Activity 3: Gaussian linear regression

In this activity, we will run a Bayesian version of a Gaussian linear regression. We will simulate the data, write the model in Stan, and fit the model using MCMC.

Our primary aim is to provide a template for linear regression analysis that we can build on for various types of regression analysis. The power of Bayesian inference lies in its ability to handle bespoke model structures (based on data characteristics and our hypotheses about the system we are modelling) that may include non-Gaussian error structures and non-linear deterministic functions. A Gaussian linear regression is the reference point.

Source files

1. Stan model:
   ./practicals/3_gaussian_regression/3_model.stan
2. R script to simulate data and run MCMC:
   ./practicals/3_gaussian_regression/3_script.R

The model

Our Gaussian linear regression model will take the following form:

$$y_i \sim Normal(\mu_i, \sigma)$$
$$\mu_i = \alpha + \sum_{k=1}^{K} \beta_k x_{k,i}$$

$$\alpha \sim Normal(0, 10)$$
$$\beta_k \sim Normal(0, 10)$$
$$\sigma \sim Uniform(0, 1e4)$$

Q: Can you draw a DAG for this model?
Hint: If you get stuck, the answer is in ./practicals/DAGs.pdf

Stan model

Let's take a look at how to write this model in Stan. See .../3_model.stan. This model is setup very similar to our model of the mean from activity 1. Let's discuss the differences in each model block, including introducing an important new parameter block (transformed parameters).

data{}
The data model block now includes an integer K that defines the number of predictor variables and a matrix $x_{i,k}$ of values for our predictors. Notice that this matrix has a row for each sample n and a column for each predictor variable k.

parameters{}
The parameters block now includes our regression coefficients: the intercept $alpha$ (α) and the effect size $beta$ (β) of each predictor variable.

transformed parameters{}
This model block was not included in our model of the mean. This is where we can define parameters that are derived deterministically from other parameters and data in the model. In this case, our linear regression determines the mean μ of the normal distribution used as our likelihood in the *model{}* block below.

Note that we are using a matrix multiplication to multiply the matrix $x_{i,k}$ of predictor variables by our vector of regression coefficients β_k. This notation provides flexibility to include more (or fewer) predictor variables in the model by simply modifying the data and without needing to rewrite the model code.

model{}
This block is very similar to our model of the mean. The difference is only that mu (μ) is now derived in the transformed parameters block rather than having a prior in the model block, and we have added priors for $alpha$ (α) and $beta$ (β).

We have introduced a new distribution as the prior for the standard deviation $sigma$ (σ) in our likelihood. This is the half-cauchy distribution ("half" because sigma is defined to be greater than 0 in the parameters block). The Cauchy distribution is a t-distribution with one degree of freedom. This is similar to a normal distribution with "fat" tails (i.e. a higher probability of outliers). This is commonly used as the prior for standard deviation parameters, although there are other viable alternatives (e.g. uniform, half-normal).

Simulate data
We provided code to simulate data for our regression analysis in .../3_script.R.

The way the data are simulated mirrors how the model is written in the stan code. The parameter values are randomly drawn from distributions (analogous to defining the priors), the expected value of the response variable is derived deterministically from the linear regression, there is normally distributed random noise added on. We even used the same matrix multiplication trick (x * beta) so that you can convince yourself that this is equivalent to the longer form regression notation (beta1 * x1 + beta2 * x2).

Notice that the regression coefficients (alpha and the betas) are simulated randomly so that they change each time you re-simulate the data. This is useful to see how well the model can recover different combinations of these "true" parameter values in its posterior parameter estimates.

Note that we can easily modify our sample size and the number of predictor variables in our data/model by changing the values in the simulation code. This is useful to explore how the model results might change with small sample sizes and different numbers of covariates/predictors.

Run MCMC and evaluate results
This section is identical to our previous work with the model of the mean.