greta

simple and scalable statistical modelling in R

Nick Golding  greta-dev.github.io/greta
types of statistical software

one model
DirFactor, effectFusion

one class of model
glm, lme4, INLA

(almost) any model
BUGS, JAGS, Stan
types of statistical software

one model
  DirFactor, effectFusion

one class of model
  glm, lme4, INLA

(almost) any model
  BUGS, JAGS, Stan, greta
example

```r
x <- iris$Petal.Length
y <- iris$Sepal.Length

int = normal(0, 5)
coef = normal(0, 3)
sd = lognormal(0, 3)

mean <- int + coef * x
distribution(y) = normal(mean, sd)

m <- model(int, coef, sd)
draws <- mcmc(m, n_samples = 1000)
bayesplot::mcmc_trace(draws)
```
**Stan**

```stan
data {
  real alpha;
  real beta;
  real<lower=0> sigma2;
  int<lower=0> J;
  int y[J];
  vector[J] Z;
  int n[J];
}

transformed data {
  real<lower=0> sigma;
  sigma <- sqrt(sigma2);
}

parameters {
  real theta1;
  real theta2;
  vector[J] X;
}

model {
  real p[J];
  theta1 ~ normal(0, 32);  // 32^2 = 1024
  theta2 ~ normal(0, 32);
  X ~ normal(alpha + beta * Z, sigma);
  y ~ binomial_logit(n, theta1 + theta2 * X);
}
```

**JAGS**

```jags
for(j in 1 : J) {
  y[j] ~ dbin(p[j], n[j])
  X[j] ~ dnorm(mu[j], tau)
  mu[j] <- alpha + beta * Z[j]
}
theta[1] ~ dnorm(0.0, 0.001)
theta[2] ~ dnorm(0.0, 0.001)
```

**greta**

```r
theta = normal(0, 32, dim = 2)
mu <- alpha + beta * Z
X = normal(mu, sigma)
distribution(y) = binomial(n, p)
```
intuitive

\[
\text{int} = \text{normal}(0, 5) \\
\text{coef} = \text{normal}(0, 3) \\
\text{sd} = \text{lognormal}(0, 3)
\]

\[
\text{mean} \leftarrow \text{int} + \text{coef} \times x \\
\text{distribution}(y) = \text{normal}(\text{mean}, \text{sd})
\]
Get started with greta

Installation

You can install the stable version of greta from CRAN:

```r
install.packages("greta")
```

Alternatively, you can install the latest release, or the development version, from GitHub using the devtools package:

```r
devtools::install_github("greta-dev/greta")  # latest release
devtools::install_github("greta-dev/greta@dev")  # development version
```

```r
library(greta)
```

TensorFlow

Before you can fit models with greta, you will also need to have a working installation of Google's TensorFlow python package (version 1.0.0 or higher). greta exports `install_tensorflow()` from the tensorflow R package, which will attempt to install the latest version of TensorFlow from within your R session.

```r
install_tensorflow()
```
gradient-based inference
- automatic differentiation
- efficient linear algebra
- highly parallel
scalable probit regression with 50 predictors
types of statistical software

one model
  DirFactor, effectFusion

one class of model
  glm, lme4, INLA

(almost) any model
  BUGS, JAGS, Stan
deep learning software is less redundant

canned estimators

keras

tensorflow
extendable

**greta.dynamics**
a greta extension for modelling dynamical systems

- R ★ 1 ♦️ 2 Updated 12 days ago

**greta.gp**
a greta extension for Gaussian process modelling

- R ★ 4 ♦️ 1 Updated on 9 Sep

**greta.multivariate**
a greta extension for multivariate modelling

- ♦️ 1 Updated on 26 Aug

**greta.gam**
a greta extension for generalised additive modelling using mgcv

- R ★ 3 ♦️ 2 Updated on 23 Aug
extendable

Gaussian processes in greta

greta gp extends greta to let you define Gaussian processes as part of your model. It provides a syntax to create and combine GP kernels, and use them to define either full rank or sparse Gaussian processes.

```
# kernel & GP
kernel <- rbf(rbf_len, rbf_var) + bias(1)
f = gp(x, kernel)

# likelihood
distribution(y) = normal(f, obs_sd)

# prediction
f_plot <- project(f, x_plot)
```

github.com/greta-dev/greta.gp
what next?

- variational inference
- samplers for big data
- discrete samplers
- differential equations
why ‘greta’?

Grete Hermann  (1901-1984)

wrote the first algorithms for computer algebra

… without a computer

(I didn’t want people saying ‘greet’, so spelled the package *greta* instead)