Varying Intercepts Models

General Principles

To model the relationship between a dependent variable and an independent variable while allowing for different intercepts across groups or clusters, we can use a *Varying Intercepts* model. This approach is particularly useful when data are grouped (e.g., by subject, location, or time period) and we expect the baseline level of the outcome to vary across these groups.

Considerations

> **Note**
>
> - We have the same considerations as for Regression for a continuous variable.
> - The main idea of varying intercepts is to generate an intercept for each group, allowing each group to start at different levels. Thus, the intercept α_k is defined based on the k declared groups.
> - Each intercept has its own prior - i.e., a hyper-prior .
> - In the code below, the intercept `alpha` for each of the k declared groups shares two *hyper-priors*, `a_bar` and `sigma`, which are respectively modeled by a Normal and an Exponential distribution.

Example

Below is an example code snippet demonstrating Bayesian regression with varying intercepts using the Bayesian Inference (BI) package. The data consists of a dependent variable representing individuals' survival (*surv*) and an independent categorical variable (*tank*), which indicates the tank where the individual was born, with a total of 48 tanks. This example is based on McElreath (2018).

Python (Raw)

```python
from BI import bi
import numpy as np

# Setup device------------------------------------------------
m = bi(platform='cpu')

# Import Data & Data Manipulation ------------------------------------------------
# Import
from importlib.resources import files
data_path = files('BI.resources.data') / 'reedfrogs.csv'
m.data(data_path, sep=';')
# Manipulate
m.df["tank"] = np.arange(m.df.shape[0])

# Define model ------------------------------------------------
def model(tank, surv, density):
    sigma = m.dist.exponential( 1,  name = 'sigma')
    a_bar = m.dist.normal( 0., 1.5,  name = 'a_bar')
    alpha = m.dist.normal( a_bar, sigma, shape= tank.shape, name = 'alpha')
    p = alpha[tank]
    m.dist.binomial(total_count = density, logits = p, obs=surv)

# Run sampler ------------------------------------------------
m.fit(model)

# Diagnostic ------------------------------------------------
m.summary()
```

Python (Build in function)

```python
from BI import bi
import numpy as np

# Setup device------------------------------------------------
m = bi(platform='cpu')

# Import Data & Data Manipulation ------------------------------------------------
# Import
from importlib.resources import files
```

```python
data_path = files('BI.resources.data') / 'reedfrogs.csv'
m.data(data_path, sep=';')
# Manipulate
m.df["tank"] = np.arange(m.df.shape[0])

# Define model ------------------------------------------------
def model(tank, surv, density):
    alpha = m.effects.varying_intercept(N_groups=48,group_id=tank, group_name = 'tank')
    m.dist.binomial(total_count = density, logits = alpha, obs=surv)

# Run sampler ------------------------------------------------
m.fit(model)

# Diagnostic ------------------------------------------------
m.summary()
```

R

```r
library(BI)

# setup platform------------------------------------------------
m=importbi(platform='cpu')

# Import data ------------------------------------------------
m$data(paste(system.file(package = "BI"),"/data/reedfrogs.csv", sep = ''), sep=';')
m$df$tank = c(0:(nrow(m$df)-1)) # Manipulate
m$data_to_model(list('tank', 'surv', 'density')) # Manipulate
m$data_on_model$tank = m$data_on_model$tank$astype(jnp$int32) # Manipulate
m$data_on_model$surv = m$data_on_model$surv$astype(jnp$int32) # Manipulate


# Define model ------------------------------------------------
model <- function(tank, surv, density){
  # Parameter prior distributions
  sigma = bi.dist.exponential( 1,  name = 'sigma',shape=c(1))
  a_bar =  bi.dist.normal(0, 1.5, name='a_bar',shape=c(1))
  alpha = bi.dist.normal(a_bar, sigma, name='alpha', shape =c(48))
  p = alpha[tank]
  # Likelihood
  m$binomial(total_count = density, logits = p, obs=surv)
}
```

```
# Run MCMC ------------------------------------------------
m$fit(model) # Optimize model parameters through MCMC sampling

# Summary ------------------------------------------------
m$summary() # Get posterior distribution
```

Mathematical Details

We model the relationship between the independent variable X and the outcome variable Y while accounting for varying intercepts α for each group where $k(i)$ give us group belonging for observation i, using the following equation:

$$Y_i \sim \text{Normal}(\mu_i, \sigma)$$
$$\mu_i = \alpha_{[k(i)]} + \beta X_i$$

$$\beta \sim \text{Normal}(0, 1)$$

$$\sigma \sim \text{Exponential}(1)$$

$$\alpha_{[k]} \sim \text{Normal}(\bar{\alpha}, \varsigma)$$

$$\bar{\alpha} \sim \text{Normal}(0, 1)$$
$$\varsigma \sim \text{Exponential}(1)$$

Where:

- Y_i is the outcome variable for observation i.
- $\alpha_{[k(i)]}$ is the varying intercept corresponding to the group k of observation i.
- β is the regression coefficient.
- σ is a standard deviation parameter, which here has an Exponential prior that constrains it to be positive.
- $\bar{\alpha}$ is the overall mean intercept.
- ς is the variance of the intercepts across groups.

Notes

> **Note**
>
> - We can apply multiple variables similarly to Chapter 2.
> - We can apply interaction terms similarly to Chapter 3.
> - We can apply categorical variables similarly to Chapter 4.
> - We can apply varying intercepts with any distribution developed in previous chapters.

Reference(s)

McElreath, Richard. 2018. *Statistical Rethinking: A Bayesian Course with Examples in r and Stan*. Chapman; Hall/CRC.