Bayes Test Practice 1

Sean van der Merwe

2022/04/04

Instructions

```
DistNames <-
c('Beta','Binomial','ChiSquare','Exponential','F','Gamma','Normal','LogNormal','t')
```

The following distributions are easy to simulate from in Excel (using the inverse CDF functions built-in): Beta, Binomial, ChiSquare, Exponential, F, Gamma, Normal, LogNormal, t.

1. For each of these distributions, pick interesting parameters and simulate a single sample of size 50. Put these numbers on a spreadsheet in a table of size 50 by 9, with headings in the first row, and save it.
2. Import your spreadsheet into R and give a summary table. Discuss whether the average of each column seems reasonable.
3. Use Stan to fit each distribution to its sample. You will need to specify some priors of your choosing here and there (e.g. for the t df). Thus you will be doing 9 fits in total. For each fit discuss the trace plot and convergence.
4. Obtain parameter estimates from your simulations and compare them to the parameters your chose, and discuss any meaningful differences.

100 marks for following the instructions properly, -50 for having too similar parameters and numbers to another person, -50 for handing in late.

Example Solution

Part 1

Beta	Binomial	ChiSquare	Exponential	F	Gamma	Normal	LogNormal	t
=BETA.INV(RAND(), 3, 8)	=BINOM.INV(20, 0.4, RAND())	=CHISQ.INV(RAND(), 7)	=LN(RAND())/(-1.5)	=F.INV(RAND(), 4, 35)	=GAMMA.INV(RAND(), 3, 2)	=NORM.INV(RAND(), 3.4, 2.5)	=LOGNORM.INV(RAND(), 1.2, 0.4)	=T.INV(RAND(), 4)

Part 2

Here we load the data and give a summary.

```
"Test1Practice2022Bayes.xlsx" |> openxlsx::read.xlsx("Numbers") -> d
"Test1Practice2022Bayes.xlsx" |> openxlsx::read.xlsx("Parameters") -> pars

pars |> kable()
```

Num	Desc	Par1	Par2	Par3	Ex
1	Beta	3.0	8.0	NA	0.2727273

Num	Desc	Par1	Par2	Par3	Ex
2	Binomial	20.0	0.4	NA	8.0000000
3	ChiSquare	7.0	NA	NA	7.0000000
4	Exponential	1.5	NA	NA	0.6666667
5	F	4.0	35.0	NA	1.0606061
6	Gamma	3.0	0.5	NA	6.0000000
7	Normal	3.4	2.5	NA	3.4000000
8	LogNormal	1.2	0.4	NA	3.5966397
9	t	4.0	0.0	1	0.0000000

```
d |> summary() |> kable()
```

Beta	Binomial	ChiSquare	Exponential	F	Gamma	Normal	LogNormal	t
Min. :0.0540	Min. : 4.00	Min. : 1.657	Min. :0.002765	Min. :0.0999	Min. : 0.6784	Min. :-3.411	Min. :1.407	Min. :-3.21272
1st Qu.:0.1478	1st Qu.: 7.00	1st Qu.: 4.126	1st Qu.:0.156398	1st Qu.:0.6060	1st Qu.: 3.0841	1st Qu.: 2.064	1st Qu.:2.555	1st Qu.:-0.64272
Median :0.2625	Median : 8.00	Median : 6.794	Median :0.403495	Median :0.9481	Median : 4.2988	Median : 3.308	Median :3.476	Median : 0.05910
Mean :0.2618	Mean : 8.02	Mean : 6.971	Mean :0.624252	Mean :1.0871	Mean : 5.0766	Mean : 3.725	Mean :3.629	Mean :-0.04075
3rd Qu.:0.3578	3rd Qu.: 9.00	3rd Qu.: 8.626	3rd Qu.:0.941209	3rd Qu.:1.4348	3rd Qu.: 6.4401	3rd Qu.: 5.646	3rd Qu.:4.477	3rd Qu.: 0.53238
Max. :0.6009	Max. :13.00	Max. :15.648	Max. :2.849005	Max. :3.0403	Max. :14.8405	Max. :10.305	Max. :7.716	Max. : 1.95351

The averages seem to be within a reasonable distance of the expected values, thus we conclude that the reparameterisations were done appropriately.

Part 3

First we load Stan.

```
library(parallel)
library(rstan)
mycores <- max(1,floor(detectCores(logical = FALSE)*0.75))
options(mc.cores = mycores)
rstan_options(auto_write = TRUE)
```

Then we define all the models.

```
// This Stan block defines a Beta model, by Sean van der Merwe, UFS
data {
  int<lower=1> n;                       // number of observations
  real<lower=0, upper=1> y[n];       // observations
}
// The parameters of the model
parameters {
  real<lower=0> a;
  real<lower=0> b;
}
model {
  y ~ beta(a, b);
}

// This Stan block defines a Binomial model, by Sean van der Merwe, UFS
data {
  int<lower=1> n;                       // number of observations
  int<lower=1> N;                 // Binomial upper limit
  int<lower=0, upper=N> y[n];       // observations
}
parameters {
  real<lower=0, upper=1> p;
}
model {
  y ~ binomial(N, p);
}

// This Stan block defines a ChiSquare model, by Sean van der Merwe, UFS
data {
  int<lower=1> n;                       // number of observations
  real<lower=0> y[n];       // observations
}
// The parameters of the model
parameters {
  real<lower=0> m;
}
model {
  y ~ chi_square(m);
}

// This Stan block defines a Exponential model, by Sean van der Merwe, UFS
data {
  int<lower=1> n;                       // number of observations
  real<lower=0> y[n];       // observations
}
// The parameters of the model
parameters {
  real<lower=0> l;
}
model {
  y ~ exponential(l);
}

// This Stan block defines an F model, by Sean van der Merwe, UFS
functions{
    real f_lpdf(real x, real d1, real d2){
//    return(beta_lpdf(d1*x/d2/(d1*x/d2 + 1) | d1/2, d2/2));
      return (  0.5 * (-d1 * log(d1 * x + d2) -
```

```
                        d2 * log(d1 * x + d2) + d1 * log(x) + d1 * log(d1) +
                        d2 * log(d2) - 2 * log(x)) - lbeta(d1/2, d2/2)          );
    }
}
data {
  int<lower=1> n;                          // number of observations
  real<lower=0> y[n];       // observations
}
// The parameters of the model
parameters {
  real<lower=0> d1;
  real<lower=0> d2;
}
model {
  for (i in 1:n) {
    y[i] ~ f(d1, d2);
  }       // Stan doesn't have an F distribution
  d1 ~ exponential(0.1);
  d2 ~ exponential(0.1);
}

// This Stan block defines a Gamma model, by Sean van der Merwe, UFS
data {
  int<lower=1> n;                          // number of observations
  real<lower=0> y[n];       // observations
}
// The parameters of the model
parameters {
  real<lower=0> a;
  real<lower=0> l;
}
model {
  y ~ gamma(a, l);
}

// This Stan block defines a Normal model, by Sean van der Merwe, UFS
data {
  int<lower=1> n;                          // number of observations
  real y[n];       // observations
}
// The parameters of the model
parameters {
  real m;
  real<lower=0> s;
}
model {
  y ~ normal(m, s);
}

// This Stan block defines a LogNormal model, by Sean van der Merwe, UFS
data {
  int<lower=1> n;                          // number of observations
  real<lower=0> y[n];       // observations
}
// The parameters of the model
parameters {
  real m;
  real<lower=0> s;
```

```
}
model {
  y ~ lognormal(m, s);
}

// This Stan block defines a t model, by Sean van der Merwe, UFS
data {
  int<lower=1> n;                         // number of observations
  real y[n];       // observations
}
// The parameters of the model
parameters {
  real m;
  real<lower=0> s;
  real<lower=0.5> v;
}
model {
  y ~ student_t(v, m, s);
  target += log(v) - 3*log(v+0.75) - 2*log(s);   // joint objective prior
}
```

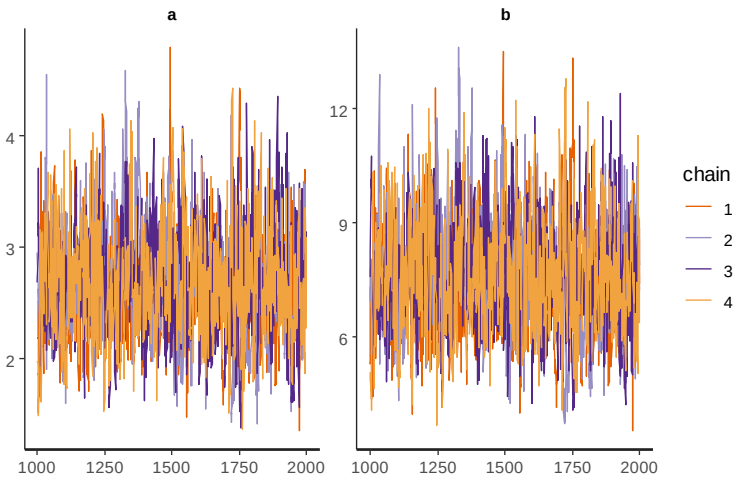
Then we save all the models. We mark all the models and the saving as 'Do not evaluate' meaning that we must run them manually every time we move to a new computer.

```
models <- list(Beta, Binomial, ChiSquare, Exponential, Fdist, Gamma, Normal,
LogNormal, tdist)
names(models) <- DistNames
saveRDS(models, file = 'models.Rds')
```

And we load the models that we previously saved. This is generally bad practice, but saves a lot of time when knitting.

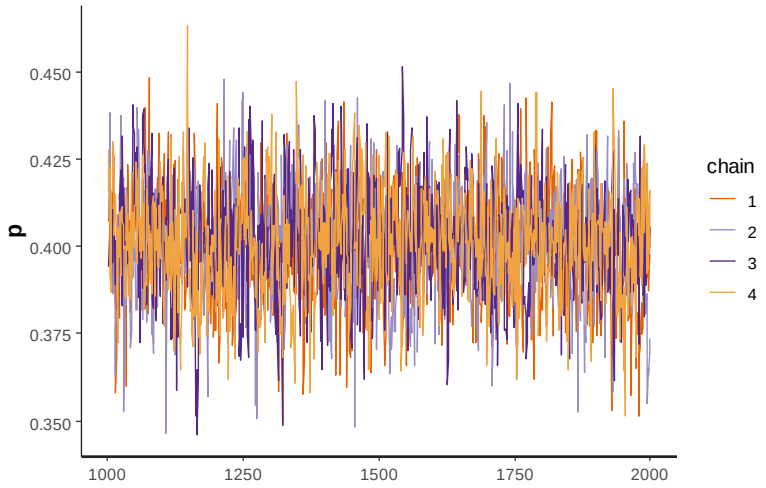
Finally, we run the models and look at the output.

```
n <- nrow(d)
for (m in 1:(nrow(pars))) {
  stan_data <- list(n=nrow(d), y=d[[m]], N=20)
  ModelFit <- sampling(models[[m]], stan_data, chains = mycores)
  ModelFit |> traceplot() |> print()
  kable(round(summary(ModelFit)$summary, 3)) |> print()
  kable(pars[m,]) |> print()
}
```



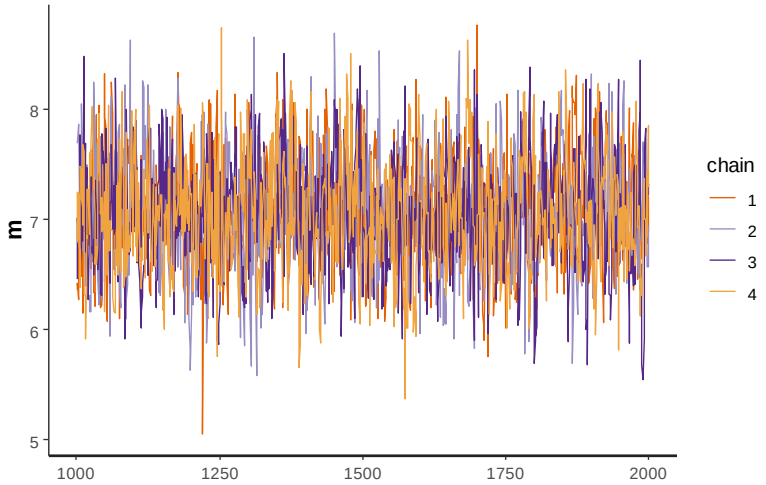
	mean	se_mean	sd	2.5%	25%	50%	75%	97.5%	n_eff	Rhat
a	2.680	0.019	0.498	1.782	2.335	2.643	3.005	3.734	708.474	1.010
b	7.612	0.056	1.512	4.873	6.566	7.524	8.559	10.829	720.316	1.009
lp__	34.363	0.033	1.003	31.490	33.967	34.681	35.079	35.354	901.342	1.003

Num	Desc	Par1	Par2	Par3	Ex
1	Beta	3	8	NA	0.2727273



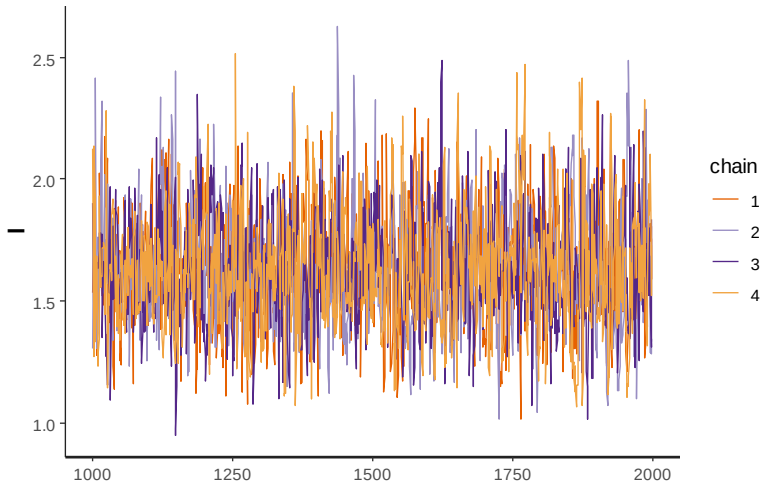
	mean	se_mean	sd	2.5%	25%	50%	75%	97.5%	n_eff	Rhat
p	0.401	0.000	0.016	0.37	0.39	0.401	0.412	0.432	1461.630	1.001
lp__	-675.370	0.019	0.764	-677.55	-675.53	-675.084	-674.895	-674.842	1673.621	1.003

	Num	Desc	Par1	Par2	Par3	Ex
2	2	Binomial	20	0.4	NA	8



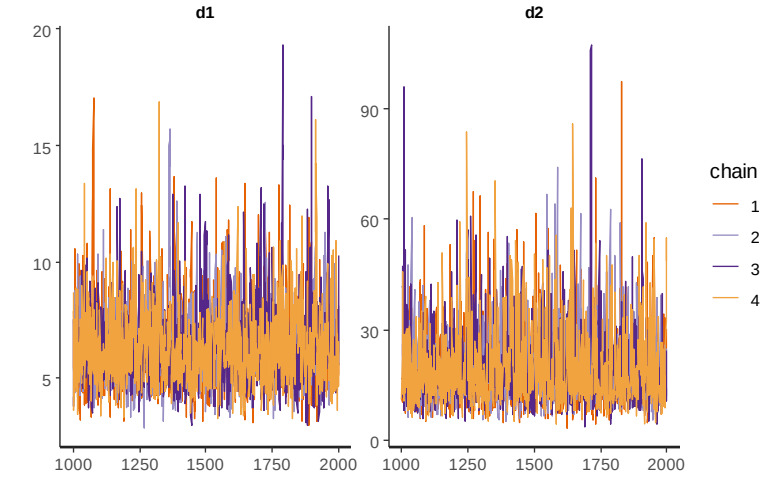
	mean	se_mean	sd	2.5%	25%	50%	75%	97.5%	n_eff	Rhat
m	7.067	0.012	0.497	6.118	6.732	7.058	7.398	8.061	1656.601	1.000
lp__	45.686	0.017	0.721	43.722	45.507	45.971	46.144	46.195	1769.178	1.001

	Num	Desc	Par1	Par2	Par3	Ex
3	3	ChiSquare	7	NA	NA	7



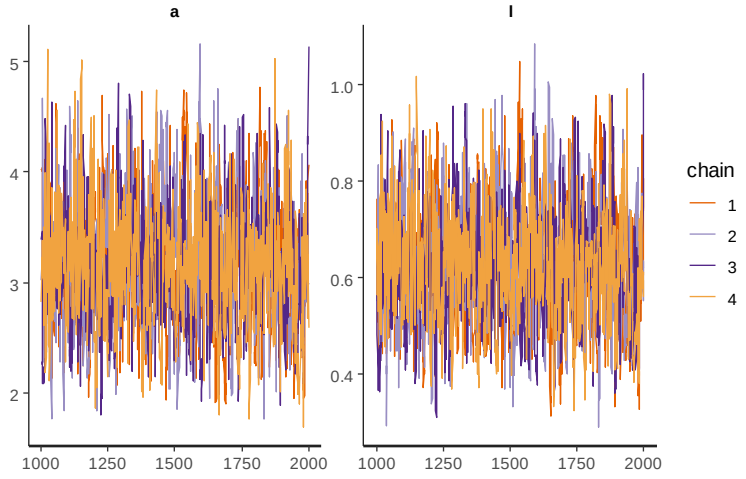
	mean	se_mean	sd	2.5%	25%	50%	75%	97.5%	n_eff	Rhat
l	1.636	0.006	0.237	1.198	1.471	1.626	1.791	2.130	1586.458	1.000
lp__	-26.494	0.020	0.748	-28.631	-26.662	-26.206	-26.014	-25.959	1432.971	1.005

	Num	Desc	Par1	Par2	Par3	Ex
4	4	Exponential	1.5	NA	NA	0.6666667



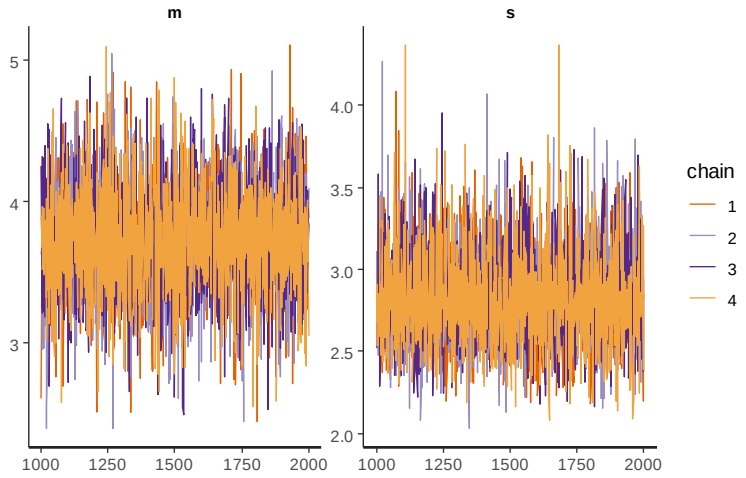
	mean	se_mean	sd	2.5%	25%	50%	75%	97.5%	n_eff	Rhat
d1	6.438	0.042	1.769	3.862	5.207	6.158	7.343	10.542	1796.272	1.000
d2	19.781	0.243	10.560	7.065	12.487	17.386	24.328	46.694	1886.883	1.000
lp__	-44.541	0.025	1.011	-47.280	-44.942	-44.224	-43.826	-43.557	1653.221	1.005

	Num	Desc	Par1	Par2	Par3	Ex
5	5	F	4	35	NA	1.060606



	mean	se_mean	sd	2.5%	25%	50%	75%	97.5%	n_eff	Rhat
a	3.151	0.020	0.572	2.121	2.740	3.134	3.526	4.341	800.799	1.001
l	0.623	0.004	0.123	0.405	0.535	0.620	0.703	0.881	812.637	1.001
lp__	-118.992	0.028	0.976	-121.610	-119.400	-118.691	-118.275	-118.016	1211.071	1.003

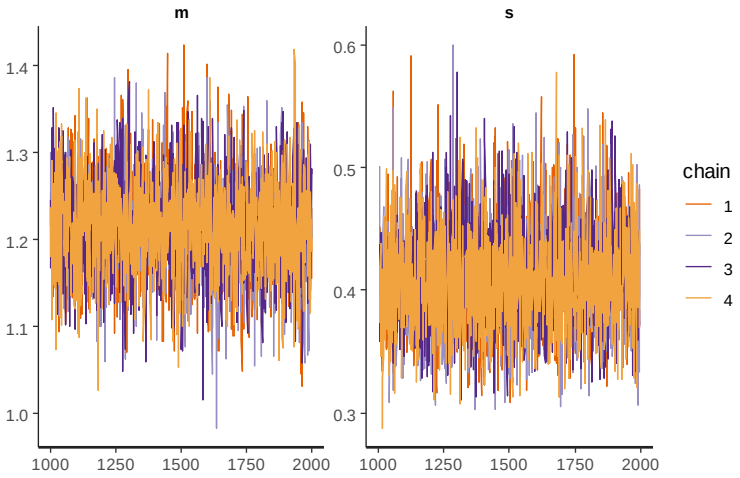
	Num	Desc	Par1	Par2	Par3	Ex
6	6	Gamma	3	0.5	NA	6



	mean	se_mean	sd	2.5%	25%	50%	75%	97.5%	n_eff	Rhat
m	3.717	0.007	0.402	2.922	3.438	3.713	3.984	4.508	3111.971	1
s	2.836	0.005	0.296	2.325	2.629	2.809	3.015	3.489	3242.427	1
lp__	-75.222	0.025	1.020	-77.920	-75.636	-74.890	-74.499	-74.237	1624.007	1

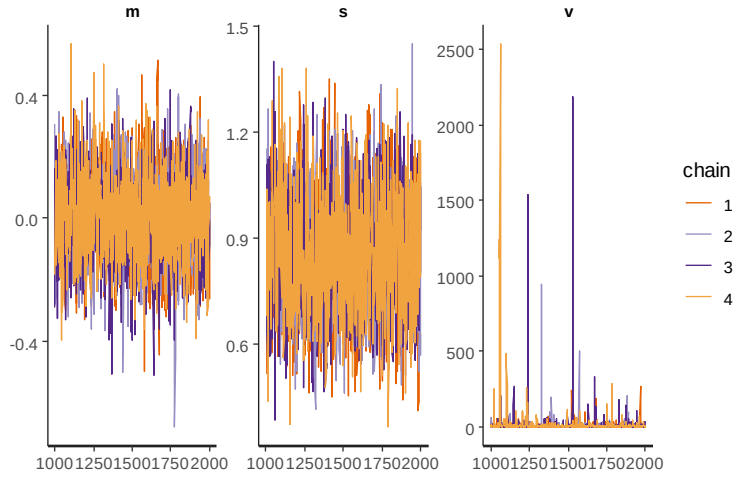
	Num	Desc	Par1	Par2	Par3	Ex

	Num	Desc	Par1	Par2	Par3	Ex
7	7	Normal	3.4	2.5	NA	3.4



	mean	se_mean	sd	2.5%	25%	50%	75%	97.5%	n_eff	Rhat
m	1.214	0.001	0.058	1.101	1.175	1.215	1.251	1.326	3145.011	1.000
s	0.409	0.001	0.043	0.335	0.379	0.406	0.435	0.502	3167.708	1.001
lp__	19.626	0.023	0.995	16.979	19.245	19.909	20.343	20.610	1824.392	1.000

	Num	Desc	Par1	Par2	Par3	Ex
8	8	LogNormal	1.2	0.4	NA	3.59664



	mean	se_mean	sd	2.5%	25%	50%	75%	97.5%	n_eff	Rhat
m	0.004	0.003	0.142	-0.272	-0.092	0.002	0.100	0.275	2408.876	1.001
s	0.848	0.004	0.161	0.545	0.734	0.842	0.960	1.165	1794.026	1.000
v	13.318	1.947	72.702	1.565	2.859	4.399	7.945	62.641	1394.136	1.003
lp__	-48.677	0.036	1.327	-52.112	-49.299	-48.352	-47.706	-47.143	1367.831	1.003

	Num	Desc	Par1	Par2	Par3	Ex
9	9	t	4	0	1	0

Part 4

Comparing the posterior mean estimates from the summaries to the selected parameters we see good agreement in all cases.