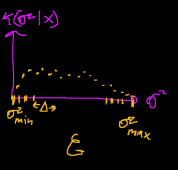
Grid Sampling Algorithm

1) Create a grid by first picking σ^2_{min}, σ^2_{max}, Δ.



$$G := \{\sigma^2_{min}, \sigma^2_{min}+\Delta, \sigma^2_{min}+2\Delta, \ldots, \sigma^2_{max}-\Delta, \sigma^2_{max}\}$$

For example, in our case let $\sigma^2_{min}=0$, $\sigma^2_{max}=1000$, $\Delta=0.1$

$$G = \{0, 0.1, 0.2, \ldots, 999.9, 1000\}$$

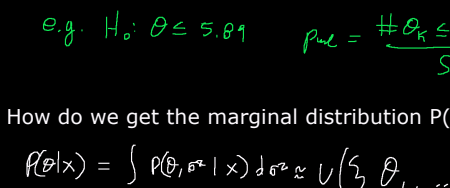
2) Approximate the value of c, the normalization constant via Riemann sum (if continuous).

$$c = \frac{1}{\int_{Sample(\sigma^2)} k(\sigma^2|x)\,d\sigma^2} \approx \frac{1}{\Delta \sum_{\sigma^2 \in G} k(\sigma^2|x)}$$

3) Compute the "sampling CDF" for all grid points:

$$= \int_{-\infty}^{\sigma^2_0} p(\sigma^2|x)\,d\sigma^2$$

$$F(\sigma^2_0|x) := P(\sigma^2 \le \sigma^2_0|x) \approx \sum_{\{\sigma^2 : \sigma^2 \in G,\ \sigma^2 \le \sigma^2_0\}} c\,k(\sigma^2|x)$$



4) Draw u from U(0,1) and locate $\sigma^2_{sample} = \min_{\sigma^2 \in G} \{F(\sigma^2|x) \ge u\}$

If you want to draw many samples, repeat step 4. You only need to do steps 1-3 once.

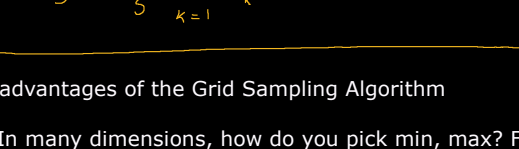
Returning to our problem, how do we sample from the posterior P(θ, σ^2 | X)?

(1) Sample σ^2_sample from P(σ^2 | X) using the grid sampler.
(2) Sample θ_sample from P(θ | σ^2, X) via rnorm.

Return [θ_sample, σ^2_sample]

At this point, we're not limited to distributions we know about nor limited to using conjugate priors. We're liberated!

We agree we can sample now from any arbitrary posterior. But how do we get point estimates?



I now have as many samples as I wish from my posterior (you want more? Let the computer churn more). You have S samples:

$$\begin{bmatrix}\theta_1\\ \sigma^2_1\end{bmatrix}, \begin{bmatrix}\theta_2\\ \sigma^2_2\end{bmatrix}, \ldots, \begin{bmatrix}\theta_S\\ \sigma^2_S\end{bmatrix}$$

How do we use these samples to get point estimates?

$$\hat{\theta}_{MMSE} \approx \frac{1}{S}\sum_{k=1}^{S} \theta_k$$

$\hat{\theta}_{MMAE} \approx$ SampleMedian$[\theta_1, \ldots, \theta_S]$ i.e. you order all the samples and return the middle value

$\hat{\theta}_{MAP}$ complicated... you need you use smoothing... so forget it

$$\hat{\sigma}^2_{MMSE} = \frac{1}{S}\sum_{k=1}^{S} \sigma^2_k$$

$$\hat{\sigma}^2_{MMAE} = \text{SampleMedian}[\sigma^2_1, \ldots, \sigma^2_S]$$

How do we get CR's?

$$CR_{\theta, 1-\alpha_0} := \left[Q[\theta|x, \tfrac{\alpha_0}{2}], Q[\theta|x, 1-\tfrac{\alpha_0}{2}]\right] \approx \left[\text{SampleQuantile}[\theta_1, \ldots, \theta_S, \tfrac{\alpha_0}{2}], \text{SampleQuantile}[\theta_1, \ldots, \theta_S, 1-\tfrac{\alpha_0}{2}]\right]$$

How do we get pvals for hypothesis tests?

$$H_0: \theta \in \Theta_0, \quad p_{val} := P(\theta \in \Theta_0|x) = \int_{\Theta_0} p(\theta|x)\,d\theta \approx \frac{1}{S}\sum_{k=1}^{S} 1_{\theta_k \in \Theta_0}$$

e.g. $H_0: \theta \le 5.89$ $\quad p_{val} = \frac{\#\theta_k \le 5.89}{S} =$ proportion ≤ 5.89

How do we get the marginal distribution P(θ | X)?

$$p(\theta|x) = \int_{Sample} p(\theta, \sigma^2|x)\,d\sigma^2 \approx U(\{\theta_1, \ldots, \theta_S\})$$

(Samples)

uniform discrete distribution

How do we get the posterior predictive distribution P(X* | X)?

$$p(X_*|x) = \int_0^\infty \int_{\mathbb{R}} \underbrace{p(X_*|\theta, \sigma^2)}_{N(\theta, \sigma^2)} \underbrace{p(\theta, \sigma^2|x)}_{??}\,d\theta\,d\sigma^2 = \int\int p(X_*, \theta, \sigma^2|x)\,d\theta\,d\sigma^2$$

You use the sample [θ_1, σ^2_1] and then draw x*_1 from rnorm(θ_1, σ^2_1). Then use the sample [θ_2, σ^2_2] and then draw x*_2 from rnorm(θ_2, σ^2_2) ..., Then use the sample [θ_S, σ^2_S] and then draw x*_S from rnorm(θ_S, σ^2_S):

$$U(\{X_{*_1}, X_{*_2}, \ldots, X_{*_S}\}) \approx P(X_*|x)$$

$$E[X_*|x] \approx \frac{1}{S}\sum_{k=1}^{S} X_{*_k}$$

Disadvantages of the Grid Sampling Algorithm

1) In many dimensions, how do you pick min, max? For parameters in multiparameter models, it's not so simple!
2) Computers have numerical underflow and overflow
3) "Curse of dimensionality". Let's say you want to sample 10,000 grid points per dimension. But you have 10 different thetas. 10,000^10 = 10^50 which is IMPOSSIBLE for the computer. As another example: let's say I want 1 billion points in 10 dimensions. That is only: $\sqrt[10]{10^9} \approx 8$ points per dimension.

That's not good resolution at all!!!

We need another solution to this problem!!!