Bayesian inference: a theory of parameter estimation

Contents


```r
setwd("~/OneDrive - Moffitt Cancer Center/Documents/IMAT/Bayesian_Inference-HMC-Stan")
rm(list=ls()) # clear all variables
library(tidyverse) #
library(magrittr) # pipe operator ( %>% )
library(ggplot2) # plotting
knitr::opts_chunk$set(error=T)
theme_set(theme_bw()) # set default ggplot theme
```

Toy model: exponential tumor growth law

Let's begin our exploration of Bayesian inference with a simple toy model to play with: the exponential tumor growth law:

$$\mathrm{TS}(t) = \mathrm{TS}_0 \cdot \exp\left(k_{ge} \cdot t\right)$$

with parameters TS_0 (initial tumor size) and k_{ge} (exponential growth rate). However, when we go to measure tumor size, even if the tumor were to exactly follow an exponential growth law, there will still be some residual error. Now comes the big decision: how should we model this residual error? There are a number of choices we could make. For this lecture, since tumor size is non-negative, let's consider a log-transformed additive constant error model. For each observation (measurement) y_j, we have:

$$\ln(\overbrace{y_j}^{\text{observation}}) = \ln(\overbrace{\mathrm{TS}_j}^{\text{true value}}) + \overbrace{\epsilon_j}^{\text{residual error}}$$

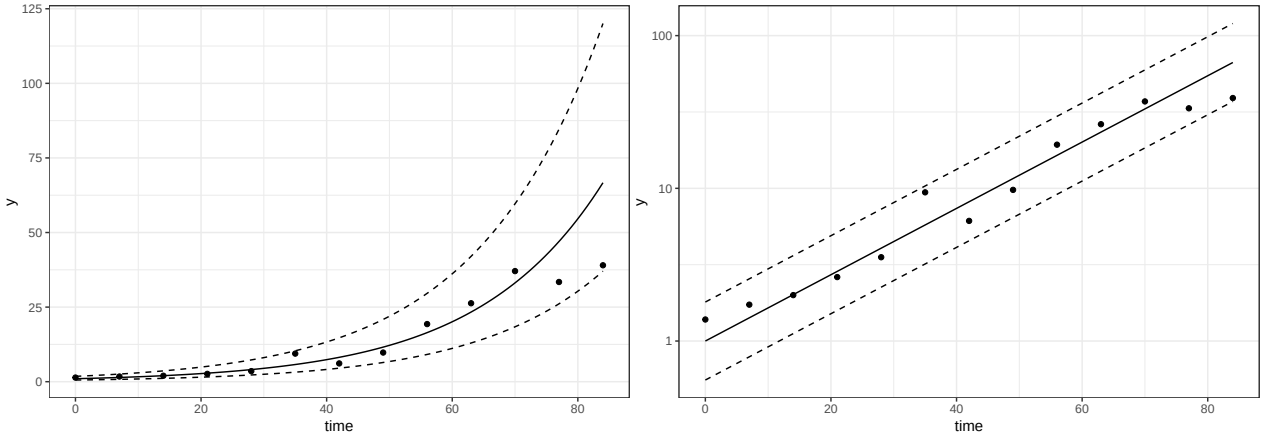
$$\overbrace{\epsilon_j}^{\text{residual error}} \sim N(\overbrace{0}^{\text{bias}}, \overbrace{\sigma}^{\text{uncertainty}})$$

Equivalently, this model could be rewritten simply as:

$$\ln(\overbrace{y_j}^{\text{observation}}) \sim N(\ln(\overbrace{\text{TS}_j}^{\text{true value}}), \overbrace{\sigma}^{\text{residual error uncertainty}})$$

Now, let's generate some data by simulation:

```
# model parameters
TS0 <- 1
kge <- 0.05
sigma <- 0.3

# exponential tumor growth model
exp.model <- function(time) TS0 * exp(kge*time)

# simulate data
sim.data <- tibble(time = seq(0, 90, 7), TS = exp.model(time)) %>% # true TS
  rowwise() %>%
  mutate(y = rlnorm(n=1, meanlog=log(TS), sdlog=sigma)) %>% # data (observed TS)
ungroup()

# plot simulated data
p.sim.data.linear <- ggplot(data = sim.data, mapping = aes(x=time, y=y)) +
  # data (observed TS)
  geom_point() +
  # model (true TS)
  stat_function(fun=exp.model) +
  # 95% CI lower bound
  stat_function(fun=~exp(log(exp.model(.x)) - 1.96*sigma), linetype="dashed") +
  # 95% CI upper bound
  stat_function(fun=~exp(log(exp.model(.x)) + 1.96*sigma), linetype="dashed")

p.sim.data.log <- p.sim.data.linear + scale_y_log10()

p.sim.data.linear
p.sim.data.log
```



Bayes' Theorem

Now suppose we want to infer the tumor exponential growth rate k_{ge} from measured tumor size y. In this case, our quantity of interest is $k_{ge} \mid y$. In general, we can consider any sort of quantity of interest given any

sort of data $\theta \mid y$. Now how can we estimate this quantity? Moreover, how can we quantity the uncertainty in this estimate?

The Bayesian approach to this problem treats y and θ as random variables with joint probability density function (pdf) $p(y, \theta)$. One of the consequences of this level of abstraction is that data y and parameters θ are qualitatively and conceptually indistinguishable! This is quite a conceptual leap for our everyday intuition and will take some time to sink in, but the ramifications of this level of abstaction are massive!

Now, in Probability Theory, we have two fundamental rules:

$$\text{chain rule:} \quad p(y, \theta) = p(y \mid \theta)p(\theta) \tag{1}$$

$$\text{law of total probability:} \quad p(y) = \int_{\Theta} p(y, \theta) d\theta \tag{2}$$

From these two fundamental rules, we can derive Bayes' Theorem:

$$\overbrace{p(\theta \mid y)}^{\text{posterior}} = \frac{\overbrace{p(y \mid \theta)}^{\text{likelihood}} \times \overbrace{p(\theta)}^{\text{prior}}}{\underbrace{p(y)}_{\text{evidence}}} = \frac{p(y \mid \theta) \times p(\theta)}{\int_{\Theta} p(y \mid \theta')p(\theta')d\theta'}$$

Since the evidence $p(y) = \int_{\Theta} p(y \mid \theta')p(\theta')d\theta'$ is usually intractable and is constant relative to the quantity of interest θ, we can take it to be some normalizing proportionality constant:

$$\overbrace{p(\theta \mid y)}^{\text{posterior}} \propto \overbrace{p(y \mid \theta)}^{\text{likelihood}} \times \overbrace{p(\theta)}^{\text{prior}}$$

This, by the way, is precisely why we often resort to sampling methods rather than directly computing the posterior distribution (more on this later).

We can visualize these distributions by varying parameter θ on the x-axis and plotting the probability density function (pdf) on the y-axis.

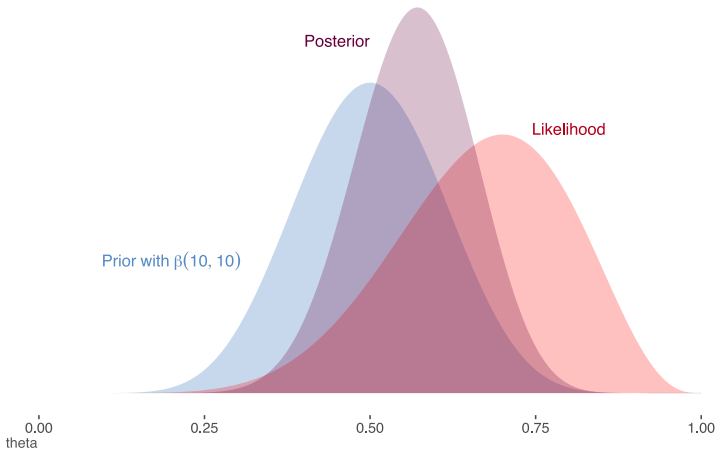


Figure 1: Prior, likelihood, and posterior distributions

Applying Bayes' Theorem to the exponential tumor growth law

Okay, this all looks fine and dandy, but now how does this relate back to our toy model of exponential tumor growth? For simplicity, let's assume that $\mathrm{TS}_0 = \hat{\mathrm{TS}}_0$ is known and we're trying to estimate unknown k_{ge}.

Likelihood

First, we'll take a look at the likelihood $p(y \mid k_{ge}; \hat{\mathrm{TS}}_0)$. Recall that we had:

$$\ln(y_j) \sim N(\ln(\mathrm{TS}_j), \sigma)$$

We can more explicit about our parameters:

$$\ln(y_j) \mid k_{ge}; \hat{\mathrm{TS}}_0 \sim N\left(\ln\left[\mathrm{TS}\left(t_j \mid k_{ge}; \hat{\mathrm{TS}}_0\right)\right], \sigma\right)$$

Now given the probability density function (pdf) of a log-normal distribution $\ln(y) \sim N(\ln(\mu), \sigma)$:

$$p(y) = \frac{1}{y\sigma\sqrt{2\pi}} \cdot \exp\left(-\frac{1}{2} \cdot \left(\frac{\ln y - \ln \mu}{\sigma}\right)^2\right)$$

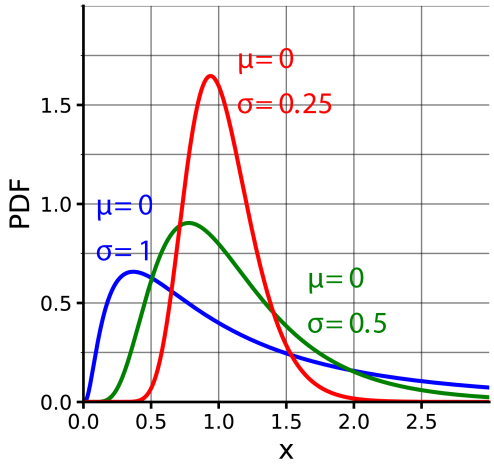


Figure 2: Pdf of log-normal distribution

we obtain the following likelihood function:

$$p(y_j \mid k_{ge}; \hat{\mathrm{TS}}_0) = \frac{1}{y_j\sigma\sqrt{2\pi}} \cdot \exp\left(-\frac{1}{2} \cdot \left(\frac{\ln y_j - \ln\left[\mathrm{TS}\left(t_j \mid k_{ge}; \hat{\mathrm{TS}}_0\right)\right]}{\sigma}\right)^2\right)$$

This was just for one observation. Now suppose we have m observations. Since all observations y have independent and identically distributed (i.i.d.) residual errors ϵ conditioned on model parameters, we have:

$$
\begin{aligned}
p(y \mid k_{ge}; \hat{\mathrm{TS}}_0) &= \prod_{j=1}^{m} p(y_j \mid k_{ge}; \hat{\mathrm{TS}}_0) \\
&= \left(2\pi\sigma^2\right)^{-\frac{m}{2}} \cdot \prod_{j=1}^{m} \frac{1}{y_j} \cdot \exp\left(-\frac{1}{2} \cdot \left(\frac{\ln y_j - \ln\left[\mathrm{TS}\left(t_j \mid k_{ge}; \hat{\mathrm{TS}}_0\right)\right]}{\sigma}\right)^2\right)
\end{aligned}
$$

Okay, great! Now in practice, it's much easier to work to sums than products, so we normally consider the logarithm of the likelihood (or prior, posterior), so this is what we're going to do from now on:

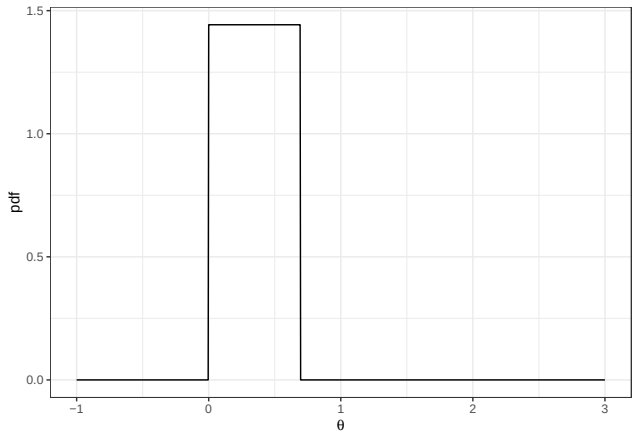
$$
\begin{aligned}
LL &= \ln\left[p(y \mid k_{ge}; \hat{\mathrm{TS}}_0)\right] \\
&= \sum_{j=1}^{m} \ln\left[p(y_j \mid k_{ge}; \hat{\mathrm{TS}}_0)\right] \\
&= \overbrace{-\frac{m}{2}\ln(2\pi\sigma^2) - \sum_{j=1}^{m} \ln y_j}^{\text{additive constant w.r.t. } k_{ge}} \quad \overbrace{-\frac{1}{2\sigma^2}}^{\substack{\text{multiplicative} \\ \text{constant} \\ \text{w.r.t. } k_{ge}}} \quad \cdot \overbrace{\sum_{j=1}^{m}\left(\ln y_j - \ln\left[\mathrm{TS}\left(t_j \mid k_{ge}; \hat{\mathrm{TS}}_0\right)\right]\right)^2}^{\text{sum of squared errors (SSE)}}
\end{aligned}
$$

Prior

Now we need to define the prior distribution $p(k_{ge})$, which contains all of our prior beliefs about the plausible values that k_{ge} can take. There are a few options that we can take:

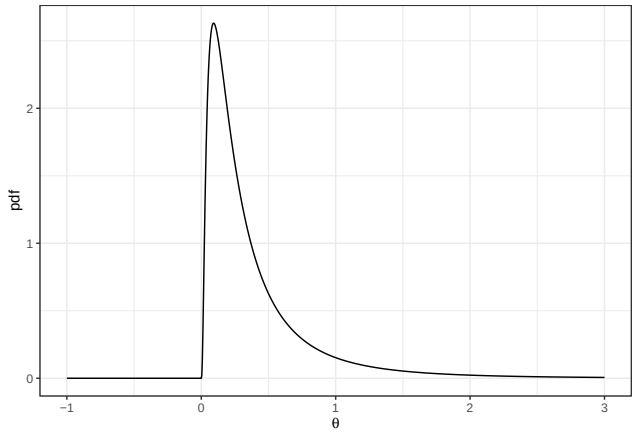
1. All values are plausible: this produces an improper uninformative prior $k_{ge} \sim U(-\infty, \infty)$. This is "improper," because $\ln\left[p(k_{ge})\right] = \lim_{\substack{a\to-\infty \\ b\to\infty}} -\ln(b-a)$ is undefined while $\int_{-\infty}^{\infty} p(k_{ge}) = 1$.
2. Alternatively, we may recognize that tumors have positive growth rate. In this case, we get another improper prior $k_{ge} \sim U(0, \infty)$. This contains a little more information than the previous option but still fairly uninformative.
3. However, we also recognize that there are physiological and biological bounds to tumor growth. Therefore, we can construct an upper bound, say $b = \ln 2$. Now our prior become proper: $k_{ge} \sim U(0, b)$ with $\ln\left[p(k_{ge})\right] = \begin{cases} -\ln b & \text{if } 0 < k_{ge} < b \\ 0 & \text{otherwise} \end{cases}$.

```
ggplot() +
  labs(x=bquote(theta), y="pdf") +
  stat_function(fun=~dunif(.x, min=0, max=log(2)), n=1001, xlim=c(-1,3))
```



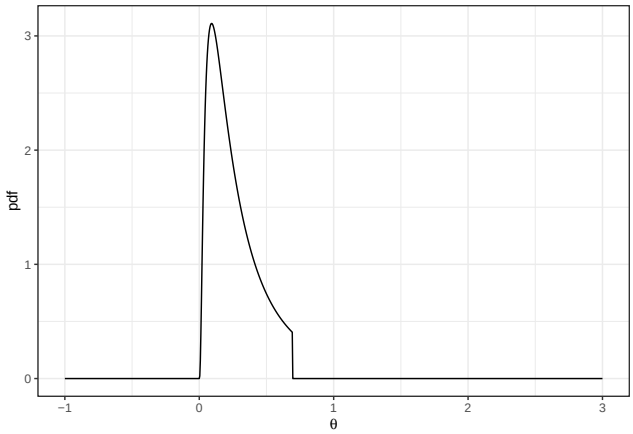
4. However, not every value on the interval $(0, b)$ may be equally plausible. A common choice for priors on biological parameters that restricted to be positive is the log-normal distribution: $\ln(k_{ge}) \sim N(\ln(\mu), \sigma)$, so that $\ln[p(k_{ge})] = -\ln(k_{ge}\sigma\sqrt{2\pi}) - \frac{1}{2}\left(\frac{\ln k_{ge} - \ln \mu}{\sigma}\right)^2$, where μ is the geometric mean and $\sigma > 0$ is a dispersion parameter. We can control how informative (uninformative) is our prior by decreasing (increasing) σ. By default, I like to start with $\sigma = 1$.

```
ggplot() +
  labs(x=bquote(theta), y="pdf") +
  stat_function(fun=~dlnorm(.x, meanlog=log(0.25), sdlog=1), n=1001, xlim=c(-1,3))
```



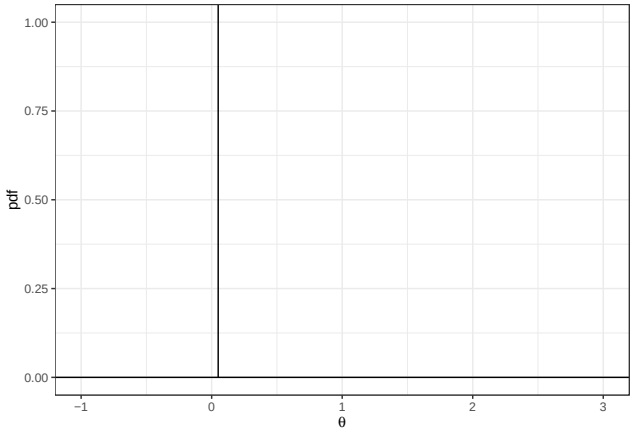
5. We can combine the previous two options by truncated the log-normal distribution as $\ln(k_{ge}) \sim N(\ln(\mu), \sigma)\ T(\ln(0), \ln(b))$, so that $\ln[p(k_{ge})] = \begin{cases} -\ln k_{ge} - \frac{1}{2}\left(\frac{\ln k_{ge} - \ln \mu}{\sigma}\right)^2 - \ln\left(\int_{-\infty}^{\ln b} e^{-\frac{1}{2}\left(\frac{x - \ln \mu}{\sigma}\right)^2} dx\right) & \text{if } 0 < k_{ge} < b \\ -\infty & \text{otherwise} \end{cases}$.

```
ggplot() +
  labs(x=bquote(theta), y="pdf") +
  stat_function(fun=~EnvStats::dlnormTrunc(.x, meanlog=log(0.25), sdlog=1,
                                           min=0, max=log(2)),
                n=1001, xlim=c(-1,3))
```



6. Finally, for completeness, we can go to the opposite extreme and assume perfect prior information, which just amounts to fixing $k_{ge} = \hat{k}_{ge}$. In this case, k_{ge} follows a Dirac delta distribution, so that $\ln\left[p(k_{ge})\right] = \ln\left[\delta(k_{ge} - \hat{k}_{ge})\right] = \begin{cases} \infty & \text{if } k_{ge} - \hat{k}_{ge} = 0 \\ -\infty & \text{otherwise} \end{cases}$, such that $\int_{-\infty}^{\infty} p(k_{ge}) = 1$. Notice that this is what we did in the case of $\mathrm{TS}_0 = \hat{\mathrm{TS}}_0$.

```
ggplot() +
  labs(x=bquote(theta), y="pdf") +
  geom_hline(yintercept=0) +
  geom_segment(data = data.frame(x=0.05, xend=0.05, y=0, yend=Inf),
               mapping = aes(x=x, xend=xend, y=y, yend=yend)) +
  coord_cartesian(xlim=c(-1,3), ylim=c(0,1))
```



Posterior

Now given likelihood and prior distributions, we can define the posterior distribution as proportional to the product of the prior and likelihood pdf's. Or it's usually better to take the log-transformation of prior and likelihood, add them, then exponentiate. For low-dimension problems like our toy model, we can approximate the evidence $p(y) = \int_{\Theta} p(y \mid \theta')p(\theta')d\theta'$ to ensure that the total volume under the surface is 1. Now here are the posterior distributions as they evolve over time as we collect more and more data given our likelihood function, simulated data, and 6 different priors:

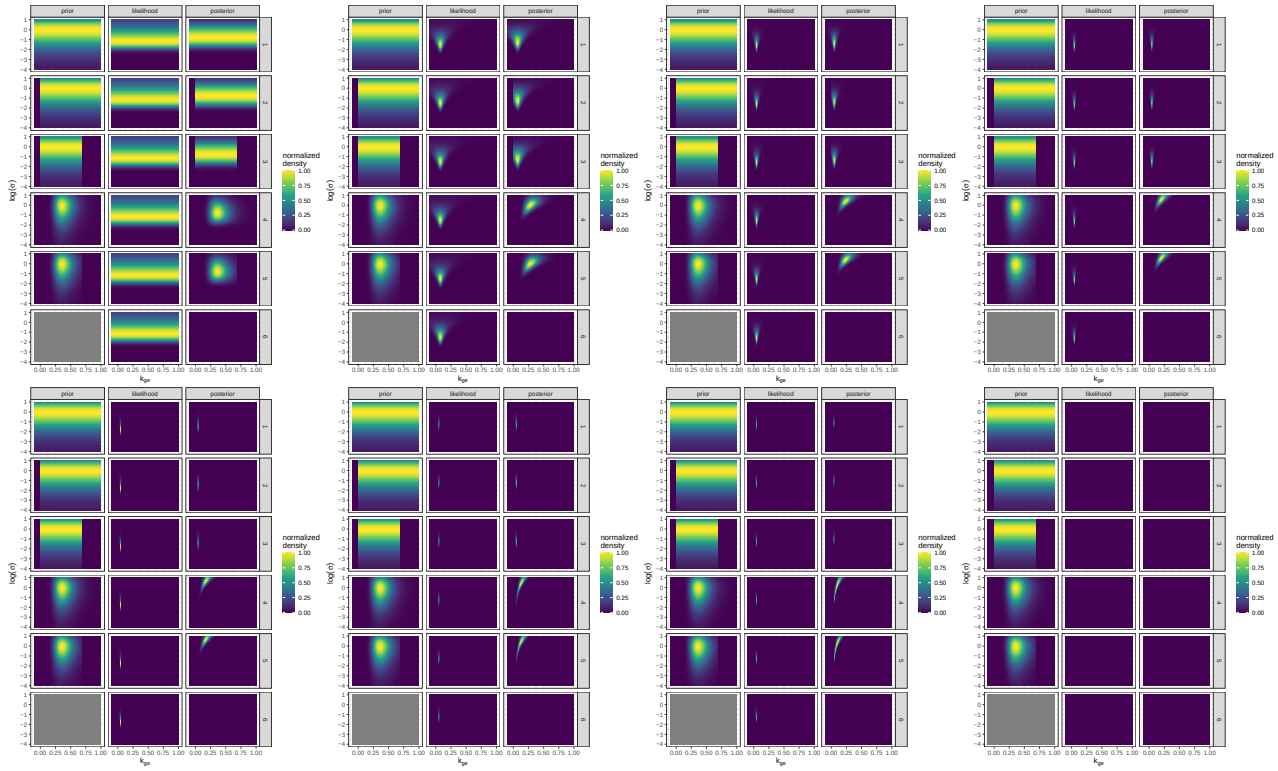
```
kge.min <- -0.1
kge.max <- 1
log.sigma.min <- -4
log.sigma.max <- 1
```

```r
grid.size <- 201

toy.model.dist <- sim.data %>%
  # fix initial condition (TS0)
  mutate(j=row_number(), TS0 = 1) %>%
  # create grid of parameters kge & sigma
  expand_grid(kge = seq(kge.min, kge.max, length.out=grid.size),
              log.sigma = seq(log.sigma.min, log.sigma.max, length.out=grid.size)) %>%
  # compute log-likelihood (LL) for each observation
  rowwise() %>%
  mutate(LL = dlnorm(y, meanlog=log(TS0)+kge*time, sdlog=exp(log.sigma), log=T)) %>%
  ungroup() %>%
  # by number of observations
  expand_grid(n.obs = 1:13) %>%
  filter(j<=n.obs) %>%
  # sum up LL across all observations
  group_by(n.obs, kge, log.sigma) %>%
  summarise(LL = sum(LL)) %>%
  # compute log priors
  mutate(log.prior.kge.1 = 0,
         log.prior.kge.2 = if_else(kge>0, 0, -Inf),
         log.prior.kge.3 = dunif(kge, min=0, max=log(2), log=T),
         log.prior.kge.4 = dlnorm(kge, meanlog=log(0.4), sdlog=0.3, log=T),
         log.prior.kge.5 = EnvStats::dlnormTrunc(kge, meanlog=log(0.4), sdlog=0.3,
                                                 min=0, max=log(2)) %>% log(),
         log.prior.kge.6 = if_else(kge==0.05, 0, -Inf),
         # sigma is exponentially distributed with rate parameter lambda=1
         log.prior.log.sigma = log(1*exp(log.sigma-1*exp(log.sigma)))) %>%
  # make log.prior.kge into a single column
  pivot_longer(cols = contains("log.prior.kge."),
               names_to = "prior",
               values_to = "log.prior.kge",
               names_prefix = "log.prior.kge.",
               names_transform = as.numeric) %>%
  mutate(log.prior = log.prior.kge + log.prior.log.sigma) %>%
  # compute non-normalized posterior for each observation
  mutate(log.non.norm.posterior = log.prior + LL) %>%
  # approximate evidence
  ungroup() %>%
  mutate(dA = (kge.max-kge.min)*(log.sigma.max-log.sigma.min)/grid.size^2,
         log.evidence = sum(dA*exp(log.non.norm.posterior)) %>% log()) %>%
  # normalize posterior (volume = 1)
  mutate(log.posterior = log.non.norm.posterior - log.evidence) %>%
  # normalize posterior (peak = 1)
  group_by(n.obs, prior) %>%
  mutate(log.prior.norm = log.prior - max(log.prior),
         LL.norm = LL - max(LL),
         log.posterior.norm = log.posterior - max(log.posterior)) %>%
  mutate(log.posterior.norm = case_when(is.na(log.posterior.norm) ~ -Inf,
                                        T ~ log.posterior.norm)) %>%
  # make distribution types into a single column
  pivot_longer(cols=-c(n.obs, prior, kge, log.sigma, dA, log.evidence),
               names_to="dist", values_to="log.density") %>%
```

```
  ungroup()
```

```
## `summarise()` has grouped output by 'n.obs', 'kge'. You can override using the
## `.groups` argument.
```

```
numbers.of.observations <- c(1:7,13)
p <- list()
for (n in numbers.of.observations) {
  p[[n]] <- toy.model.dist %>%
    filter(n.obs==n) %>%
    mutate(dist = factor(dist,
                         levels = c("log.prior.norm", "LL.norm", "log.posterior.norm"),
                         labels = c("prior", "likelihood", "posterior"))) %>%
    drop_na(dist) %>%
    ggplot() +
    facet_grid(prior~dist) +
    scale_fill_viridis_c() +
    labs(x=bquote(k[ge]), y=bquote(log(sigma)), fill="normalized\ndensity") +
    geom_tile(aes(x=kge, y=log.sigma, fill=exp(log.density)))
}
p[[1]]
p[[2]]
p[[3]]
p[[4]]
p[[5]]
p[[6]]
p[[7]]
p[[13]]
```



Because we're only interested here in k_{ge} and not in the residual error parameter σ, we can marginalize it out

as a "nuissance" parameter using the law of total probability:

$$p(k_{ge} \mid y) = \int_{-\infty}^{\infty} p(k_{ge}, \sigma' \mid y) d\sigma'$$

```
toy.model.dist.marginal <- toy.model.dist %>%
  mutate(dist = factor(dist,
                       levels = c("log.prior", "LL", "log.posterior"),
                       labels = c("prior", "likelihood", "posterior"))) %>%
  group_by(across(-c(log.sigma, log.density))) %>%
  drop_na(dist) %>%
  summarise(log.density = sum(exp(log.density) *
                              (log.sigma.max-log.sigma.min)/grid.size) %>%
              log()) %>%
  ungroup() %>%
  # make AUC = 1
  group_by(n.obs, dist, prior) %>%
  mutate(C = log(sum(exp(log.density)) * (kge.max-kge.min)/grid.size),
         log.density.true = log.density - C) %>%
  ungroup()
```

```
## `summarise()` has grouped output by 'n.obs', 'kge', 'prior', 'dA',
## 'log.evidence'. You can override using the `.groups` argument.
```

```
toy.model.dist.marginal %>%
  filter(is.element(n.obs, c(1:7, 13))) %>%
  ggplot() +
  labs(x=bquote(k[ge]), y="pdf", color="distribution") +
  facet_wrap(prior~n.obs, scales="free_y", nrow=6) +
  scale_x_continuous(breaks=c(0, 0.5, 1)) +
  geom_line(aes(kge, exp(log.density.true), group=dist, color=dist))
```

```
## Warning: Removed 3216 rows containing missing values or values outside the scale range
## (`geom_line()`).
```

Estimating our quantity of interest: summarizing the posterior

Great! Now we've characterized the posterior distribution of our quantity of interest $p(\theta \mid y)$! But now in applications, we need some statistics to help us in making decisions, such as model selection, whether or not to treat, etc. Many useful statistics are broadly classified according to measures of central tendency and uncertainty. Additionally, we can even look at higher moments, such as skew and kurtosis, but these aren't usually of great interest.

Central tendency

Examples of statistics measuring central tendency include the mean (i.e., expected value, first moment), median, and mode (i.e., maximum *a posteriori* (MAP)).

estimate	formula
mean	$\mu = E[\theta \mid y] = \int_\Theta \theta p(\theta \mid y) d\theta$
median	$\arg_\theta \left[\int_\Theta \theta p(\theta \mid y) d\theta = \frac{1}{2}\right]$
mode	$\arg\max_\theta p(\theta \mid y)$

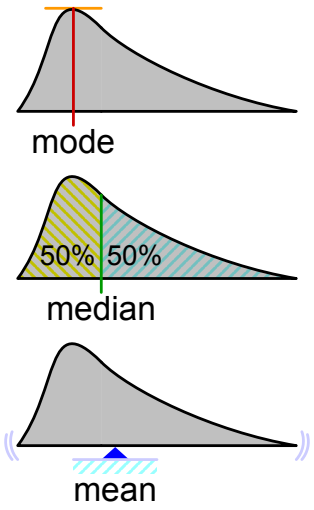


Figure 3: Central tendency (Wikipedia)

```
toy.model.dist.summary <- toy.model.dist %>%
  group_by(n.obs, prior, dist, dA) %>%
  summarise(mean.kge = sum(kge*exp(log.density)*dA),
            mode.kge = kge[log.density==max(log.density)] %>% mean(),
            mean.log.sigma = sum(log.sigma*exp(log.density)*dA),
            mode.log.sigma = log.sigma[log.density==max(log.density)] %>% mean()) %>%
  ungroup() %>%
  mutate(dist = factor(dist,
                       levels = c("log.prior.norm", "LL.norm", "log.posterior.norm"),
                       labels = c("prior", "likelihood", "posterior"))) %>%
  drop_na(dist)
```

```
## `summarise()` has grouped output by 'n.obs', 'prior', 'dist'. You can override
## using the `.groups` argument.
```

```
p[[1]] + geom_point(data = toy.model.dist.summary %>% filter(n.obs==1),
                    mapping = aes(x=mode.kge, y=mode.log.sigma),
                    shape = "+", color = "red")
```

```
## Warning: Removed 1 row containing missing values or values outside the scale range
## (`geom_point()`).
```

```
p[[3]] + geom_point(data = toy.model.dist.summary %>% filter(n.obs==3),
                    mapping = aes(x=mode.kge, y=mode.log.sigma),
                    shape = "+", color = "red")
```

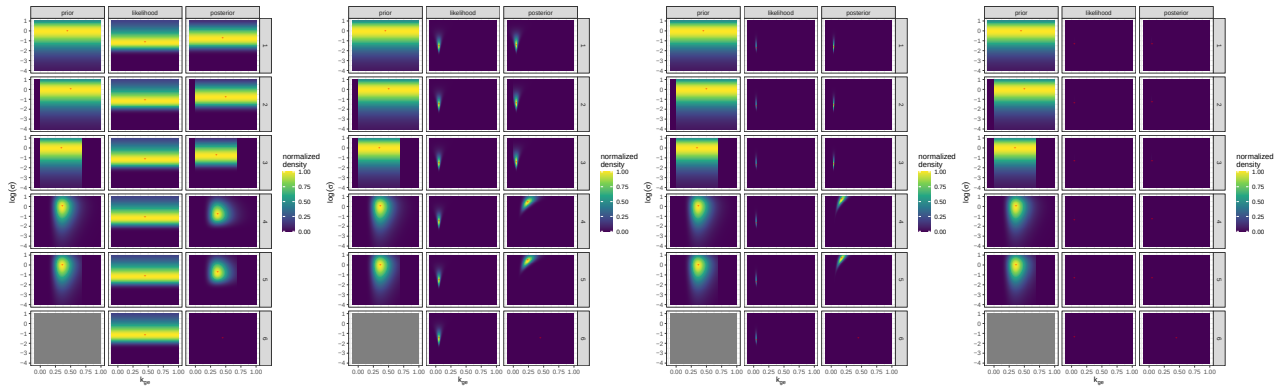
```
## Warning: Removed 1 row containing missing values or values outside the scale range
## (`geom_point()`).
```

```
p[[5]] + geom_point(data = toy.model.dist.summary %>% filter(n.obs==5),
                    mapping = aes(x=mode.kge, y=mode.log.sigma),
                    shape = "+", color = "red")
```

```
## Warning: Removed 1 row containing missing values or values outside the scale range
## (`geom_point()`).
```

```
p[[13]] + geom_point(data = toy.model.dist.summary %>% filter(n.obs==13),
                     mapping = aes(x=mode.kge, y=mode.log.sigma),
                     shape = "+", color = "red")
```

```
## Warning: Removed 1 row containing missing values or values outside the scale range
## (`geom_point()`).
```



Uncertainty

Examples of statistics measuring uncertainty include the variance (i.e., second moment) and credible intervals. Credible intervals come in all sorts of flavors. HDI, ETI, BCI, SPI, SI.

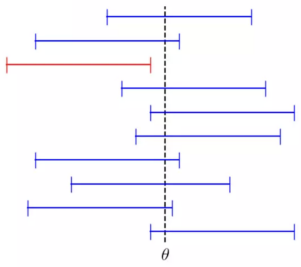
Confidence intervals

estimate	formula
variance	$\sigma^2 = E\left[(\theta - \mu)^2 \mid y\right]$
smallest credible interval/region (SCI/R) / highest density interval/region (HDI/R)	
quantile-based credible intervals	$(\mu - Z_{1-\frac{\alpha}{2}} \cdot \sigma, \mu + Z_{1-\frac{\alpha}{2}} \cdot \sigma)$

Maximum likelihood (posterior) approach

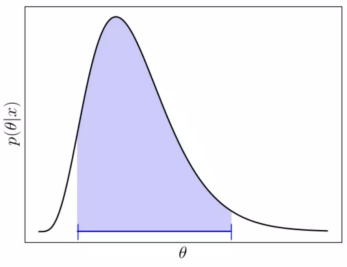
One approach in obtaining an estimate for $k_{ge} \mid y$ is to take the mode of the posterior distribution: the maximum *a posteriori* (MAP). You will also often hear of the maximum likelihood (ML) approach, which is a special case of MAP, but when considering a non-informative, flat prior, so that $p(\theta \mid y) \propto p(y \mid \theta)$. Recall that our log-likelihood function (after multiplying by -2, for convenience):

$$
\begin{aligned}
-2LL &= -2\ln(p(y \mid k_{ge})) \\
&= -2\sum_{j=1}^{m} \ln(p(y_j \mid k_{ge})) \\
&= \overbrace{m\ln(2\pi\sigma^2) + 2\sum_{j=1}^{m} \ln y_j}^{\text{additive constant w.r.t. } k_{ge}} + \overbrace{\frac{1}{\sigma^2}}^{\substack{\text{multiplicative} \\ \text{constant} \\ \text{w.r.t. } k_{ge}}} \cdot \overbrace{\sum_{j=1}^{m} \left(\ln y_j - \ln\left[\mathrm{TS}\left(t_j \mid k_{ge}; \hat{\mathrm{TS}}_0\right)\right]\right)^2}^{\text{sum of squared errors (SSE)}}
\end{aligned}
$$



90% Confidence Interval

- Frequentist approach
- θ is fixed, but unknown
- $X_1, ..., X_n$ are drawn from F_θ 10 times
- Build interval for each $(X_1, ..., X_n)$
- 9/10 of the intervals contain the true θ



90% Credible Interval

- Bayesian approach
- Associate to θ a probability measuring our *belief*
- $x_1, ..., x_n$ are fixed observations
- update posterior *belief* on θ
- Build interval containing θ with probability $= 90\%$

Figure 4: Credible intervals versus confidence intervals

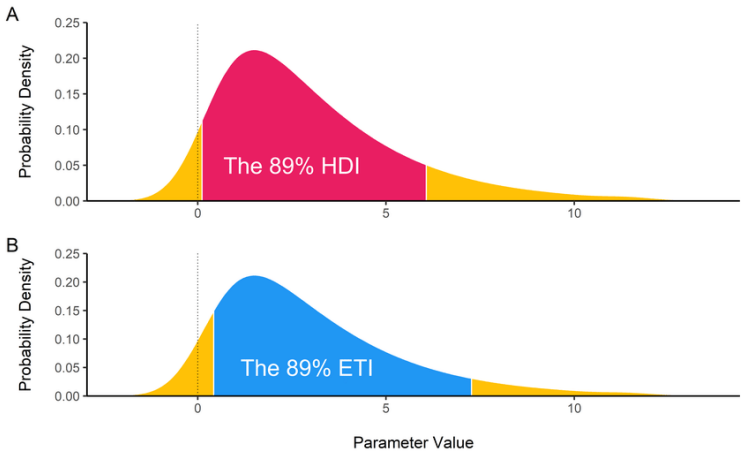


Figure 5: Credible intervals

Central tendency

Now, if we assume no prior information, we can get a point estimate by minimizing $-2LL$ by setting $\frac{\partial[-2LL]}{\partial k_{ge}} \mid_{k_{ge}=\hat{k}_{ge}} = 0$. We can attempt to solve this analytically, or we can solve this numerically. Here, we'll focus on the numerical solution. Notice that the first few terms are constants with respect to our parameter of interest k_{ge}. Dropping these terms then gives us the log-transformed sum of squared errors (SSE) as our objective function:

$$\hat{k}_{ge} = \arg\min_{k_{ge}} \sum_{j=1}^{m} \left(\ln y_j - \ln \left[\mathrm{TS}\left(t_j \mid k_{ge}; \mathrm{T}\hat{\mathrm{S}}_0 \right) \right] \right)^2$$

The MLE for σ will then be $\hat{\sigma} = RMSE = \sqrt{\frac{SSE(\hat{k}_{ge})}{m}}$, where RMSE is the log-transformed root mean squared error. Note that if we had chosen our observations to be normally distributed as opposed to log-normally distributed, we would end up with performing an ordinary least squares (OLS) regression with $\hat{\sigma} = RMSE = \sqrt{\frac{\left(y_j - \mathrm{TS}(t_j; \hat{k}_{ge}, \mathrm{T}\hat{\mathrm{S}}_0)\right)^2}{m}}$ (proof left as an exercise to the interested student). This shows that behind one's choice of an objective function lies an implied observation model. In the Bayesian framework, we make this model explicit and derive the objective function.

Now this was for the likelihood function with an non-informative prior. If, for example, we take a log-normal prior $\ln(k_{ge}) \sim N(\ln(\mu_{k_{ge}}), \sigma_{k_{ge}})$, then we can update our objective function as follows:

$$-2\ln(p(k_{ge} \mid y)) = -2\sum_{j=1}^{m} \ln(\overbrace{p(y_i \mid k_{ge})}^{\text{likelihood}}) - 2\ln(\overbrace{p(k_{ge})}^{\text{prior}}) + \overbrace{C}^{\substack{\text{additive} \\ \text{constant}}}$$

$$= \overbrace{C + m\ln(2\pi\sigma^2) + 2\sum_{j=1}^{m} \ln y_j}^{\text{additive constant w.r.t. } k_{ge}} + \overbrace{\frac{1}{\sigma^2}}^{\substack{\text{multiplicative} \\ \text{constant} \\ \text{w.r.t. } k_{ge}}} \cdot \overbrace{\sum_{j=1}^{m} \left(\ln y_j - \ln \left[\mathrm{TS}\left(t_j \mid k_{ge}; \mathrm{T}\hat{\mathrm{S}}_0 \right) \right] \right)^2}^{\text{sum of squared errors (SSE)}}$$

$$+ \overbrace{2\ln(k_{ge}) + \ln(2\pi\sigma_{k_{ge}}^2) + \left(\frac{k_{ge} - \mu_{k_{ge}}}{\sigma_{k_{ge}}} \right)^2}^{\text{prior}}$$

$$\hat{k}_{ge} = \arg\min_{k_{ge}} \frac{1}{\sigma^2} \cdot \overbrace{\sum_{j=1}^{m} \left(\ln y_j - \ln \left[\mathrm{TS}\left(t_j \mid k_{ge}; \mathrm{T}\hat{\mathrm{S}}_0 \right) \right] \right)^2}^{\text{likelihood}} + \overbrace{2\ln(k_{ge}) + \ln(2\pi\sigma_{k_{ge}}^2) + \left(\frac{k_{ge} - \mu_{k_{ge}}}{\sigma_{k_{ge}}} \right)^2}^{\text{prior}}$$

Notice here that because we've added a prior, we can no longer ignore the multiplicative constant $\frac{1}{\sigma^2}$. We have to now explicitly take our residual error parameter σ into account. What this parameter essentially does here is give weight to the data relative to prior beliefs. If σ is big, then the data carries little information on parameter k_{ge}. On the contrary, if σ is small, then the data carries a lot of information on parameter k_{ge} and the prior becomes unimportant.

Uncertainty

Now what about getting an MLE for variance and credible intervals? Here, we can take the second derivative: $\frac{\partial^2[LL]}{\partial k_{ge}^2} \mid_{k_{ge}=\hat{k}_{ge}}$. If we assume that our parameter of interest k_{ge} is normally distributed, then we get the following equation for standard error (SE):

$$\text{se}(\hat{k}_{ge}) = \frac{1}{\sqrt{\frac{\partial^2[LL]}{\partial k_{ge}^2} \mid_{k_{ge}=\hat{k}_{ge}}}}$$

Still working under the assumption of a normal distribution, credible intervals can then be computed as:

$$\text{CI} = (\hat{k}_{ge} - Z_{1-\alpha/2} \cdot \text{se}(\hat{k}_{ge}) \ , \ \hat{k}_{ge} + Z_{1-\alpha/2} \cdot \text{se}(\hat{k}_{ge}))$$

where $Z_{1-\alpha/2}$ is the Z-score at a significance level of α. For $\alpha = 0.05$, we get $Z_{0.975} \approx 1.96$.

Above, we considered the parameter k_{ge}, but since we defined a log-normal prior distribution, it may be more useful to take the log transformation, do all the inference as above, and then exponentiate in the end.

Expectations approach

An alternative approach to defining an estimate is by considering its expected value:

$$\hat{k}_{ge} = E[\theta \mid y] = \int_{\Theta} \theta p(\theta \mid y) d\theta$$

Now we get this integral, which is usually intractable. As with the ML/MAP approach, we can solve this numerically. To do so, we can take samples $\theta^{(n)}$ for $n = 1 \dots N$ that are independent an identically distributed (i.i.d.) according to our target distribution $p(\theta \mid y)$. Then we can estimate the expected value as:

$$\hat{k}_{ge} = \hat{E}[\theta \mid y] = \frac{1}{N} \cdot \sum_{n=1}^{N} \theta^{(n)}$$

Now how do go about producing independent samples that identically follow the target distribution? Enter Markov Chain Monte Carlo (MCMC) methods! Unfortunately, we do not have time or space to go over MCMC methods. We plan on continuing this series in the next semester.