Imperial College London

MACHINE LEARNING
& GLOBAL HEALTH NETWORK

Introduction to Stan for applied Bayesian analyses.

Yu Chen and Oliver Ratmann

Overview today

1. A short introduction to the *Stan* language

2. Key points

By the end of this lecture and associated tutorial,

- you will understand the key components of the *Stan* language
- you will be able to implement and run simple Bayesian models in the *Stan* language

Key references

- Sections 1.1-1.3 of the Stan manual, `https://mc-stan.org/docs/2_22/stan-users-guide/regression-models.html`

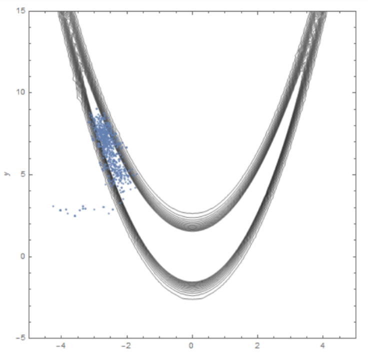
A short introduction to the *Stan* language

Stan:

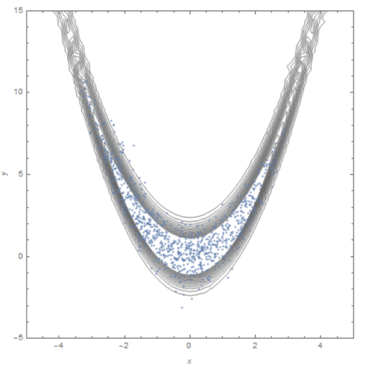
- is an open-source statistical inference software, which implements gradient-based MCMC to sample from posterior distributions.

Stan:

- is an open-source statistical inference software, which implements gradient-based MCMC to sample from posterior distributions.
- allows us to focus on statistical modelling, rather than implementing inference algorithms.
- algorithms are implemented in *C++*, and have an *R* interface (*rstan*).
- Bayesian models are written in text files, or using *R* syntax with the *rethinking* package.



off the shelf random walk MCMC



off the shelf Hamiltonian Monte Carlo

Use cases

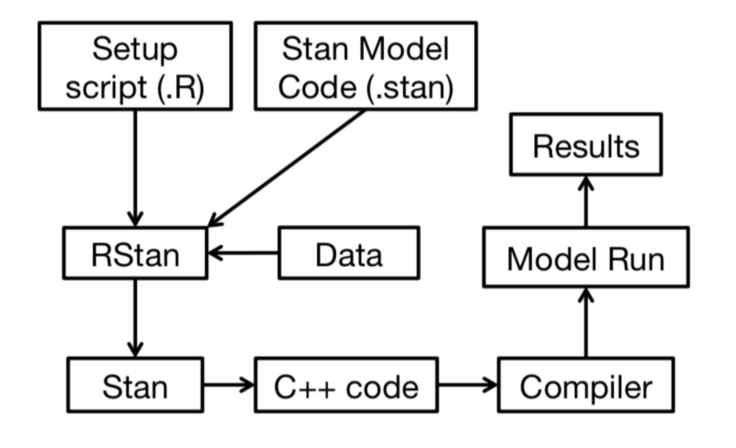
- Case Studies
  `https://mc-stan.org/users/documentation/case-studies.html`

Installation

Install *Stan* and *rstan*:

- `https://github.com/stan-dev/rstan/wiki/RStan-Getting-Started`

Using rstan


Setup script (.R)
Stan Model Code (.stan)
Results
RStan
Data
Model Run
Stan
C++ code
Compiler

Stan hello world

- Data: 100 continuous data points y_i
- Model:

$$\begin{aligned} y_i &\sim \mathcal{N}(\mu, \sigma^2) \\ \mu &\sim \mathcal{N}(0, 10^2) \\ \sigma &\sim \text{HalfCauchy}(0, 1) \end{aligned}$$

- Goal: estimate posterior density

$$p(\mu, \sigma | y) \propto p(y | \mu, \sigma) p(\mu) p(\sigma)$$

Stan hello world

Stan text model :

```
# specify Stan model
model1_text <- "
data{
    int<lower=1> N;
    real y[N];
}
parameters{
    real mu;
    real<lower=0> sigma;
}
model{
    sigma ~ cauchy( 0 , 1 );
    mu ~ normal( 0 , 10 );
    y ~ normal( mu , sigma );
}
"
```

Structure of Stan model files

- *data* block: specifies data required to fit the model

Structure of Stan model files

- *data* block: specifies data required to fit the model
- *transformed data* block: specifies temporary transformations of the data, e.g. QR decomposition of X; variables that do not change

Structure of Stan model files

- *data* block: specifies data required to fit the model
- *transformed data* block: specifies temporary transformations of the data, e.g. QR decomposition of X; variables that do not change
- *parameters* block: specifies all parameters that are fitted; cannot be assigned values directly

- *data* block: specifies data required to fit the model
- *transformed data* block: specifies temporary transformations of the data, e.g. QR decomposition of X; variables that do not change
- *parameters* block: specifies all parameters that are fitted; cannot be assigned values directly
- *transformed parameters* block: optional transformations of parameters, e.g. risk differences

Structure of Stan model files

- *data* block: specifies data required to fit the model
- *transformed data* block: specifies temporary transformations of the data, e.g. QR decomposition of X; variables that do not change
- *parameters* block: specifies all parameters that are fitted; cannot be assigned values directly
- *transformed parameters* block: optional transformations of parameters, e.g. risk differences
- *model* block: specifies the model in terms of likelihood and priors

Structure of Stan model files

- *data* block: specifies data required to fit the model
- *transformed data* block: specifies temporary transformations of the data, e.g. QR decomposition of X; variables that do not change
- *parameters* block: specifies all parameters that are fitted; cannot be assigned values directly
- *transformed parameters* block: optional transformations of parameters, e.g. risk differences
- *model* block: specifies the model in terms of likelihood and priors
- *generated quantities* block: quantities that depend on parameters and data, and do not affect inference

Stan hello world (2)

Compile and run Hameltonian Monte Carlo to obtain samples from joint posterior:

```
# compile Stan model
model1_compiled <- rstan::stan_model(
  model_name= 'model1',
  model_code = gsub('\t',' ',model1_text)
  )

#   Make data
set.seed(010680)    # use your birth date
y       <- rnorm(1e2, mean=0, sd=1)
stan_data <- list()
stan_data$N <- length(y)
stan_data$y <- y

# run Stan
model1_fit <- rstan::sampling(model1_compiled,
  data=stan_data,
  warmup=1e3,
    iter=4e3,
    chains=2,
    init=list( list(mu=1,sigma=2),list(mu=-1,sigma=0.5) )
  )
```

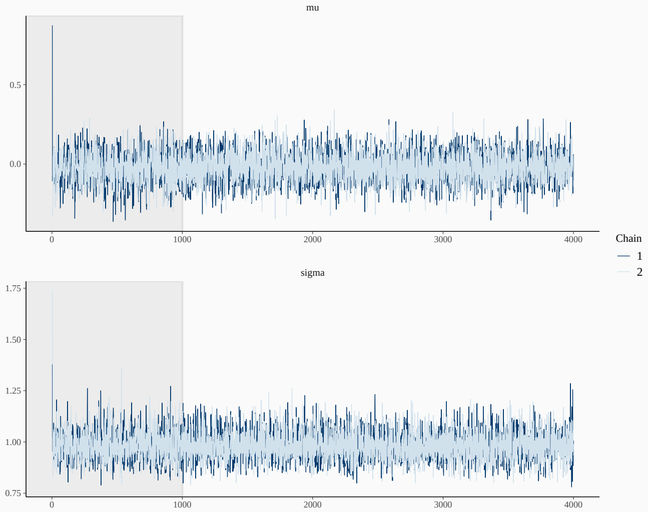
Process outputs:

```
# explore marginal posterior distributions, rhat, neff
print(run1$fit,digits=2)

# extract Monte Carlo samples including warmup
po <- rstan:::extract(run1$fit, inc_warmup = TRUE, permuted = FALSE)

# trace plots
bayesplot:::color_scheme_set("mix-blue-pink")
p <- bayesplot:::mcmc_trace(po,  pars = c("mu", "sigma"), n_warmup = 1e3,
        facet_args = list(nrow = 2, labeller = label_parsed))
pdf(file=file.path(out.dir,'normal_trace.pdf', w=10, h=8))
print(p)
dev.off()
```

Stan hello world (4)

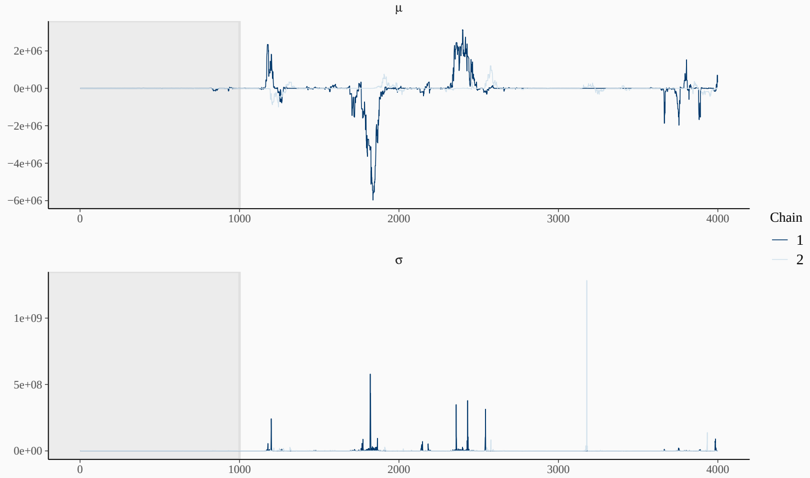


Stan with flat prior (1)

```
# data now TWO data points ONLY
y         <- c(-1,1)
stan_data <- list()
stan_data$N <- length(y)
stan_data$y <- y

# specify Stan model with flat prior on mu and sigma
model2_text <- "
data{
    int<lower=1> N;
    real y[N];
}
parameters{
    real mu;
    real<lower=0> sigma;
}
model{
    y ~ normal( mu , sigma );
}
"
```

Stan with flat prior (2)



Imperial College London

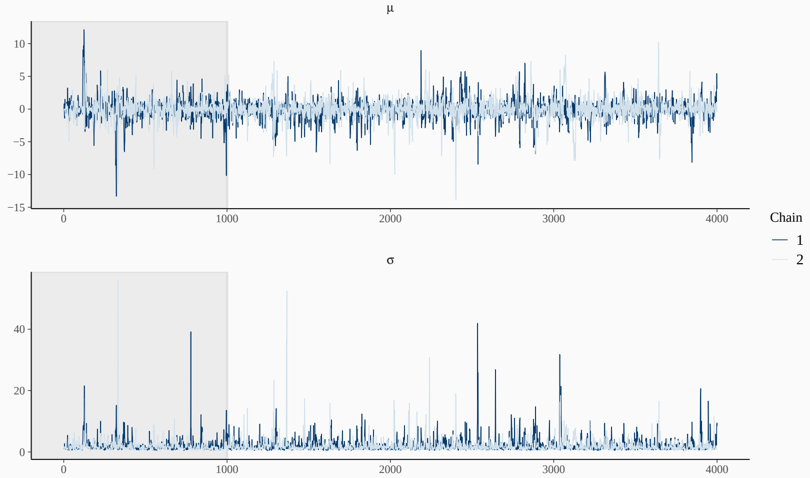
MACHINE LEARNING & GLOBAL HEALTH NETWORK

Stan with weakly informative prior (1)

```
# data now TWO data points ONLY
y        <- c(-1,1)
stan_data <- list()
stan_data$N <- length(y)
stan_data$y <- y

# specify Stan model with weakly informative prior on mu and sigma
model1_text <- "
data{
    int<lower=1> N;
    real y[N];
}
parameters{
    real mu;
    real<lower=0> sigma;
}
model{
    sigma ~ cauchy( 0 , 1 );
    mu ~ normal( 0 , 10 );
    y ~ normal( mu , sigma );
}
"
```

Stan with weakly informative prior (2)

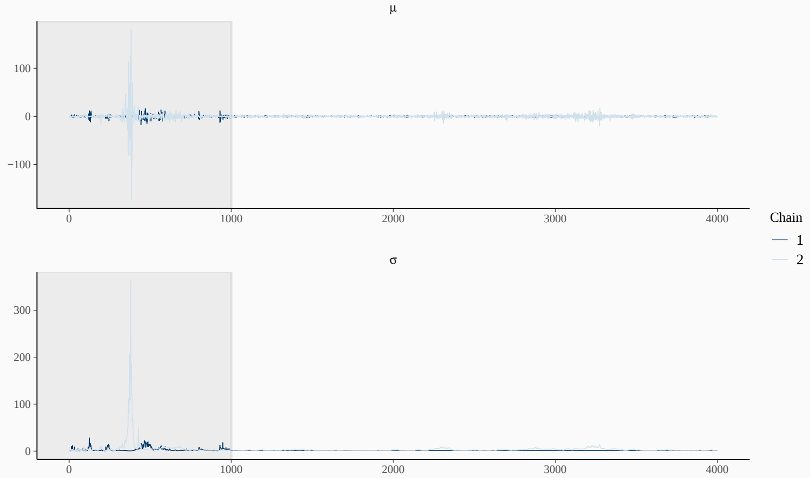


Stan with unidentifiable parameters and flat prior (1)

```
# data now TWO data points ONLY
y       <- c(-1,1)
stan_data <- list()
stan_data$N <- length(y)
stan_data$y <- y

# specify Stan model with unidentifiable parameters and flat prior
model3_text <- "
data{
    int<lower=1> N;
    real y[N];
}
parameters{
    real alpha1;
    real alpha2;
    real<lower=0> sigma;
}
transformed parameters{
    real mu= alpha1 + alpha2;
}
model{
    sigma ~ cauchy( 0 , 1 );
    y ~ normal( mu , sigma );
}
"
```

Stan with unidentifiable parameters and flat prior (2)

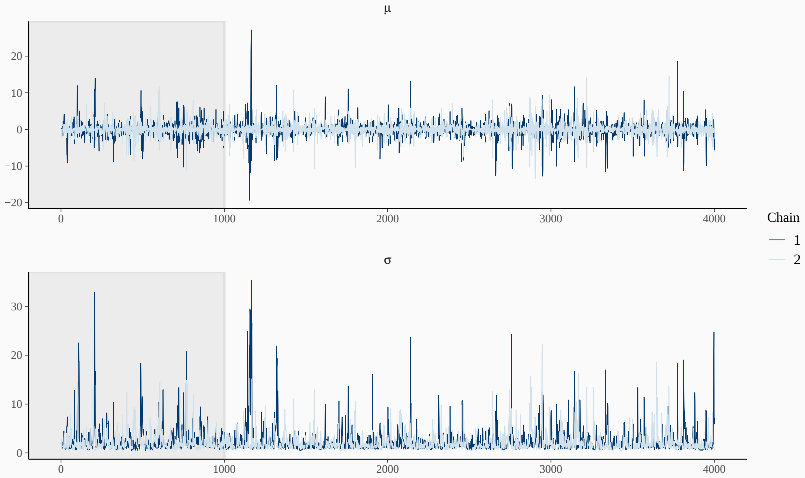


Stan with unidentifiable parameters and weakly informative prior (1)

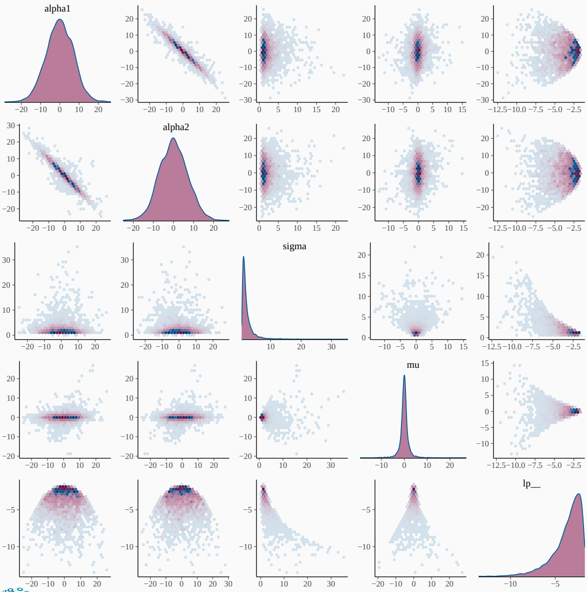
```r
# data now TWO data points ONLY
y        <- c(-1,1)
stan_data <- list()
stan_data$N <- length(y)
stan_data$y <- y

# specify Stan model with unidentifiable parameters and weakly informative prior
model4_text <- "
data{
    int<lower=1> N;
    real y[N];
}
parameters{
    real alpha1;
    real alpha2;
    real<lower=0> sigma;
}
transformed parameters{
    real mu= alpha1 + alpha2;
}
model{
    sigma ~ cauchy( 0 , 1 );
    alpha1 ~ normal(0, 10);
    alpha2 ~ normal(0, 10);
    y ~ normal( mu , sigma );
}
"
```

Stan with unidentifiable parameters and weakly informative prior (2)



Stan with unidentifiable parameters and weakly informative prior (3)



Key points

Stan.

- *Stan* provides a convenient interface to a Hamiltonian Monte Carlo sampler that can efficiently sample from high dimensional and highly correlated posterior distributions.
- Automated numerical inference frees time to focus on statistical modelling.

Further reading.

- Radford Neal, Handbook of Markov Chain Monte Carlo, Chapter 5 MCMC Using Hamiltonian Dynamics (2011).