

Bayesian modelling using R + STAN = RSTSAN

[https://github.com/clobos/Seminario\\_STAN\\_UNESP](https://github.com/clobos/Seminario_STAN_UNESP)

Cristian Villegas, ESALQ/USP

UNESP Botucatu (29/08/2019)

Stan



## Brief introduction to Stan

Stan is a state-of-the-art platform for statistical modeling and high-performance statistical computation. Thousands of users rely on Stan for statistical modeling, data analysis, and prediction in the social, biological, and physical sciences, engineering, and business.

# Brief introduction to Stan

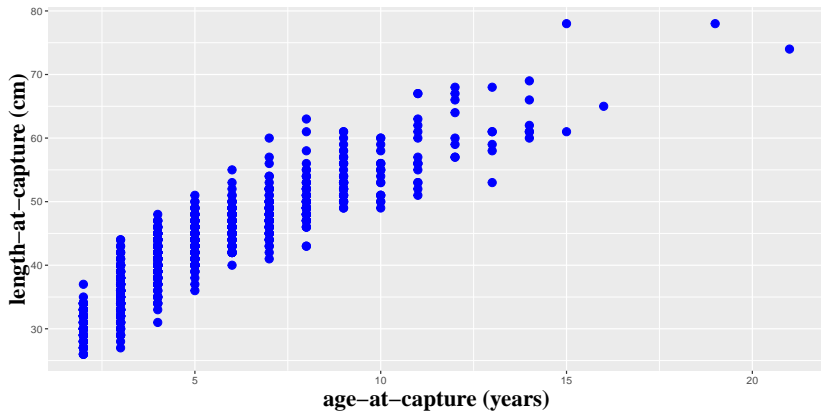
Users specify log density functions in Stan's probabilistic programming language and get:

- ▶ full Bayesian statistical inference with MCMC sampling (NUTS, HMC)
- ▶ approximate Bayesian inference with variational inference (ADVI)
- ▶ penalized maximum likelihood estimation with optimization (L-BFGS)

## Brief introduction to Stan

Stan's math library provides differentiable probability functions & linear algebra (C++ autodiff). Additional R packages provide expression-based linear modeling, posterior visualization, and leave-one-out cross-validation.

# Motivation Example



# The Von Bertalanffy growth model (Classical approach)

The Von Bertalanffy growth model is given by

$$Y_i = \beta_1 (1 - \exp\{-\beta_2 (x_i - \beta_3)\}) + \varepsilon_i \quad \text{with} \quad \varepsilon_i \sim N(0, \sigma^2),$$

where  $Y_i$  is the length-at-capture and  $x_i$  is the age-at-capture for the  $i$ th fish, respectively. The parameter interpretation of the Von Bertalanffy growth model is given by

- ▶  $\beta_1$  is the asymptotic (average) length
- ▶  $\beta_2$  is the growth rate coefficient (units are  $\text{years}^{-1}$ ), and
- ▶  $\beta_3$  represent the age when (average) length was zero.

# The Von Bertalanffy growth model (Classical approach)

In R code the Von Bertalanffy growth model is given by

```
LVB<-function(x,beta1, beta2, beta3){  
  beta1*(1-exp(-beta2*(x-beta3)))  
}
```



## The Von Bertalanffy growth model (Classical approach)

```
fit_LVB <- nls(Length ~ LVB(Age, beta1, beta2, beta3),  
start = list(beta1 = max(dados.amostra$Length),  
              beta2 = 0.5,  
              beta3= 0),  
data = dados.amostra)  
  
c(coef(fit_LVB),sigma=sigma(fit_LVB))
```

```
##          beta1          beta2          beta3          sigma  
## 73.8227693    0.1080882 -3.1551711    3.2475942
```

## First Case: Bayesian Inference with Stan code

```
Von_Bertanlanffy_mcmc <- '  
data {  
  int<lower = 0> N ;  
  vector[N] x ;  
  vector[N] y ;  
}  
parameters {  
  real<lower = .0> beta1 ;  
  real<lower = .0> beta2 ;  
  real beta3 ;  
  real<lower = .0> sigma ;  
}  
model {  
  y ~ normal(beta1*(1-exp(-beta2*(x-beta3))), sigma) ;  
}  
'
```

## Bayesian Inference: Stan code

```
fit_Von_Bertanlanffy <- stan(  
  model_code = Von_Bertanlanffy_mcmc,  
  data = list(N = nrow(dados.amostra),  
              x = dados.amostra$Age,  
              y = dados.amostra$Length),  
  chain = 3,  
  iter = 100,  
  warmup = 10,  
  thin = 1,  
  refresh=0)
```

## MCMC diagnostics using the bayesplot package

```
parameters<- c(paste('beta',1:3, sep=""), 'sigma')

CI_theta <- summary(ajuste_Von_Bertanlanffy,
                    pars = parameters,
                    probs = c(0.025, 0.975))$summary
print(round(CI_theta,2))
```

##		mean	se_mean	sd	2.5%	97.5%	n_eff	Rhat
##	beta1	25.45	13.46	21.95	0.37	50.18	2.66	2.30
##	beta2	2.46	1.88	2.31	0.54	5.70	1.52	17.00
##	beta3	0.84	0.30	0.45	0.35	1.91	2.28	1.97
##	sigma	10.32	5.17	12.87	0.90	42.48	6.20	1.78

# MCMC diagnostics using the bayesplot package

```
traceplot(ajuste_Von_Bertanlanffy, pars = parameters,  
          inc_warmup = TRUE)
```

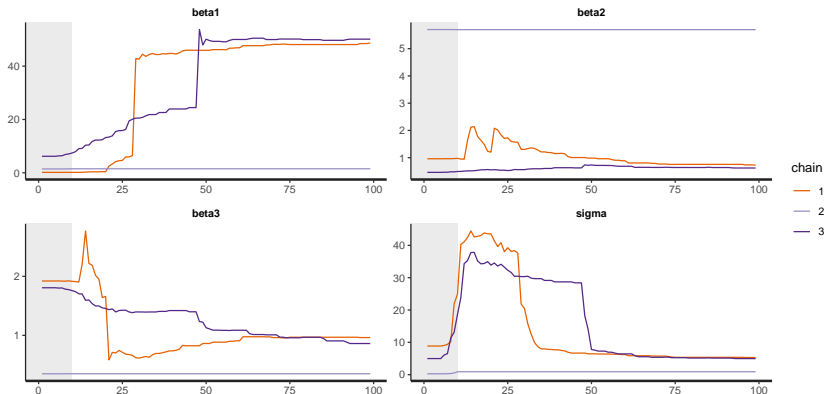


Figure 2: Traceplots for the Von Bertalanffy growth model

# MCMC diagnostics using the bayesplot package

```
mcmc_combo(mcmc_chain, pars = parameters, n_warmup=0)
```

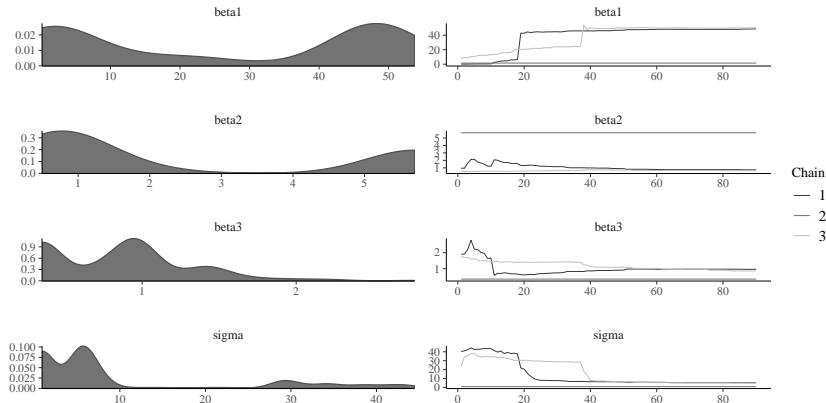


Figure 3: Posterior distributions and traceplots for the Von Bertalanffy growth model

# MCMC diagnostics using the bayesplot package

```
mcmc_pairs(mcmc_chain, pars = parameters)
```

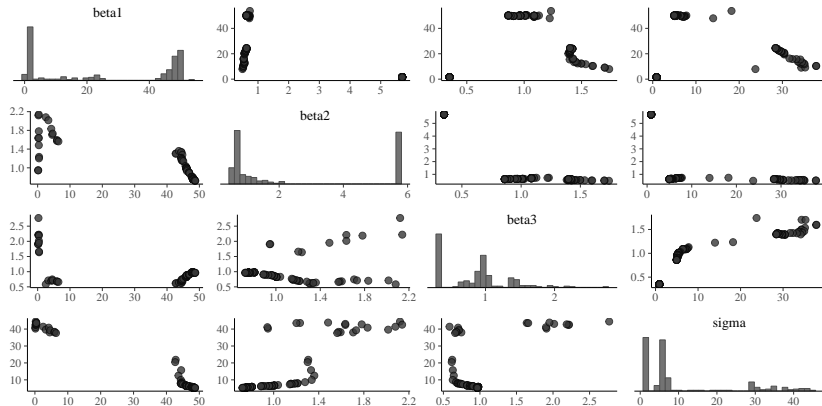


Figure 4: Scatterplots of MCMC draws for the Von Bertalanffy growth model

# Comparing fitted curves based on Classical and Bayesian Inference

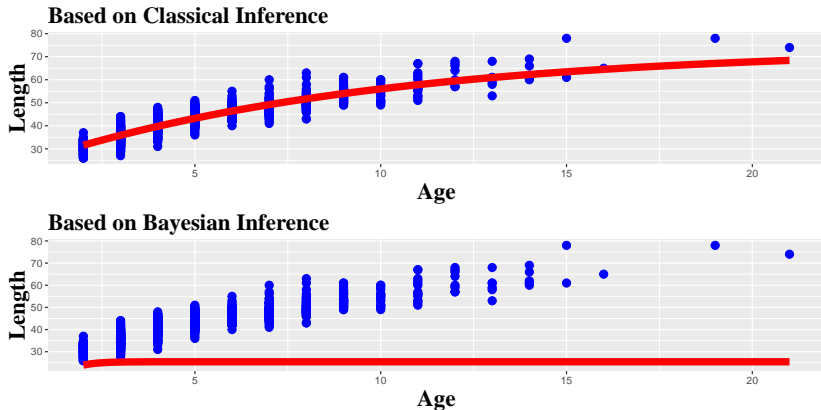


Figure 5: Fitted curves based on classical and bayesian Inference for the Von Bertalanffy growth model



## Comparing the parameter estimation based on Classical and Bayesian Inference

```
c(coef(fit_LVB),sigma=sigma(fit_LVB))
```

```
##          beta1          beta2          beta3          sigma
## 73.8227693    0.1080882  -3.1551711    3.2475942
```

```
print(CI_theta[,1])
```

```
##          beta1          beta2          beta3          sigma
## 25.4525584    2.4567863    0.8429762   10.3197269
```

## Second Case: Bayesian Inference with Stan code

```
Von_Bertanlanffy_mcmc <- '  
data {  
  int<lower = 0> N ;  
  vector[N] x ;  
  vector[N] y ;  
}  
parameters {  
  real<lower = .0> beta1 ;  
  real<lower = .0> beta2 ;  
  real beta3 ;  
  real<lower = .0> sigma ;  
}  
model {  
  y ~ normal(beta1*(1-exp(-beta2*(x-beta3))), sigma) ;  
}  
'
```

## Bayesian Inference: Stan code

```
fit_Von_Bertanlanffy <- stan(  
  model_code = Von_Bertanlanffy_mcmc,  
  data = list(N = nrow(dados.amostra),  
              x = dados.amostra$Age,  
              y = dados.amostra$Length),  
  chain = 3,  
  iter = 11000,  
  warmup = 1000,  
  thin = 10,  
  refresh=0)
```

## MCMC diagnostics using the bayesplot package

```
parameters<- c(paste('beta',1:3, sep=""), 'sigma')

CI_theta <- summary(ajuste_Von_Bertanlanffy,
                    pars = parameters,
                    probs = c(0.025, 0.975))$summary
print(round(CI_theta,2))
```

##		mean	se_mean	sd	2.5%	97.5%	n_eff	Rhat
##	beta1	74.09	0.05	2.54	69.63	79.49	2847.90	1
##	beta2	0.11	0.00	0.01	0.09	0.13	2885.81	1
##	beta3	-3.18	0.01	0.28	-3.76	-2.66	2932.61	1
##	sigma	3.25	0.00	0.07	3.11	3.40	3051.84	1

# MCMC diagnostics using the bayesplot package

```
traceplot(ajuste_Von_Bertanlanffy, pars = parameters,  
          inc_warmup = TRUE)
```

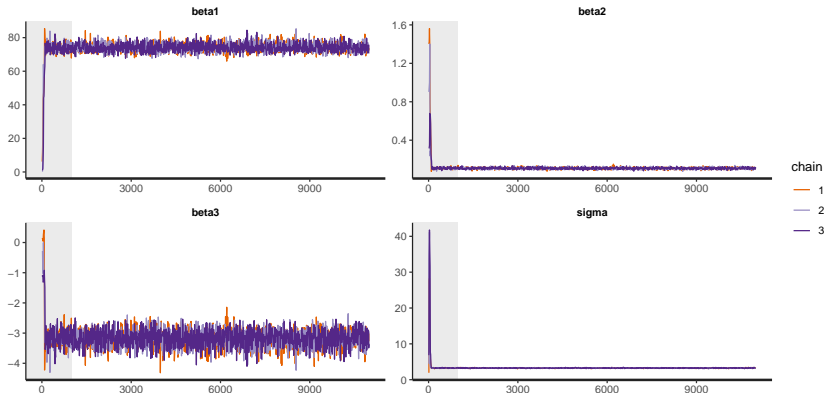


Figure 6: Traceplots for the Von Bertalanffy growth model

# MCMC diagnostics using the bayesplot package

```
mcmc_combo(mcmc_chain, pars = parameters, n_warmup=0)
```

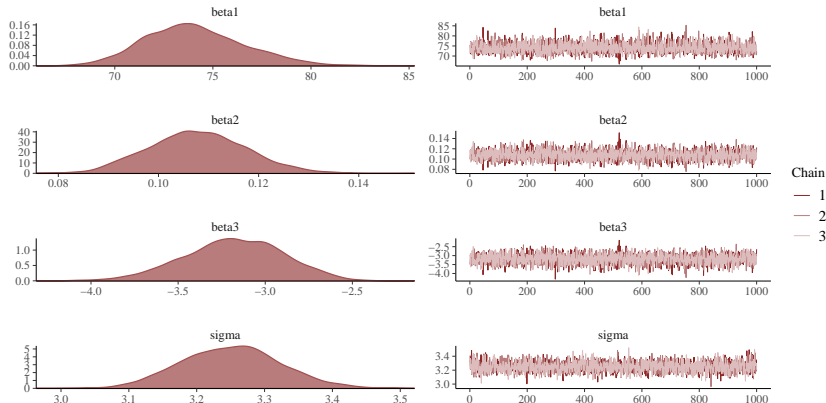


Figure 7: Posterior distributions and traceplots for the Von Bertalanffy growth model

# MCMC diagnostics using the bayesplot package

```
mcmc_pairs(mcmc_chain, pars = parameters)
```

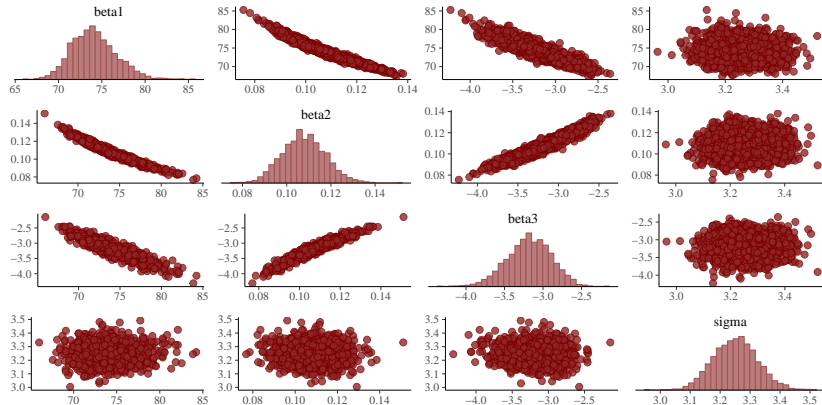


Figure 8: Scatterplots of MCMC draws for the Von Bertalanffy growth model

# Comparing fitted curves based on Classical and Bayesian Inference

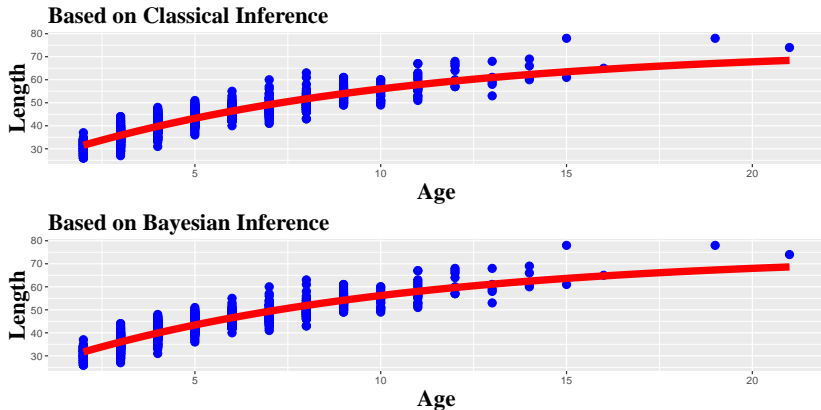


Figure 9: Fitted curves based on classical and bayesian Inference for the Von Bertalanffy growth model



## Comparing the parameter estimation based on Classical and Bayesian Inference

```
c(coef(fit_LVB),sigma=sigma(fit_LVB))
```

```
##          beta1          beta2          beta3          sigma
## 73.8227693    0.1080882   -3.1551711    3.2475942
```

```
print(CI_theta[,1])
```

```
##          beta1          beta2          beta3          sigma
## 74.0883836    0.1077937   -3.1803626    3.2509435
```

## Normal distribution without covariates in Stan

$Y \sim N(\mu, \sigma^2)$ . Therefore,  $\theta = (\mu, \sigma^2)^\top$ . Here, we specify the prior distribution for each parameter.

## Bayesian Inference: Stan code (specific prior)

```
normal_dist_example<- '  
data {  
  int<lower=0> N;  
  vector[N] y;  
}  
parameters {  
  real mu;  
  real<lower=0> sigma;  
}  
model {  
  y ~ normal(mu, sigma);  
  mu~ normal(0,1e6);  
  sigma ~ student_t(3,0,1);  
}  
,
```

## Bayesian Inference: Stan code (specific prior)

```
normal_dist_fit <- stan(model_code = normal_dist_example,  
  data = list(N = dim(stackloss)[1],  
    y = stackloss$stack.loss),  
  chain = 3,  
  iter = 11000,  
  warmup = 1000,  
  thin = 10,  
  refresh = 0)
```

## MCMC diagnostics using the bayesplot package

```
parameters<- c("mu", "sigma")  
  
CI_theta <- summary(normal_dist_fit,  
                    pars = parameters,  
                    probs = c(0.025, 0.975))$summary  
print(round(CI_theta,2))
```

##	mean	se_mean	sd	2.5%	97.5%	n_eff	Rhat
## mu	17.50	0.04	2.15	13.18	21.80	2970.46	1
## sigma	9.82	0.03	1.53	7.40	13.27	2973.68	1

# MCMC diagnostics using the bayesplot package

```
traceplot(normal_dist_fit, pars = parameters,  
          inc_warmup = TRUE)
```

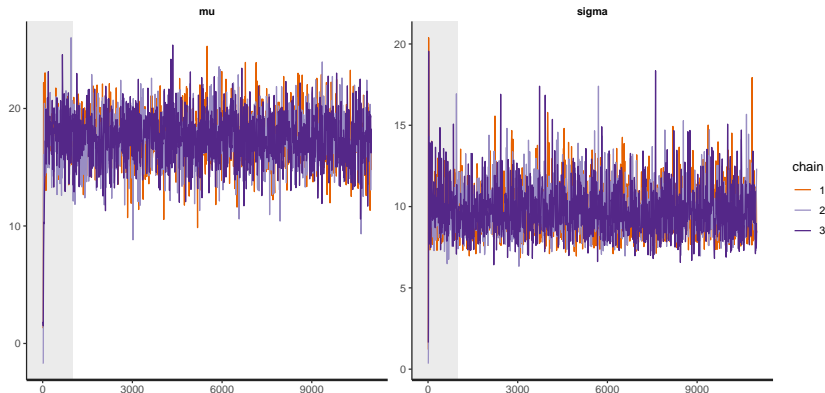


Figure 10: Traceplots for the Normal distribution

# MCMC diagnostics using the bayesplot package

```
mcmc_combo(mcmc_chain, pars = parameters, n_warmup=0)
```

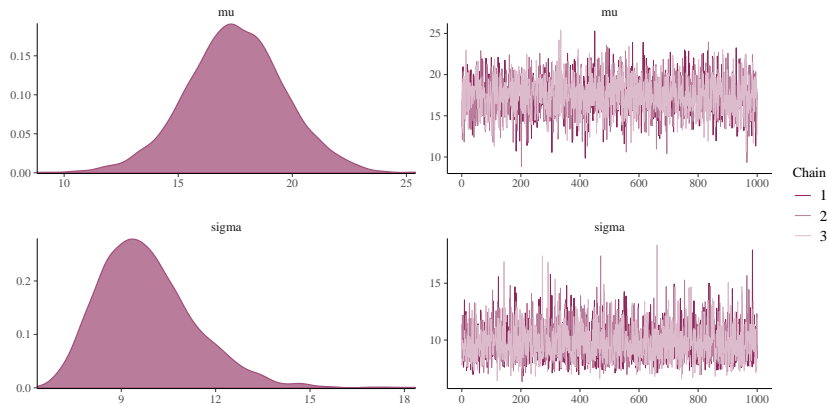


Figure 11: Posterior distributions and traceplots for the Normal distribution

# MCMC diagnostics using the bayesplot package

```
mcmc_pairs(mcmc_chain, pars = parameters)
```

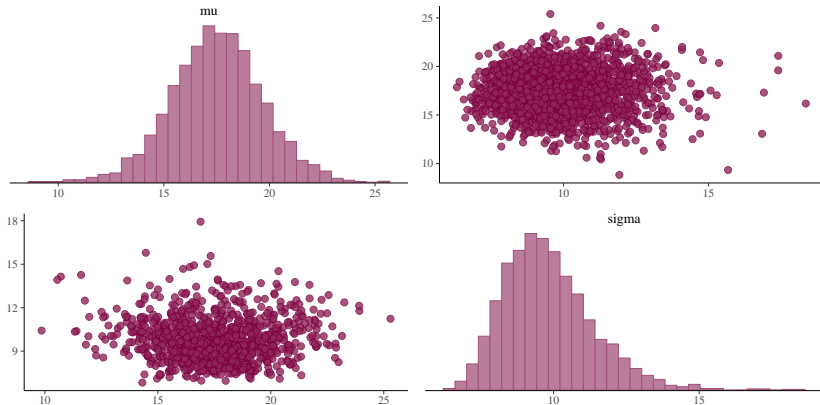


Figure 12: Scatterplots of MCMC draws for the Normal distribution



# Logistic Regression

Binomial response with logit link function. Here, we do not specify the prior distribution for each parameter.

# Motivation Example

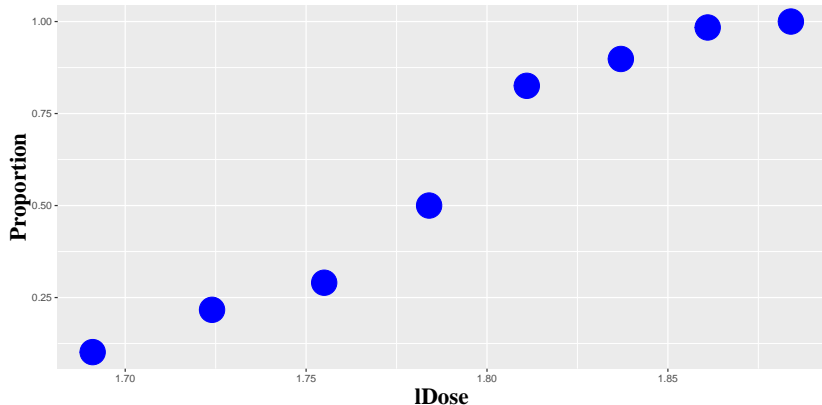


Figure 13: Scatterplot of proportions versus  $\log(\text{Dose})$

## Bayesian Inference: Stan code

```
logistic_example<- 'data {  
  int<lower=0> N;vector[N] x;  
  int<lower=0> y[N]; int<lower=0> n[N];  
}  
parameters {real beta1;real beta2;  
}  
transformed parameters {  
  real exp_eta[N]; real<lower=0, upper=1> prob[N];  
  for (i in 1:N) {exp_eta[i] = exp(beta1 + beta2*x[i]);  
  prob[i]= exp_eta[i]/(exp_eta[i] + 1);  
}  
}  
model {  
  y ~ binomial_logit(n, beta1 + beta2 * x);  
}  
,
```

## Bayesian Inference: Stan code

```
logistic_fit <- stan(model_code = logistic_example,  
data = list(N = dim(beetleDat)[1],  
            n = beetleDat$n,  
            x = beetleDat$lDose,  
            y = beetleDat$x),  
            chain = 3,  
            iter = 11000,  
            warmup = 1000,  
            thin = 10,  
            refresh=0)
```

## MCMC diagnostics using the bayesplot package

```
parameters<- c(paste('beta',1:2, sep=""))  
  
CI_theta <- summary(logistic_fit,  
                    pars = parameters,  
                    probs = c(0.025, 0.975))$summary  
print(round(CI_theta,2))
```

##		mean	se_mean	sd	2.5%	97.5%	n_eff	Rhat
##	beta1	-61.37	0.11	5.25	-72.01	-51.43	2420.70	1
##	beta2	34.64	0.06	2.95	29.07	40.58	2422.88	1

# MCMC diagnostics using the bayesplot package

```
traceplot(logistic_fit, pars = parameters,  
          inc_warmup = TRUE)
```

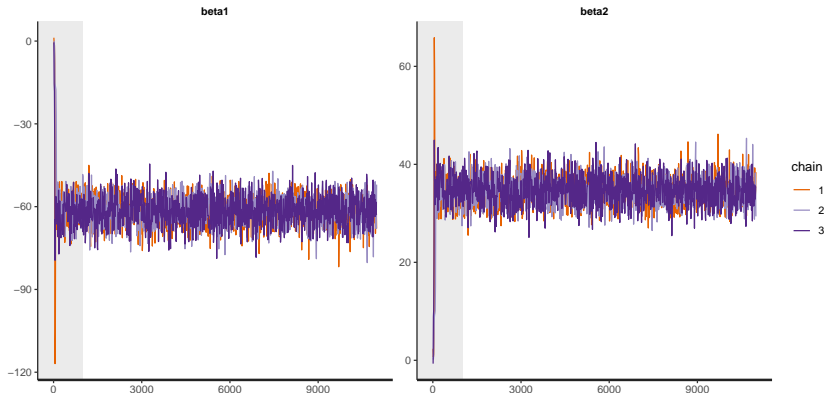


Figure 14: Traceplots for the Logistic regression model

# MCMC diagnostics using the bayesplot package

```
mcmc_combo(mcmc_chain, pars = parameters, n_warmup=0)
```

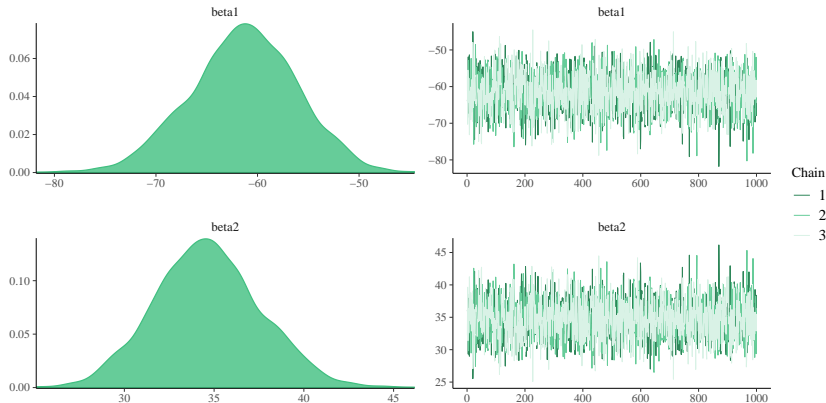


Figure 15: Posterior distributions and traceplots for the Logistic regression model

# MCMC diagnostics using the bayesplot package

```
mcmc_pairs(mcmc_chain, pars = parameters)
```

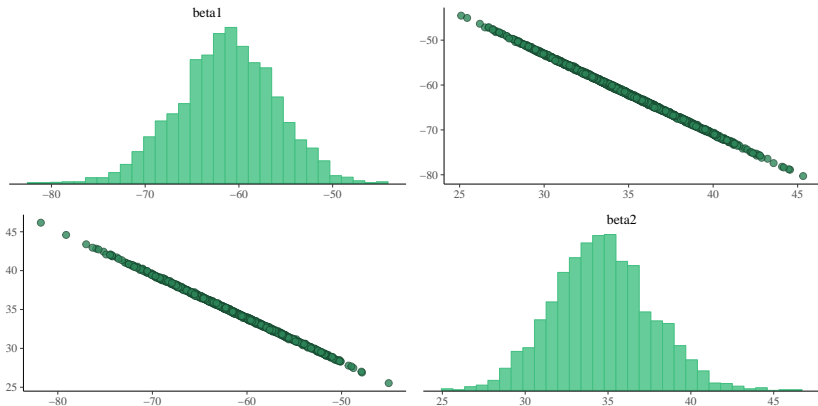


Figure 16: Scatterplots of MCMC draws for the Logistic Regression model

<https://www.youtube.com/watch?v=uSjsJg8fcwY>



# Fitted curve based on Bayesian Inference

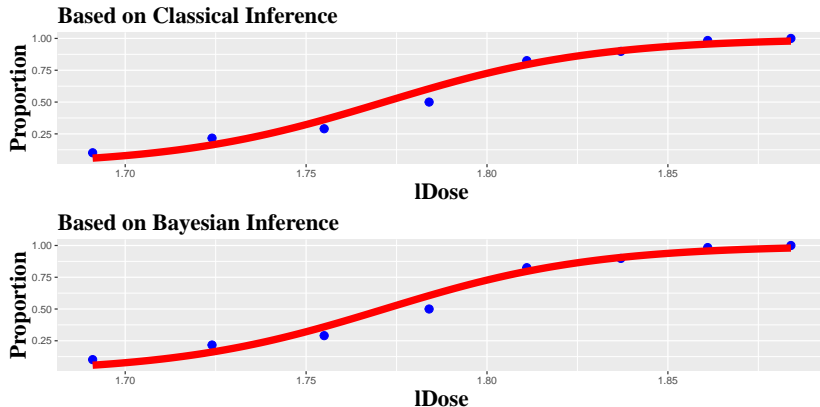


Figure 17: Fitted curves based on classical and bayesian Inference for the Logistic Regression model

## Comparing the parameter estimation based on Classical and Bayesian Inference

```
c(beta1=coef(fitLogistic)[1],beta2=coef(fitLogistic)[2])
```

```
## beta1.(Intercept)      beta2.lDose  
##           -60.74013           34.28593
```

```
print(CI_theta[,1])
```

```
##      beta1      beta2  
## -61.37130  34.64121
```

## Normal Linear Model

$$Y_i = \beta_1 + \beta_2 x_i + \varepsilon_i \quad \text{with} \quad \varepsilon_i \sim N(0, \sigma^2),$$

where  $Y_i$  is stack loss and  $x_i$  is flow of cooling air, respectively. Therefore  $\theta = (\beta_1, \beta_2, \sigma^2)^\top$ . Here, we do not specify the prior distribution for each parameter.

## Motivation Example

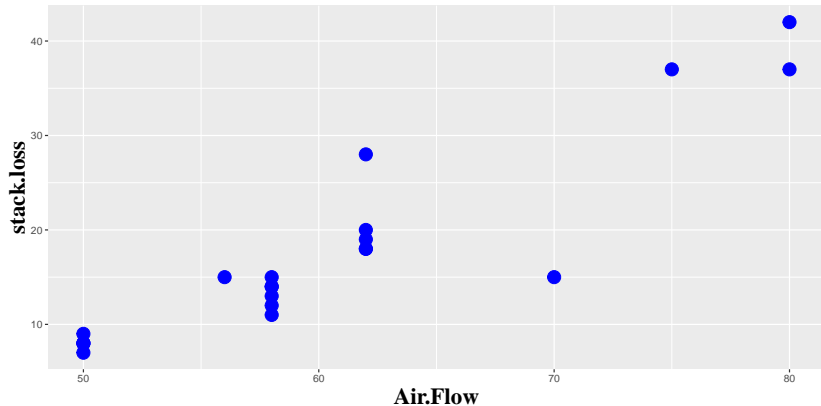


Figure 18: Scatterplot of stack.loss versus Air.Flow

## Bayesian Inference: Stan code

```
lm_example<- '  
data {  
  int<lower=0> N;  
  vector[N] x;  
  vector[N] y;  
}  
parameters {  
  real beta1;  
  real beta2;  
  real<lower=0> sigma;  
}  
model {  
  y ~ normal(beta1+beta2*x, sigma);  
}  
'
```

## Bayesian Inference: Stan code

```
lm_fit <- stan(model_code = lm_example,  
data = list(N = dim(stackloss)[1],  
            x = stackloss$Air.Flow,  
y = stackloss$stack.loss),  
            chain = 3,  
            iter = 11000,  
            warmup = 1000,  
            thin = 10,  
            refresh=0)
```

## MCMC diagnostics using the bayesplot package

```
parameters<- c(paste('beta',1:2, sep=""), 'sigma')

CI_theta <- summary(lm_fit,
                    pars = parameters,
                    probs = c(0.025, 0.975))$summary
print(round(CI_theta,2))
```

##		mean	se_mean	sd	2.5%	97.5%	n_eff	Rhat
##	beta1	-43.85	0.12	6.57	-56.83	-30.91	2816.31	1
##	beta2	1.02	0.00	0.11	0.80	1.23	2831.99	1
##	sigma	4.39	0.01	0.77	3.20	6.14	2976.60	1

# MCMC diagnostics using the bayesplot package

```
traceplot(lm_fit, pars = parameters, inc_warmup = TRUE)
```

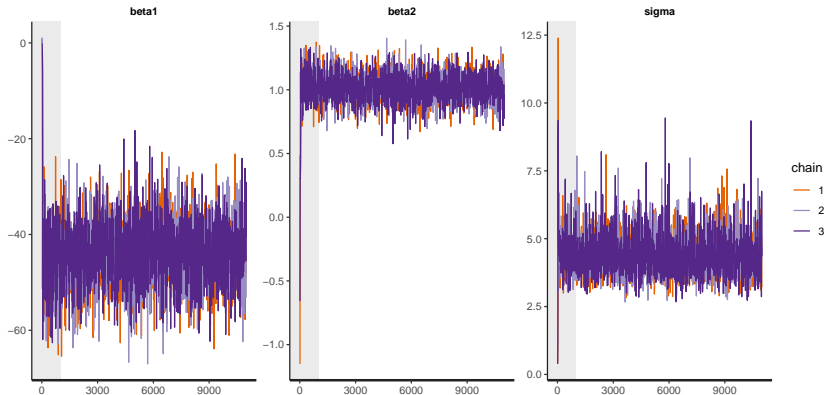


Figure 19: Traceplots for the Linear Normal



# MCMC diagnostics using the bayesplot package

```
mcmc_combo(mcmc_chain, pars = parameters, n_warmup=0)
```

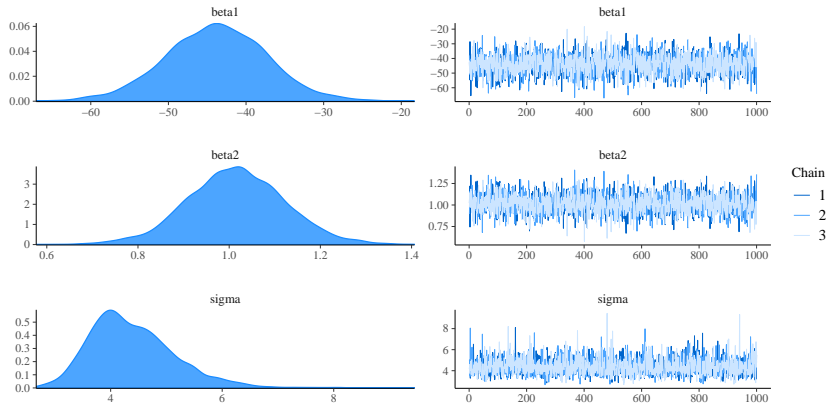


Figure 20: Posterior distributions and traceplots for the Linear Normal model

# MCMC diagnostics using the bayesplot package

```
mcmc_pairs(mcmc_chain, pars = parameters)
```

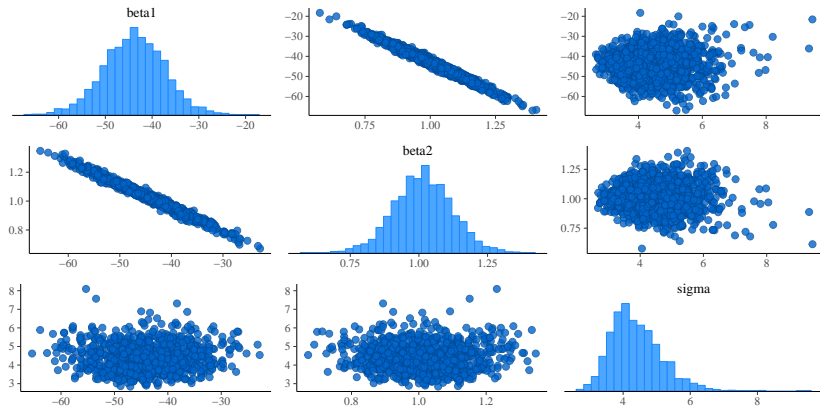


Figure 21: Scatterplots of MCMC draws for the Linear Normal model

## Comparing the parameter estimation based on Classical and Bayesian Inference

```
c(beta1=coef(fit)[1],beta2=coef(fit)[2],  
  sigma=sigma(fit))
```

##	beta1.(Intercept)	beta2.Air.Flow	sigma
##	-44.132025	1.020309	4.098242

```
print(CI_theta[,1])
```

##	beta1	beta2	sigma
##	-43.847882	1.015100	4.393619

# Comparing fitted curves based on Classical and Bayesian Inference

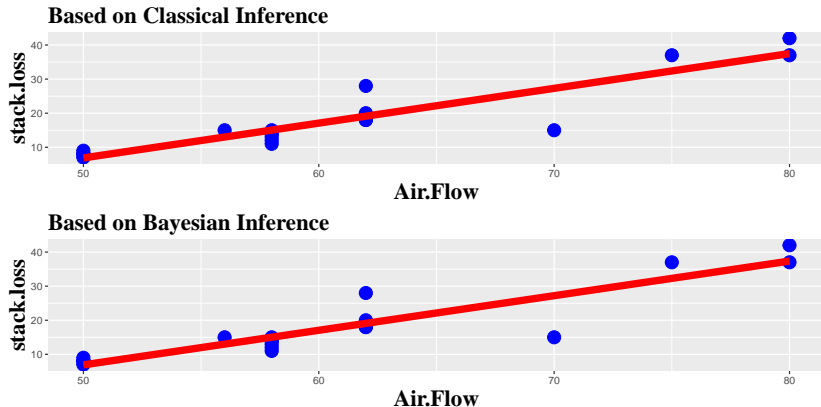


Figure 22: Fitted curves based on classical and bayesian Inference for the Normal Linear model

## More R packages based on Stan

- ▶ Bayesian Applied Regression Modeling via Stan: `rstanarm` R package.
- ▶ Interactive Visual and Numerical Diagnostics and Posterior Analysis for Bayesian Models `shinystan` R package.

Do I have more time for the shinystan r package?

```
rm(list=ls())  
load("modelo_Von_Bertanlanffy.RData")  
launch_shinystan(ajuste_Von_Bertanlanffy)
```

## References

- ▶ Baptiste Auguie (2017). gridExtra: Miscellaneous Functions for “Grid” Graphics. R package version 2.3.  
<https://CRAN.R-project.org/package=gridExtra>
- ▶ Jonah Gabry and Tristan Mahr (2018). bayesplot: Plotting for Bayesian Models. R package version 1.6.0.  
<https://CRAN.R-project.org/package=bayesplot>
- ▶ Jonah Gabry (2018). shinystan: Interactive Visual and Numerical Diagnostics and Posterior Analysis for Bayesian Models. R package version 2.5.0.  
<https://CRAN.R-project.org/package=shinystan>
- ▶ Stan Development Team (2018). RStan: the R interface to Stan. R package version 2.18.2. <http://mc-stan.org/>.

# References

- ▶ [https://mc-stan.org/docs/2\\_20/stan-users-guide/index.html](https://mc-stan.org/docs/2_20/stan-users-guide/index.html)
- ▶ [https://mc-stan.org/docs/2\\_20/reference-manual/index.html](https://mc-stan.org/docs/2_20/reference-manual/index.html)
- ▶ [https://mc-stan.org/docs/2\\_20/functions-reference/index.html](https://mc-stan.org/docs/2_20/functions-reference/index.html)