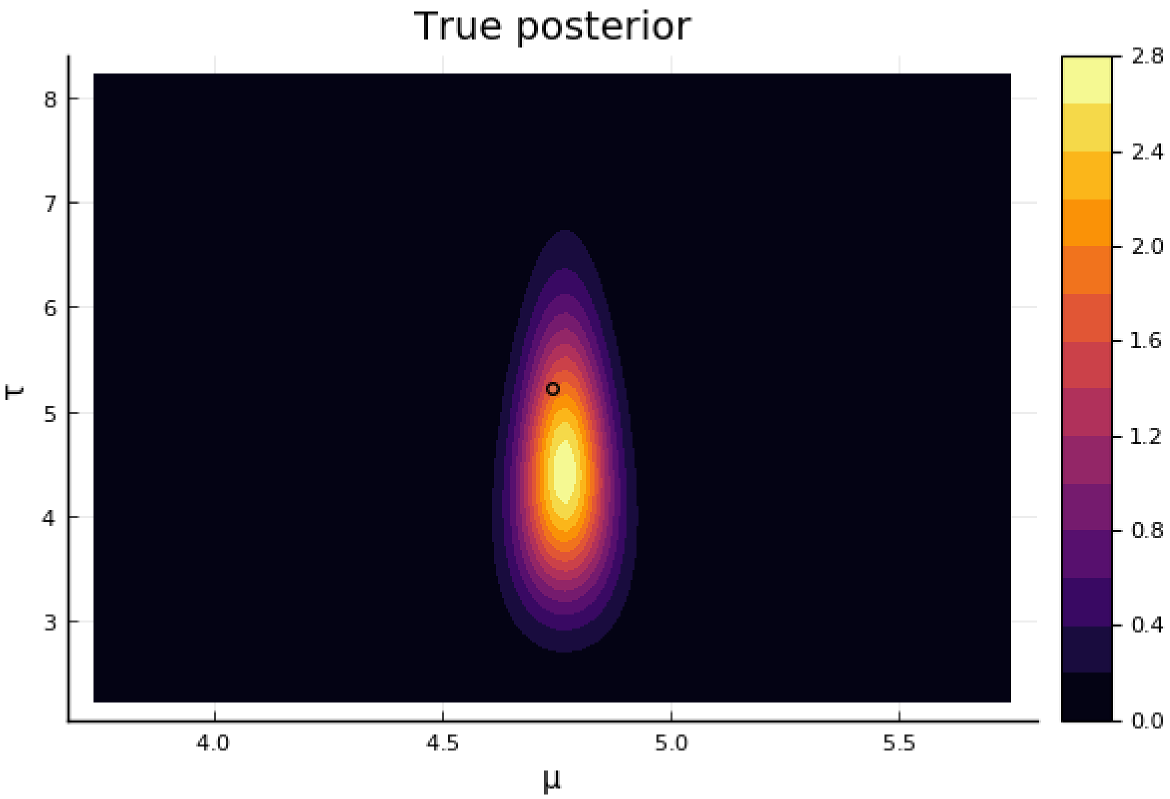
```
function Distributions.logpdf(d::VariationalDistribution, x::Real, τ::Real)
    return logpdf(d.q1, x) + logpdf(d.q2, τ)
end
```

```
opt_posterior =   NormalGamma(4.7682222302172335, 52.0, 26.0, 5.775007404266209)
```

```
opt_posterior = NormalGamma(
    (λ₀ * μ₀ + N * mean(x)) / (λ₀ + N),
    λ₀ + N,
    a₀ + N / 2,
    b₀ + 0.5 * (N * var(x) + (λ₀ * N * (mean(x) - μ₀)^2) / (λ₀ + N))
)
```

xrange = 3.7406232592167825:0.020202020202020204:5.7406232592167825

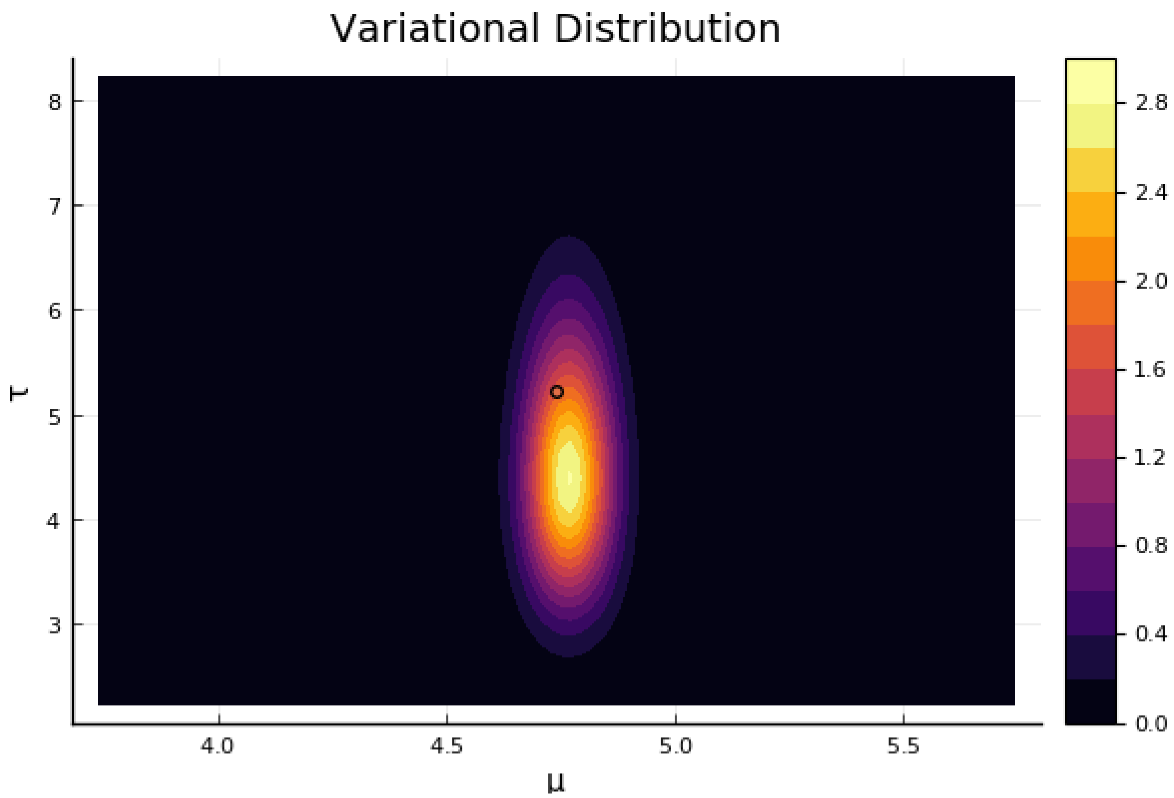
yrange = 2.223334152638107:0.06060606060606060594:8.223334152638106




```
begin
    contourf(xrange, yrange, (x,y)->exp(logpdf(opt_posterior, x, y)), title="True posterior",
xlabel="μ", ylabel="τ")
    scatter!([μ], [τ], lab="")
end
```

```
230217234, σ=0.06493410776925966), Distributions.Gamma{Float64}(α=26.5, θ=0.17210957314877
```

```
opt_variational = VariationalDistribution(
    Normal(mus[end], sqrt(σs[end])),
    Gamma(as[end], 1 / bs[end])
)
```




```
begin
    contourf(xrange, yrange, (x,y)->exp(logpdf(opt_variational, x, y)),
title="Variational Distribution", xlabel="μ", ylabel="τ")
    scatter!([μ], [τ], lab="")
end
```