Estiamting multivariate normal models by Stan I

Xulong Wang

```
library(rstan)
```

```
## Warning: package 'rstan' was built under R version 3.1.3

## Loading required package: Rcpp

## Warning: package 'Rcpp' was built under R version 3.1.3

## Loading required package: ggplot2

## Warning: package 'ggplot2' was built under R version 3.1.3

## rstan (Version 2.8.0, packaged: 2015-09-19 14:48:38 UTC, GitRev: 05c3d0058b6a)
## For execution on a local, multicore CPU with excess RAM we recommend calling
## rstan_options(auto_write = TRUE)
## options(mc.cores = parallel::detectCores())
```

```
rm(list = ls())
setwd("~/Dropbox/Github/BDA")
```

Models

```
stan_multi_norm <- "

data {
int<lower=1> K; # outcomes
vector[K] mu;
cov_matrix[K] Sigma;
vector[K] y;
}

parameters {
vector[K] beta;
}

model {
beta ~ multi_normal(mu, Sigma);
}

"

stan_multi_cholesky <- "

data {
```

```
int<lower=1> K; # outcomes
vector[K] mu;
cov_matrix[K] Sigma;
vector[K] y;
}

transformed data {
matrix[K, K] L;
L <- cholesky_decompose(Sigma);
}

parameters {
vector[K] beta;
}
model {
beta ~ multi_normal_cholesky(mu, L);
}
"
```

```
load("adsp.rdt")
mdata <- adsp$mdata
Sigma <- adsp$kinship$autosome
Sigma[Sigma < 0] <- 0
```

```
stan_multi_norm <- stan_model(model_code = stan_multi_norm)
stan_multi_cholesky <- stan_model(model_code = stan_multi_cholesky)
```

```
dat <- list(K = 576, mu = rep(0, 576), y = mdata$AD1, Sigma = Sigma)

fit_multi_norm <- sampling(stan_multi_norm, chain = 2, data = dat) # 450 sec each chain
```

```
##
## SAMPLING FOR MODEL '7af896306de3facfa0fe6c48f3b1745e' NOW (CHAIN 1).
##
## Chain 1, Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 1, Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 1, Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 1, Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 1, Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 1, Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 1, Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 1, Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 1, Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 1, Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 1, Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 1, Iteration: 2000 / 2000 [100%]  (Sampling)
## #  Elapsed Time: 235.652 seconds (Warm-up)
## #                236.004 seconds (Sampling)
## #                471.656 seconds (Total)
##
##
## SAMPLING FOR MODEL '7af896306de3facfa0fe6c48f3b1745e' NOW (CHAIN 2).
```

```
##
## Chain 2, Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 2, Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 2, Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 2, Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 2, Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 2, Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 2, Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 2, Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 2, Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 2, Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 2, Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 2, Iteration: 2000 / 2000 [100%]  (Sampling)
## #  Elapsed Time: 233.012 seconds (Warm-up)
## #                236.666 seconds (Sampling)
## #                469.678 seconds (Total)
```

```
fit_multi_cholesky <- sampling(stan_multi_cholesky, data = dat) # 170 sec each chain
```

```
##
## SAMPLING FOR MODEL '8045facf6befdac43ef90d7a420be55a' NOW (CHAIN 1).
##
## Chain 1, Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 1, Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 1, Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 1, Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 1, Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 1, Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 1, Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 1, Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 1, Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 1, Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 1, Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 1, Iteration: 2000 / 2000 [100%]  (Sampling)
## #  Elapsed Time: 113.454 seconds (Warm-up)
## #                114.831 seconds (Sampling)
## #                228.286 seconds (Total)
##
##
## SAMPLING FOR MODEL '8045facf6befdac43ef90d7a420be55a' NOW (CHAIN 2).
##
## Chain 2, Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 2, Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 2, Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 2, Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 2, Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 2, Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 2, Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 2, Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 2, Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 2, Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 2, Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 2, Iteration: 2000 / 2000 [100%]  (Sampling)
## #  Elapsed Time: 117.283 seconds (Warm-up)
```

```
## #                115.173 seconds (Sampling)
## #                232.456 seconds (Total)
##
##
## SAMPLING FOR MODEL '8045facf6befdac43ef90d7a420be55a' NOW (CHAIN 3).
##
## Chain 3, Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 3, Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 3, Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 3, Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 3, Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 3, Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 3, Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 3, Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 3, Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 3, Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 3, Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 3, Iteration: 2000 / 2000 [100%]  (Sampling)
## #  Elapsed Time: 116.1 seconds (Warm-up)
## #                115.038 seconds (Sampling)
## #                231.138 seconds (Total)
##
##
## SAMPLING FOR MODEL '8045facf6befdac43ef90d7a420be55a' NOW (CHAIN 4).
##
## Chain 4, Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 4, Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 4, Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 4, Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 4, Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 4, Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 4, Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 4, Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 4, Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 4, Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 4, Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 4, Iteration: 2000 / 2000 [100%]  (Sampling)
## #  Elapsed Time: 115.128 seconds (Warm-up)
## #                115.107 seconds (Sampling)
## #                230.235 seconds (Total)
```

multi_norm() is much slower than multi_normal_cholesky

How cholesky factoring speeding up MCMC sampling in a multivariate normal model?

In MCMC, a partial derivative step propagates through all parameters, which takes the most time

But what is it exactly?

Simulation to be continued ...

```
# (Sigma <- matrix(c(10, 3, 3, 2), 2, 2))
# y <- mvrnorm(n = 1e3, mu = rep(0, 2), Sigma)
# colMeans(y)
# var(y)
#
# dat <- list(K = 2, mu = c(0, 0), Sigma = Sigma)
# (fit_multi_norm <- sampling(stan_multi_norm, data = dat))
```