Bayesian methods in economics and finance

Linear regression: Data modeling with Stan
Statistical modeling of data:

1. State your assumptions
   - Bayesian model consists of prior and likelihood

2. Fit model to your data
   - Compute posterior \( p(\theta|D) \)

3. Is the model any good?
   - Model selection
   - Posterior predictive checking

4. Refine model and repeat . . .

   All models are wrong but some are useful

George Box
Statistical modeling

Workflow of statistical modeling:

![Workflow Diagram]

Linear regression

Model for relationship between

- (several) independent variables \( \mathbf{x} = (x_1, \ldots, x_{D-1}) \)
- and dependent variable \( y \)

\[
y = w_0 + \sum_{i=1}^{D-1} w_i x_i + \epsilon
\]

- Structure: Linear relationship with parameters \( \mathbf{w} \)
- Noise: Additive observation noise \( \epsilon \sim \mathcal{N}(0, \sigma_\epsilon) \)
**Linear regression**

Model for relationship between
- (several) independent variables \( \mathbf{x} = (x_1, \ldots, x_{D-1}) \)
- and dependent variable \( y \)

\[
y = w_0 + \sum_{i=1}^{D-1} w_i x_i + \epsilon
\]

- **Structure**: Linear relationship with parameters \( \mathbf{w} \)
- **Noise**: Additive observation noise \( \epsilon \sim \mathcal{N}(0, \sigma_\epsilon) \)

Convenient matrix notation

\[
y = \mathbf{w}^T \mathbf{x} + \epsilon
\]

where \( \mathbf{w}, \mathbf{x} \in \mathbb{R}^D \) and \( x_0 \equiv 1 \)
Linear regression

Linear regression can be considered as a statistical model

\[ p(y|x) = \mathcal{N}(y|\mathbf{w}^T \mathbf{x}, \sigma_\epsilon) \]

is easily generalized using basis functions \( \Phi_i(x) \)

\[ y = \mathbf{w}^T \Phi(x) + \epsilon \]

Common basis functions include

- Polynomials of degree \( k \), i.e.
  \[ \Phi_0(x) \equiv 1, \Phi_1(x) = x, \ldots, \Phi_k(x) = x^k \]

- Radial basis functions, i.e. \( \Phi_i(x) = e^{-\frac{(x-\mu_i)^2}{\sigma_i^2}} \)
Probabilistic programming language **Stan**:

- Specify statistical model as computer program
- Inference is done automatically:
  - Model compiled to C++
  - Automatic differentiation for efficient sampling
- Posterior represented via samples
Linear regression

Stan code for linear regression:

```stan
data {
  int<lower=0> N; // Number of training data
  int<lower=0> D; // Number of regressors
  vector[D] X[N]; // Regressor inputs
  real t[N]; // Target outputs
  real<lower=0> alpha; // Hyperparameters
  real<lower=0> beta;
  real<lower=0> tau0;
} parameters {
  vector[D] w; // Regression weights
  real<lower=0> tau_eps; // Noise precision
} model {
  // Priors
  tau_eps ~ gamma(alpha, beta);
  w ~ normal(0, sqrt(1/tau0));
  // Likelihood
  for (n in 1:N) {
    real y;
    y <- transpose(w) * X[n]; // dot product w^T x_n
    t[n] ~ normal(y, sqrt(1/tau_eps));
  }
}
```

- Program specifies \( \ln p(\text{data, parameters}) \)
- Samples/optimizes/approximates \( p(\text{parameters} \mid \text{data}) \)
Illustration

Run linear regression model on some demo data:

data <- read.table("demo.csv", sep="", header=TRUE)

library(rstan)
rstan_options(auto_write = TRUE)
options(mc.cores = parallel::detectCores())

N <- dim(data)[1]
fit <- stan(model_code=lm.code,
    data=list(N=N, D=2, X=cbind(rep(1,N), data$x), t=data$y,
        alpha=1, beta=1, tau0=1))
Illustration

- *shinystan* is graphical tool to explore posterior samples
- Bayesian linear regression using uninformative priors agrees with Ordinary least squares solution
- Analyse predictions of model
  - Plot posterior uncertainty
  - Plot model residuals
- Model residuals suggest non-linear (quadratic) influence of regressor
- Fit improved model and repeat . . .