

# Multivariate Normal Example

## rmvnorm

---

```
// [[Rcpp::depends(RcppArmadillo)]]
#include <RcppArmadillo.h>

// [[Rcpp::export]]
arma::mat rmvnorm(int n, arma::vec mu, arma::mat Sigma) {
  unsigned int k = mu.n_elem;

  if (k != Sigma.n_rows || Sigma.n_rows != Sigma.n_cols) {
    Rcpp::stop("Bad dimensions");
  }

  arma::mat L = arma::chol(Sigma, "lower");
  arma::mat rnorm = arma::randn<arma::mat>(k, n);

  return (mu * arma::ones<arma::mat>(1,n) + L * rnorm).t();
}
```

```
Sigma = diag(1,2)
Sigma[1,2] = Sigma[2,1] = -0.95

d = rmvnorm(10000, rep(1,2), Sigma)
plot(d)
```

```
bench::mark(
  rmvnorm(10000, mu = rep(0, 1000), Sigma = diag(1,1000,1000)),
  MASS::mvrnorm(10000, mu = rep(0, 1000), Sigma = diag(1,1000,1000)),
  check=FALSE
)
```

## dmvnorm

---

```
// [[Rcpp::depends(RcppArmadillo)]]
#include <RcppArmadillo.h>

// [[Rcpp::export]]
arma::mat dmvnorm(arma::mat x, arma::vec mu, arma::mat Sigma, bool use_log = true) {
  //unsigned int n = mu.n_elem;
  unsigned int n_obs = x.n_cols;

  arma::mat X = x - mu * arma::ones<arma::mat>(1, n_obs);
  arma::mat L = arma::chol(Sigma, "lower");
```

```

arma::mat L_inv_t = arma::inv(arma::trimatl(L)).t();
arma::mat XL = X.t() * L_inv_t;
arma::vec XLLX = arma::sum(XL % XL, 1);

arma::vec L_diag = L.diag();

double norm = pow(2 * arma::datum::pi, n) * sqrt( arma::prod(arma::square(L_diag)));

if (use_log) {
  return -XLLX/2 - log(norm)/2;
} else {
  return arma::exp(-XLLX/2) / sqrt(norm);
}
}

```

```

dmvnorm(matrix(0, 1, 3), 0, matrix(1,1,1))

```