Bayesian NLME for safer powder storage

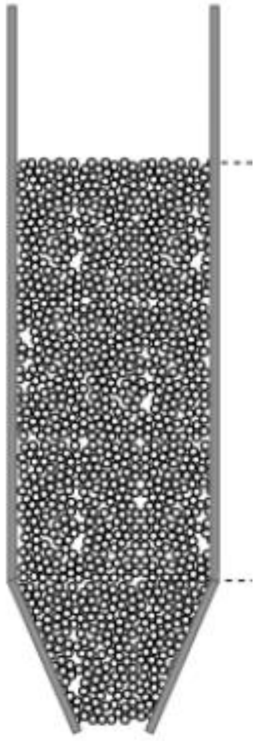
Arno Strouwen, Margot Coppens,
Martin Otava, Michael Van den Eynde

Storing powders in silos

- Models first developed for agriculture
  - Powders for pharmaceutical manufacturing of tablets & pills
- Model stress to ensure safety of silo
  - Janssen equation

$$\sigma_{v,\mathrm{bottom}} = \frac{\frac{g\rho D}{4} - b}{Ka} - \frac{\frac{g\rho D}{4} - b}{Ka} \exp\left(-\frac{4Ka}{D} z_{\mathrm{eq}}\right)$$

Estimate unknown parameters from data



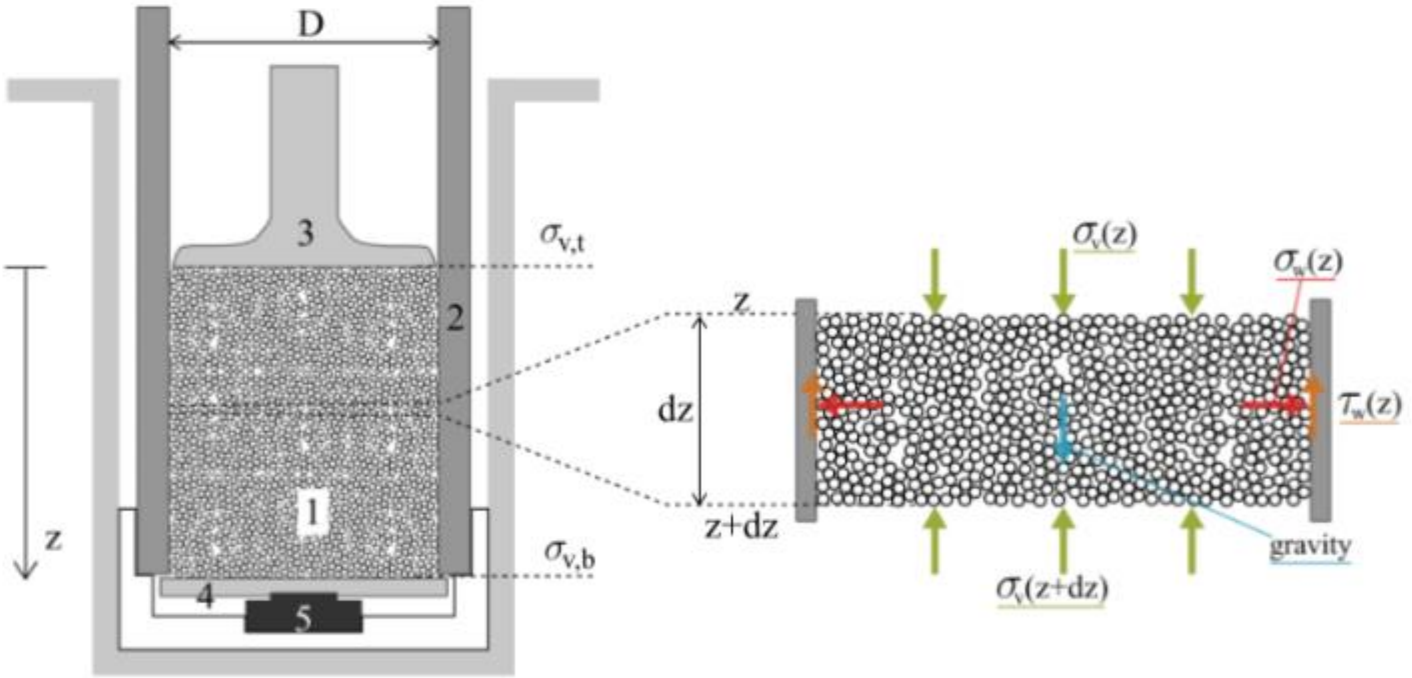
Source: deepai.org

First measurement setup

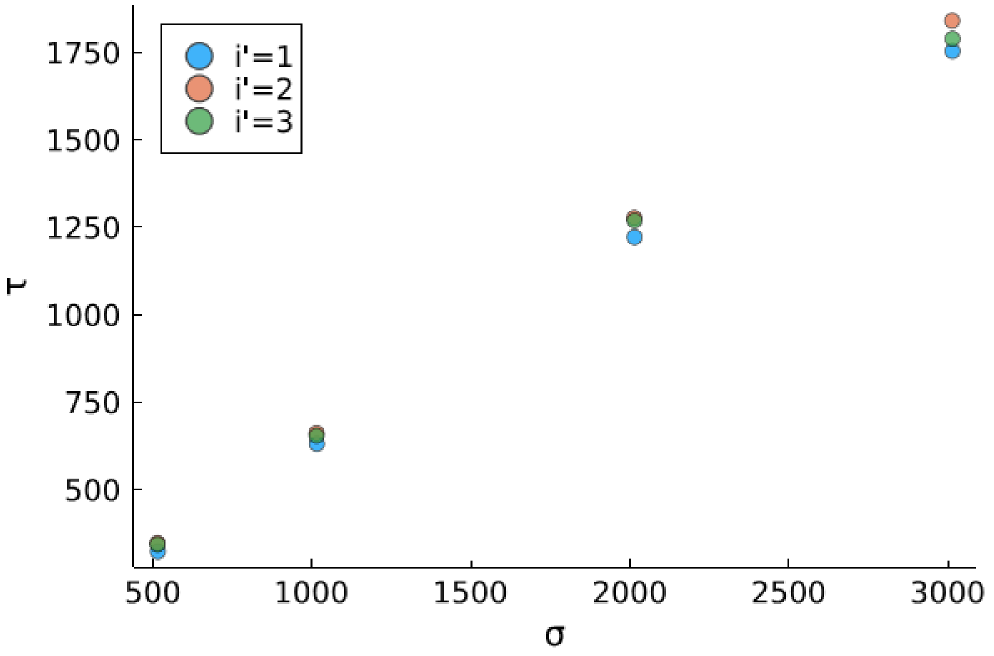
- Force balance

$$\sigma_{v,\mathrm{bottom}} = \frac{\frac{g\rho D}{4} - b}{Ka} + \left(\sigma_{v,top} - \frac{\frac{g\rho D}{4} - b}{Ka}\right) \exp\left(-\frac{4Ka}{D} z_{\mathrm{eq}}\right)$$

- Unidentifiable



Second measurement setup

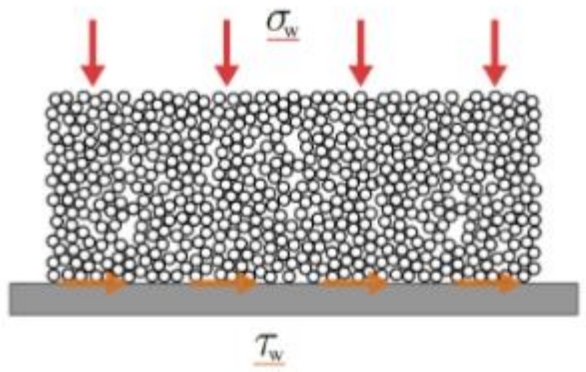


- Force balance

$$\tau = a\sigma_h + b$$

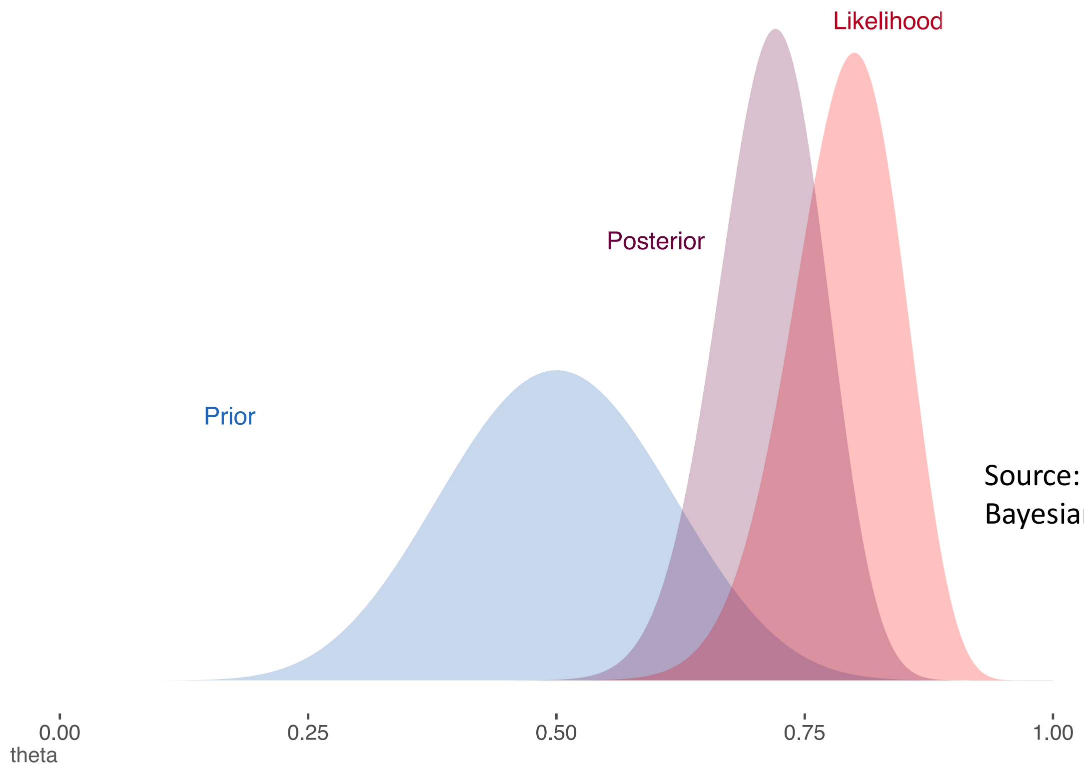
- Random effect

$$a^{i'} \sim \mathcal{N}(m_a, s_a)$$



Why Bayesian?

- Prior information available
- Joint likelihood for combined dataset is easy
  - Frequentist estimator is difficult



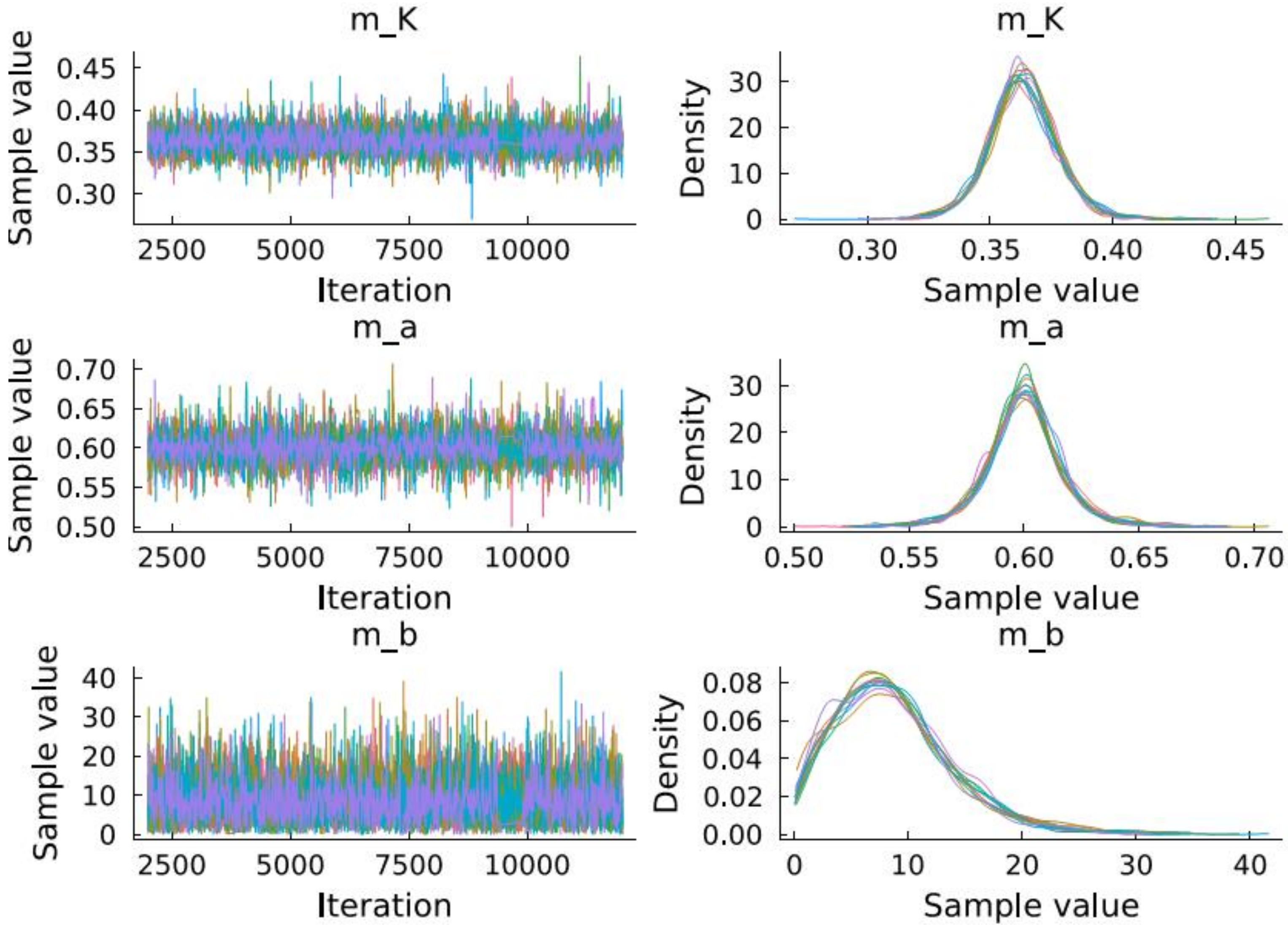
Source: Michael Clarc, Bayesian Basics

Why Julia?

$$
\begin{aligned}
p(m_K) &\sim \text{Uniform}(0,1) \\
p(m_a) &\sim \text{Uniform}(0,1) \\
p(m_b) &\sim \text{Uniform}(0,500) \\
p(s_K) &\sim \text{Uniform}(0,0.1) \\
p(s_a) &\sim \text{Uniform}(0,0.1) \\
p(s_b) &\sim \text{Uniform}(0,50) \\
p(s_1) &\sim \text{Uniform}(0,250) \\
p(s_2) &\sim \text{Uniform}(0,250) \\
p(K^i|m_K,s_K) &\sim \mathcal{N}(m_K,s_k) \\
p(a^i|m_a,s_a) &\sim \mathcal{N}(m_a,s_a) \\
p(b^i|m_b,s_b) &\sim \mathcal{N}(m_b,s_b) \\
p(a^{i'}|m_a,s_a) &\sim \mathcal{N}(m_a,s_a) \\
p(b^{i'}|m_b,s_b) &\sim \mathcal{N}(m_b,s_b) \\
p(\sigma^{i,j}_{v,\text{bottom}}|K^i,a^i,b^i,s_1) &\sim \mathcal{N}\left(\frac{\frac{g\rho^{i,j}D}{4}-b^i}{K^ia^i}+\left(\sigma^j_{v,top}-\frac{\frac{g\rho^{i,j}D}{4}-b^i}{K^ia^i}\right)\exp\left(-\frac{4K^ia^i}{D}z^{i,j}_{\text{eq}}\right),s_1\right) \\
p(\tau^{i',j'}|a^{i',j'},b^{i',j'},s_2) &\sim \mathcal{N}(a^{i'}\sigma_h^{i',j'}+b^{i'},s_2)
\end{aligned}
$$

```julia
@model function powder_mechanical_model(σ_h, τ, σ_v_top, σ_v_bottom, z_eq, m, D, g)
    m_K ~ Uniform(0.0,1.0)
    s_K ~ Uniform(0.0,0.1)
    m_a ~ Uniform(0.0,1.0)
    s_a ~ Uniform(0.0,0.1)
    m_b ~ Uniform(0.0,500.0)
    s_b ~ Uniform(0.0,50.0)
    s_1 ~ Uniform(0.0,250.0)
    s_2 ~ Uniform(0.0,250.0)
    K1 = Vector{Float64}(undef, size(σ_v_bottom,2))
    a1 = Vector{Float64}(undef, size(σ_v_bottom,2))
    b1 = Vector{Float64}(undef, size(σ_v_bottom,2))
    for i in 1:size(σ_v_bottom,2)
        K1[i] ~ Normal(m_K,s_K)
        a1[i] ~ Normal(m_a,s_a)
        b1[i] ~ Normal(m_b,s_b)
        for j in 1:size(σ_v_bottom,1)
            ρ = m[i]*4/(π*D^2*z_eq[j,i])
            term = (g*ρ*D/4-b1[i])/(K1[i]*a1[i])
            mean = term + (σ_v_top[j]-term)*exp(-4*K1[i]*a1[i]*z_eq[j,i]/D)
            σ_v_bottom[j,i] ~ Normal(mean,s_1)
        end
    end
    a2 = Vector{Float64}(undef, size(σ_h,2))
    b2 = Vector{Float64}(undef, size(σ_h,2))
    for i in 1:size(σ_h,2)
        a2[i] ~ Normal(m_a,s_a)
        b2[i] ~ Normal(m_b,s_b)
        for j in 1:size(σ_h,1)
            τ[j,i] ~ Normal(a2[i]*σ_h[j,i]+b2[i],s_2)
        end
    end
    return nothing
end
```

Posterior


m_K
Sample value
0.45
0.40
0.35
0.30
2500
5000
7500
10000
Iteration
m_K
Density
30
20
10
0
0.30
0.35
0.40
0.45
Sample value
m_a
Sample value
0.70
0.65
0.60
0.55
0.50
2500
5000
7500
10000
Iteration
m_a
Density
30
20
10
0
0.50
0.55
0.60
0.65
0.70
Sample value
m_b
Sample value
40
30
20
10
0
2500
5000
7500
10000
Iteration
m_b
Density
0.08
0.06
0.04
0.02
0.00
0
10
20
30
40
Sample value