HW8

Here I estimate two models that are similar to Model III and Model IV in the paper. I use the same predictors but add a prior over the β coefficients. In order to use some partial pooling, and since most of the coefficients are on the same scale, I use $\beta \sim \mathcal{N}(0, \tau_\beta^2)$. Since value is not on a similar [-.5, .5] range, for any parameter involving value, I use $\beta \sim \mathcal{N}(0, \tau_{\beta v}^2)$. τ_β and $\tau_{\beta v}$ both have a noninformative uniform prior. I also estimate model 3 as in the paper for comparison.

The first column are the Model III estimates from the model in the paper and the second and third columns are the Model III and IV estimates using priors on the regression weights. Each is presented with the median sample and the half-interquartile range giving implicit 50% intervals. As in the paper, the weights are multiplied by 100 so that they can be interpreted as percentage points.

	Model III (paper)		Model III		Model IV	
	estimates	hiq	estimates	hiq	estimates	hiq
Constant	60.80	2.43	61.95	2.16	61.58	2.19
Incentive	2.64	0.98	2.30	1.04	3.42	0.94
Burden	-8.91	4.74	-0.78	1.97	-1.43	2.36
Mode	15.79	4.52	2.13	2.55	3.20	3.05
Mode x Burden	-8.03	9.68	-0.64	2.13	-0.63	2.49
Incentive x Value	0.30	0.08	0.25	0.09	0.27	0.07
Incentive x Timing	3.12	1.80	2.04	1.57	1.13	1.41
Incentive x Form	-3.07	1.49	-1.68	1.34	-0.89	1.28
Incentive x Mode	-2.45	1.58	-0.32	1.22	2.67	1.76
Incentive x Burden	6.26	1.68	2.98	1.67	0.17	1.67
Incentive x Value x Burden	-0.06	0.06	-0.02	0.05	0.39	0.14
Incentive x Value x Timing	0.36	0.15	0.24	0.17	0.51	0.15
Incentive x Form x Burden					-3.32	2.12
Incentive x Timing x Burden					-0.44	2.63
Incentive x Timing x Form					-4.07	2.16
Incentive x Timing x Mode					-3.16	1.26
Incentive x Value x Form					-0.21	0.14
Incentive x Value x Mode					-0.52	0.15
τ_β^2			3.30	1.45	4.03	1.22
$\tau_{\beta v}^2$			0.31	0.20	0.52	0.19
σ^2	4.22	0.29	4.27	0.28	3.58	0.26
τ^2	18.27	1.65	18.53	1.41	18.65	1.45

We can see that many of the estimates have the same sign as the original model III and many even have similar magnitudes, suggesting that the partial pooling was not too restrictive. (Although the Incentive X Form beta was flipped, maybe an issue with the data?) Additionally, the partial pooling seems to have successfully reduced the noise in the estimates to a reasonable level. However, some of the first order terms have much lower magnitudes suggesting that it may have been better to use a separate prior for those beta weights. Here we replicate table 3 to find out what the conclusions we can make about effects.

Scenario	Model III	Model IIIb	Model IV
Low burden/prepay	-.76 + .51 x value	1.15 + .38 x value	2.83 + .485 x value
Low burden/postpay	-2.36 + .15 x value	0.89 + .14 x value	1.9375 + .025 x value
High burden/prepay	7.02 + .45 x value	4.13 + .36 x value	1.1225 + .875 x value
High burden/postpay	3.9 + .09 x value	2.09 + .12 x value	0.6675 + .365 x value

We can see that while magnitudes are slightly different, effects are largely all in the same direction. In general, effect sizes are slightly smaller in the new models although increase value takes on a much larger role in

Model IV than Model III from the paper.

Stan code for all of the models:

```
data {
  int<lower=0> N_I;
  int<lower=0> N_J;
  int J[N_I]; //sid
  int N[N_I]; //basen
  vector[N_I] y; //r
  vector[N_I] V;
  vector[N_I] I;
  vector[N_I] m;
  vector[N_I] t;
  vector[N_I] f;
  vector[N_I] v;
  vector[N_I] b;
}
parameters {
  real<lower=0> sigma_sq;
  real<lower=0> tau_sq;
  real beta_m;
  real beta_b;
  real beta_I;
  real beta_mb;
  real beta_Iv;
  real beta_It;
  real beta_If;
  real beta_Im;
  real beta_Ib;
  real beta_Ivt;
  real beta_Ivb;
  real constant;
  vector[N_J] alpha;
  vector[N_I] pi;
}
transformed parameters {
  vector[N_I] XB;
  for (i in 1:N_I)
    XB[i] <- beta_I * I[i] + beta_b * b[i] + beta_m * m[i] + beta_mb * m[i] * b[i] + beta_Iv * I[i] * v
}
model {
  alpha ~ normal(0, tau_sq);
  for(i in 1:N_I)
    y[i] ~ normal(XB[i] + alpha[J[i]], sigma_sq + V[i]);
}

data {
  int<lower=0> N_I;
  int<lower=0> N_J;
  int J[N_I]; //sid
  int N[N_I]; //basen
  vector[N_I] y; //r
  vector[N_I] V;
```

```
  vector[N_I] I;
  vector[N_I] m;
  vector[N_I] t;
  vector[N_I] f;
  vector[N_I] v;
  vector[N_I] b;
}
parameters {
  real<lower=0> sigma_sq;
  real<lower=0> tau_sq;
  real<lower=0> tau_b_sq;
  real<lower=0> tau_bv_sq;
  real beta_m;
  real beta_b;
  real beta_I;
  real beta_mb;
  real beta_Iv;
  real beta_It;
  real beta_If;
  real beta_Im;
  real beta_Ib;
  real beta_Ivt;
  real beta_Ivb;
  real constant;
  vector[N_J] alpha;
  vector[N_I] pi;
}
transformed parameters {
  vector[N_I] XB;
  for (i in 1:N_I)
    XB[i] <- beta_I * I[i] + beta_b * b[i] + beta_m * m[i] + beta_mb * m[i] * b[i] + beta_Iv * I[i] * v
}
model {
  beta_m ~ normal(0, tau_b_sq);
  beta_b ~ normal(0, tau_b_sq);
  beta_I ~ normal(0, tau_b_sq);
  beta_mb ~ normal(0, tau_b_sq);
  beta_Iv ~ normal(0, tau_bv_sq);
  beta_It ~ normal(0, tau_b_sq);
  beta_If ~ normal(0, tau_b_sq);
  beta_Im ~ normal(0, tau_b_sq);
  beta_Ib ~ normal(0, tau_b_sq);
  beta_Ivt ~ normal(0, tau_bv_sq);
  beta_Ivb ~ normal(0, tau_bv_sq);
  alpha ~ normal(0, tau_sq);
  for(i in 1:N_I)
    y[i] ~ normal(XB[i] + alpha[J[i]], sigma_sq + V[i]);
}

data {
  int<lower=0> N_I;
  int<lower=0> N_J;
  int J[N_I]; //sid
  int N[N_I]; //basen
```

```
  vector[N_I] y; //r
  vector[N_I] V;
  vector[N_I] I;
  vector[N_I] m;
  vector[N_I] t;
  vector[N_I] f;
  vector[N_I] v;
  vector[N_I] b;
}
parameters {
  real<lower=0> sigma_sq;
  real<lower=0> tau_sq;
  real<lower=0> tau_b_sq;
  real<lower=0> tau_bv_sq;
  real beta_m;
  real beta_b;
  real beta_I;
  real beta_mb;
  real beta_Iv;
  real beta_It;
  real beta_If;
  real beta_Im;
  real beta_Ib;
  real beta_Ivt;
  real beta_Ivb;
  real beta_Itb;
  real beta_Ivf;
  real beta_Ivm;
  real beta_Itf;
  real beta_Itm;
  real beta_Ifb;
  real constant;
  vector[N_J] alpha;
  vector[N_I] pi;
}
transformed parameters {
  vector[N_I] XB;
  for (i in 1:N_I)
    XB[i] <- beta_I * I[i] + beta_b * b[i] + beta_m * m[i] + beta_mb * m[i] * b[i] + beta_Iv * I[i] * v
}
model {
  beta_m ~ normal(0, tau_b_sq);
  beta_b ~ normal(0, tau_b_sq);
  beta_I ~ normal(0, tau_b_sq);
  beta_mb ~ normal(0, tau_b_sq);
  beta_Iv ~ normal(0, tau_bv_sq);
  beta_It ~ normal(0, tau_b_sq);
  beta_If ~ normal(0, tau_b_sq);
  beta_Im ~ normal(0, tau_b_sq);
  beta_Ib ~ normal(0, tau_b_sq);
  beta_Ivt ~ normal(0, tau_bv_sq);
  beta_Ivb ~ normal(0, tau_bv_sq);
  beta_Itb ~ normal(0, tau_b_sq);
  beta_Ivf ~ normal(0, tau_bv_sq);
```

```
  beta_Ivm ~ normal(0, tau_bv_sq);
  beta_Itf ~ normal(0, tau_b_sq);
  beta_Itm ~ normal(0, tau_b_sq);
  beta_Ifb ~ normal(0, tau_b_sq);
  alpha ~ normal(0, tau_sq);
  for(i in 1:N_I)
    y[i] ~ normal(XB[i] + alpha[J[i]], sigma_sq + V[i]);
}
```

Code for estimating models:

```r
setwd("~/Documents/BDA/Homework 8")
id <- read.table("incentives_data_clean.txt", skip = 13, header = T)

J <- as.integer(as.factor(id$sid))
N <- id$basen
n <- as.integer(id$r * N)
y <- id$r
V <- y * (1 - y) / N
N_J <- nlevels(as.factor(J))
N_I <- length(N)

I <- id$I
m <- id$m
t <- id$t
f <- id$f
v <- id$v
b <- id$b

dataList <- list(I=I, J=J, N=N, y=y, V=V, N_J=N_J, N_I=N_I, m=m, t=t, f=f, v=v, b=b)
fit_3 <- stan(file="incentives_model3.stan", data=dataList, iter=1000)
fit_4 <- stan(file="incentives_model4.stan", data=dataList, iter=1000)
fit_3b <- stan(file="incentives_model3b.stan", data=dataList, iter=1000)

monitor_4 <- as.data.frame(monitor(fit))
monitor_4 <- monitor_4[1:22,]
estimates_4 <- round(select(monitor_4, `50%`) * 100, digits = 2)
hiq_4 <- round((select(monitor_4, `75%`) - select(monitor_4, `25%`))*50, digits = 2)
reg_table_4 <- as.data.frame(cbind(estimates_4, hiq_4))
colnames(reg_table_4) <- c("estimates", "hiq")
reg_table_4$param <- rownames(reg_table_4)

monitor_3b <- as.data.frame(monitor(fit_3b))
monitor_3b <- monitor_3b[1:16,]
estimates_3b <- round(select(monitor_3b, `50%`) * 100, digits = 2)
hiq_3b <- round((select(monitor_3b, `75%`) - select(monitor_3b, `25%`))*50, digits = 2)
reg_table_3b <- as.data.frame(cbind(estimates_3b, hiq_3b))
colnames(reg_table_3b) <- c("estimates", "hiq")
reg_table_3b$param <- rownames(reg_table_3b)

monitor_3 <- as.data.frame(monitor(fit_3))
monitor_3 <- monitor_3[1:14,]
estimates_3 <- round(select(monitor_3, `50%`) * 100, digits = 2)
hiq_3 <- round((select(monitor_3, `75%`) - select(monitor_3, `25%`))*50, digits = 2)
```

```r
reg_table_3 <- as.data.frame(cbind(estimates_3, hiq_3))
colnames(reg_table_3) <- c("estimates", "hiq")
reg_table_3$param <- rownames(reg_table_3)

reg_table <- merge(reg_table_3, reg_table_3b, by.x = "param", by.y = "param", all = TRUE)
reg_table <- merge(reg_table, reg_table_4, by.x = "param", by.y = "param", all = TRUE)

#Low burden/prepay
x_mode <- -.5
x_burden <- -.5
x_form <- .5
x_timing <- .5

#Low burden/postpay
x_mode <- -.5
x_burden <- -.5
x_form <- .5
x_timing <- -.5

#High burden/prepay
x_mode <- -.5
x_burden <- .5
x_form <- .5
x_timing <- .5

#High burden/postpay
x_mode <- -.5
x_burden <- .5
x_form <- .5
x_timing <- -.5

estimates_3["beta_I",] + estimates_3["beta_Im",] * x_mode + estimates_3["beta_If",] * x_form + estimates
estimates_3["beta_Iv",] + estimates_3["beta_Ivb",] * x_burden + estimates_3["beta_Ivt",] * x_timing

estimates_3b["beta_I",] + estimates_3b["beta_Im",] * x_mode + estimates_3b["beta_If",] * x_form + estima
estimates_3b["beta_Iv",] + estimates_3b["beta_Ivb",] * x_burden + estimates_3b["beta_Ivt",] * x_timing

estimates_4["beta_I",] + estimates_4["beta_Im",] * x_mode + estimates_4["beta_If",] * x_form + estimates
estimates_4["beta_Iv",] + estimates_4["beta_Ivb",] * x_burden + estimates_4["beta_Ivt",] * x_timing + e
```