Bayesian Chap 5 and 6 Assignment

Meng Yuan

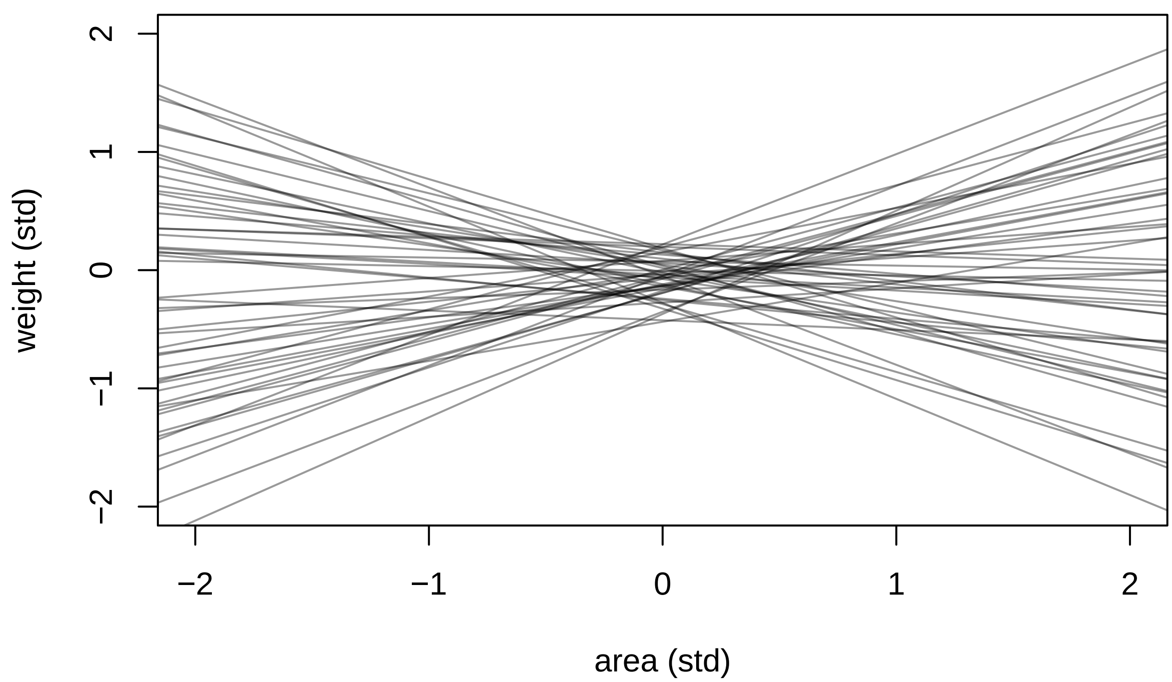
1. Modelling area -> weight

1.1 Using priors of normal distributions

```
m1.0<- quap(
    alist(
        W ~ dnorm(mu,sigma),
        mu ~ a + b_A * A,
        a ~ dnorm(0,0.2),
        b_A ~ dnorm(0,0.5),
        sigma ~ dexp(1)
    ),
    data=d
)
precis(m1.0)
```

```
##                mean         sd       5.5%     94.5%
## a     -1.475345e-08 0.08360868 -0.1336228 0.1336228
## b_A    1.883357e-02 0.09089583 -0.1264355 0.1641027
## sigma  9.912662e-01 0.06466650  0.8879166 1.0946157
```

```
# prior predictive simulation
set.seed(10)
prior <- extract.prior(m1.0)
mu <- link(m1.0, post=prior, data = list(A=c(-3,3)))
plot( NULL , xlim=c(-2,2) , ylim=c(-2,2), xlab="area (std)", ylab="weight (std)")
for (i in 1:50) lines(c(-3,3), mu[i,], col=col.alpha("black",0.4) )
```

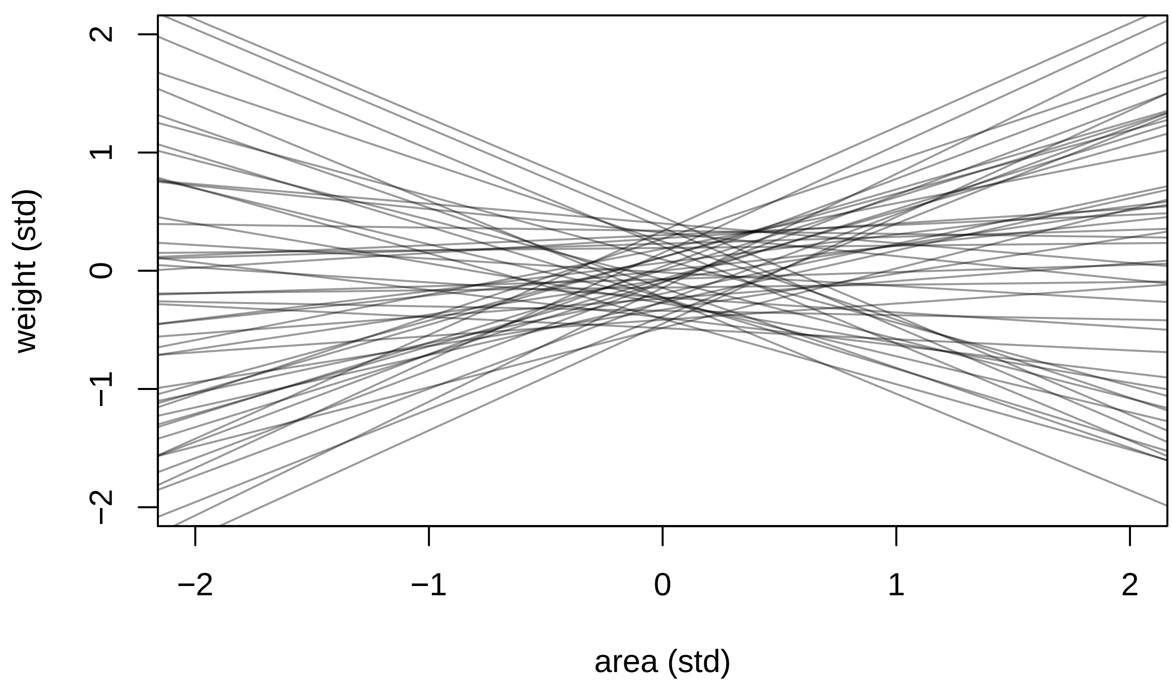


1.2 Using priors of uniform distributions

```
m1.1<- quap(
    alist(
        W ~ dnorm(mu,sigma),
        mu ~ a + b_A * A,
        a ~ dunif(-0.5,0.5),
        b_A ~ dunif(-1,1),
        sigma ~ dexp(1)
    ),
    data=d
)
precis(m1.1)
```

```
##               mean         sd       5.5%     94.5%
## a     4.511008e-08 0.09203671 -0.1470924 0.1470925
## b_A   1.947733e-02 0.09243600 -0.1282533 0.1672079
## sigma 9.912657e-01 0.06466641  0.8879163 1.0946151
```

```
# prior predictive simulation
set.seed(10)
prior <- extract.prior(m1.1)
mu <- link(m1.1, post=prior, data = list(A=c(-3,3)))
plot( NULL , xlim=c(-2,2) , ylim=c(-2,2), xlab="area (std)", ylab="weight (std)")
for (i in 1:50) lines(c(-3,3), mu[i,], col=col.alpha("black",0.4) )
```



The slope b_A is close to zero, based on the model, there's no causal influence of area on weight.

Following priors are used:

a ~ Uniform(-0.5,0.5)

b ~ Uniform(-1,1)

The standardized data has a mean of 0 and a standard deviation of 1, so the intercept a should be close to 0.

The standard deviation of area is 0.93, when the slope b is 1, a change of 0.93 in area is associated with a full standard deviation change in weight. Extremely strong is unlikely so b is mostly between -1 and 1. The prior predictive simulation shows the model's prior predictions stays within the possible range.

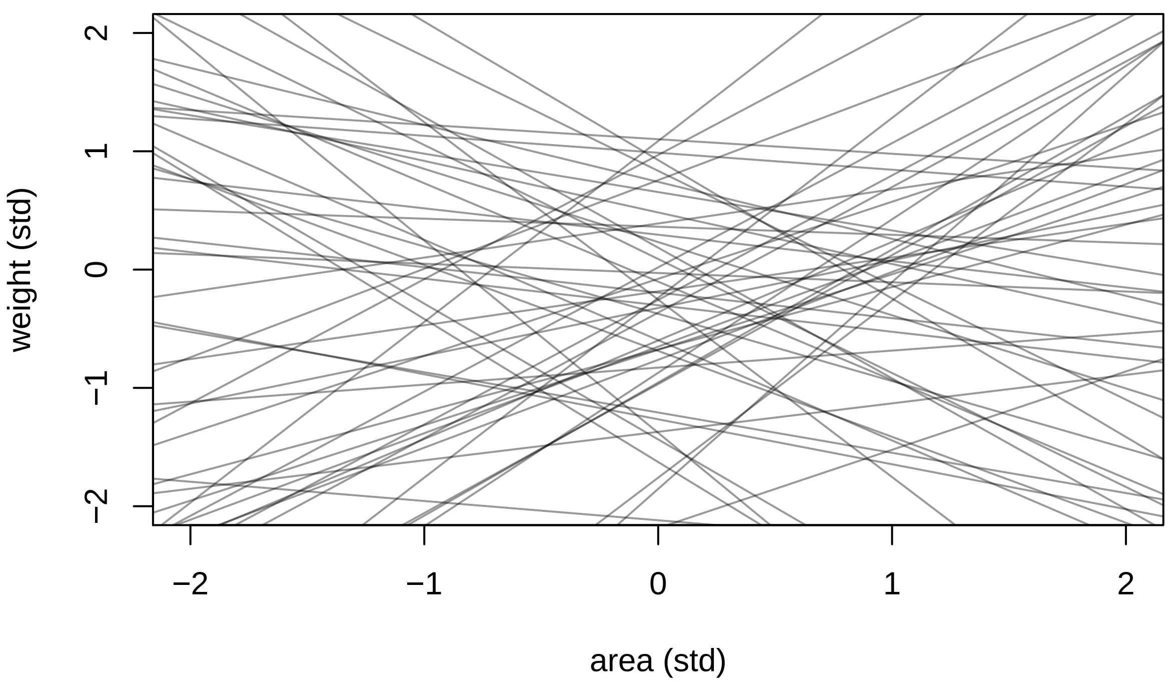
Compared to using the old priors of normal distributions, the posterior means are close.

1.3 Using less regularized priors of normal distributions

```
m1.2<- quap(
    alist(
        W ~ dnorm(mu,sigma),
        mu ~ a + b_A * A,
        a ~ dnorm(0,1),
        b_A ~ dnorm(0,1),
        sigma ~ dexp(1)
    ),
    data=d
)
precis(m1.2)
```

```
##               mean         sd       5.5%     94.5%
## a     2.726939e-07 0.09164903 -0.1464726 0.1464731
## b_A   1.931376e-02 0.09204328 -0.1277892 0.1664167
## sigma 9.912621e-01 0.06466583  0.8879136 1.0946106
```

```
# prior predictive simulation
set.seed(10)
prior <- extract.prior(m1.2)
mu <- link(m1.2, post=prior, data = list(A=c(-3,3)))
plot( NULL , xlim=c(-2,2) , ylim=c(-2,2), xlab="area (std)", ylab="weight (std)")
for (i in 1:50) lines(c(-3,3), mu[i,], col=col.alpha("black",0.4) )
```



Even though the priors of the slope and intercept are vaguer, the posterior means are close and the results are consistent.

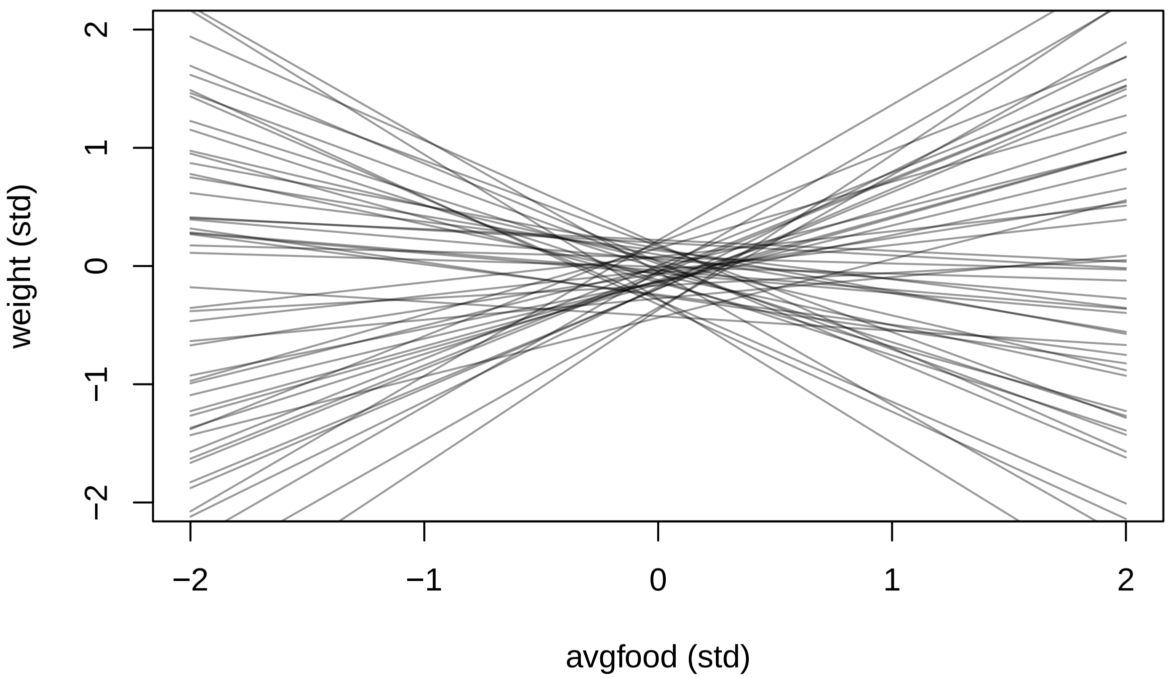
2. Modelling avgfood -> weight

2.1 Using priors of normal distributions

```
m2.0 <- quap(
    alist(
        W ~ dnorm( mu , sigma ),
        mu <- a + b_F * F,
        a ~ dnorm(0,0.2),
        b_F ~ dnorm(0,0.5),
        sigma ~ dexp(1)
    ), data=d )
precis(m2.0)
```

```
##                mean         sd       5.5%     94.5%
## a      2.401175e-07 0.08360025 -0.1336091 0.1336096
## b_F   -2.421146e-02 0.09088512 -0.1694634 0.1210405
## sigma  9.911451e-01 0.06465877  0.8878079 1.0944823
```

```
# prior predictive simulation
set.seed(10)
prior <- extract.prior( m2.0 )
mu <- link( m2.0 , post=prior , data=list( F=c(-3,3) ) )
plot( NULL , xlim=c(-2,2) , ylim=c(-2,2), xlab="avgfood (std)", ylab="weight (std)" )
for ( i in 1:50 ) lines( c(-2,2) , mu[i,] , col=col.alpha("black",0.4) )
```



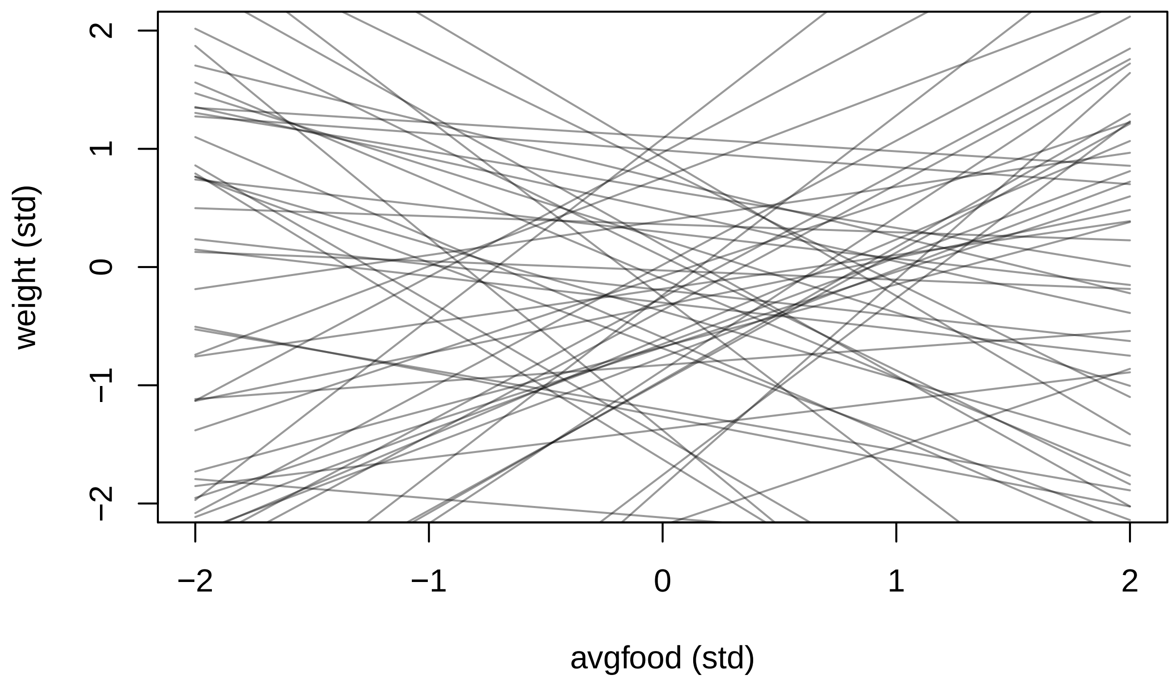
2.2 Using less regularized priors of normal distributions

```
m2.1 <- quap(
    alist(
        W ~ dnorm( mu , sigma ),
        mu <- a + b_F*F,
        a ~ dnorm(0,1),
```

```
        b_F ~ dnorm(0,1),
        sigma ~ dexp(1)
    ), data=d )
precis(m2.1)
```

```
##                mean         sd       5.5%     94.5%
## a      1.174875e-05 0.09165332 -0.1464680 0.1464915
## b_F   -2.483866e-02 0.09204759 -0.1719485 0.1222712
## sigma  9.913089e-01 0.06468541  0.8879291 1.0946887
```

```
# prior predictive simulation
set.seed(10)
prior <- extract.prior( m2.1 )
mu <- link( m2.1 , post=prior , data=list( F=c(-2,2) ) )
plot( NULL , xlim=c(-2,2) , ylim=c(-2,2), xlab="avgfood (std)", ylab="weight (std)" )
for ( i in 1:50 ) lines( c(-2,2) , mu[i,] , col=col.alpha("black",0.4) )
```



b_F is close to zero, based on the model, there's no causal influence of avgfood on weight. Compared to the more sensible priors from 2.1, the posterior means from vaguer priors are close.

3. Modelling groupsize -> weight

3.1 Using priors of normal distributions

Need to adjust for avgfood as a covariate.

```
# avgfood, groupsize -> weight
m3.0<- quap(
    alist(
        W ~ dnorm(mu,sigma),
        mu ~ a + b_F * F + b_G * G,
        a ~ dnorm(0,1),
        c(b_F,b_G) ~ dnorm(0,1),
        sigma ~ dexp(1)
```

```r
    ),
    data=d
)
# round( vcov( m3 ) , 4 )
# pairs(m3)
# prior predictive simulation
# set.seed(10)
# prior <- extract.prior( m3.0 )
# mu <- link( m3.0 , post=prior , data=list( F=c(-2,2), G=c(-2,2) ) )
# plot( NULL , xlim=c(-2,2) , ylim=c(-2,2), xlab="avgfood (std)", ylab="weight (std)" )
# for ( i in 1:50 ) lines( c(-2,2) , mu[i,] , col=col.alpha("black",0.4) )
precis(m3.0)
```

```
##                mean         sd       5.5%      94.5%
## a      2.440174e-07 0.08690205 -0.1388860  0.1388865
## b_F    5.914115e-01 0.19541332  0.2791033  0.9037197
## b_G   -6.888534e-01 0.19541490 -1.0011641 -0.3765426
## sigma  9.395180e-01 0.06134177  0.8414820  1.0375540
```

3.2 Using less regularized priors of normal distributions

```r
# avgfood, groupsize -> weight
m3.1<- quap(
    alist(
        W ~ dnorm(mu,sigma),
        mu ~ a + b_F * F + b_G * G,
        a ~ dnorm(0,1),
        c(b_F,b_G) ~ dnorm(0,1),
        sigma ~ dexp(1)
    ),
    data=d
)
precis(m3.1)
```

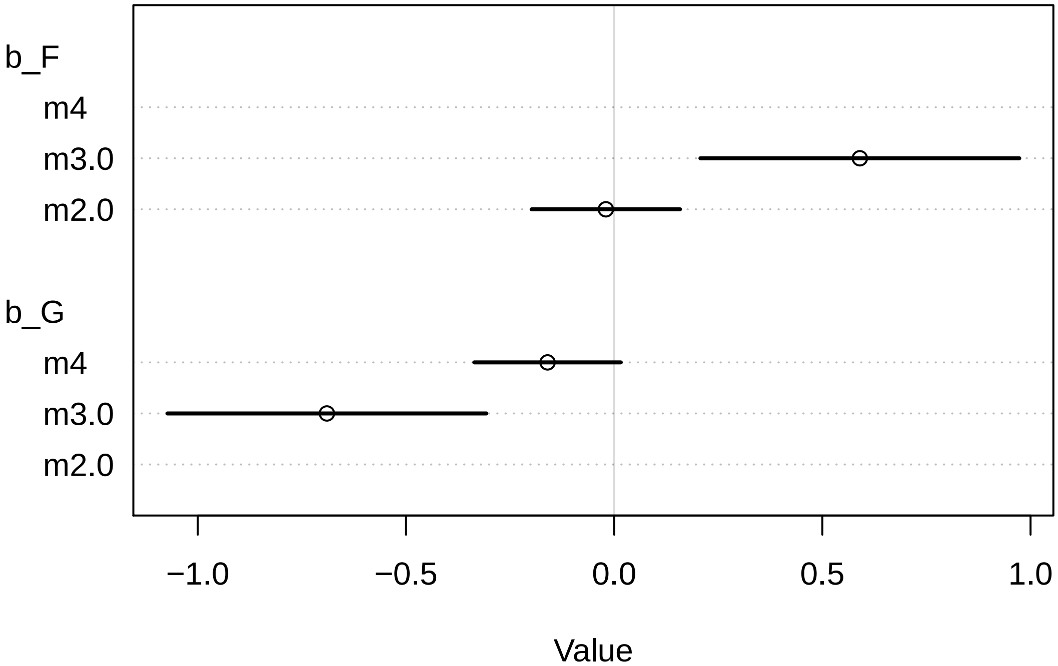
```
##                mean         sd       5.5%      94.5%
## a      6.345332e-07 0.08690203 -0.1388856  0.1388869
## b_F    5.914126e-01 0.19541328  0.2791044  0.9037207
## b_G   -6.888569e-01 0.19541485 -1.0011676 -0.3765462
## sigma  9.395178e-01 0.06134174  0.8414819  1.0375538
```

The posterior means from vaguer priors are very close.

3.3 Posterior distribution

```r
# groupsize -> weight
m4<- quap(
    alist(
        W ~ dnorm(mu,sigma),
        mu ~ a + b_G * G,
        a ~ dnorm(0,0.2),
        b_G ~ dnorm(0,0.5),
        sigma ~ dexp(1)
    ),
    data=d
```

```
)
plot( coeftab(m2.0, m3.0, m4) , pars=c("b_F","b_G") )
```

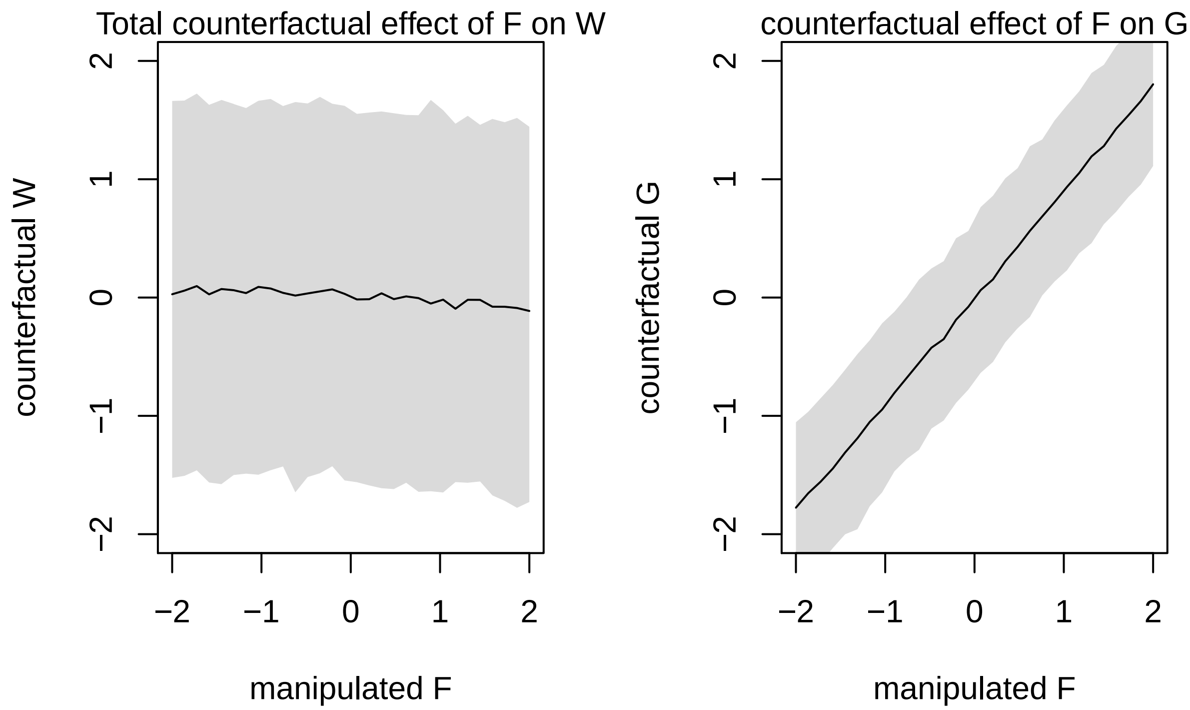


The model shows avgfood is positively associated with weight and groupsize is negatively associated with groupsize when controlling for each other. The posterior means of both associations increased after considering both predictor varables.

```
m5<- quap(
    alist(
        # avgfood, groupsize -> weight
        W ~ dnorm(mu,sigma),
        mu ~ a + b_F * F + b_G * G,
        a ~ dnorm(0,0.2),
        c(b_F,b_G) ~ dnorm(0,0.5),
        sigma ~ dexp(1),
        # avgfood -> groupsize
        G ~ dnorm(mu_F,sigma_F),
        mu_F ~ a_F + b_FG * F,
        a_F ~ dnorm(0,0.2),
        b_FG ~ dnorm(0,0.5),
        sigma_F ~ dexp(1)
    ),
    data=d
)
F_seq <- seq( from=-2 , to=2 , length.out=30 )
sim_dat <- data.frame( F=F_seq )
post <- extract.samples( m5 )
G_sim <- with( post , sapply( 1:30 ,
    function(i) rnorm( 1e3 , a_F + b_FG * F_seq[i] , sigma_F ) ) )
W_sim <- with( post , sapply( 1:30 ,
    function(i) rnorm( 1e3 , a + b_F*F_seq[i] + b_G*G_sim[,i] , sigma ) ) )
#display counterfactual predictions
par(mfrow=c(1,2))
```

```
plot( sim_dat$F , colMeans(W_sim) , ylim=c(-2,2) , type="l" ,
    xlab="manipulated F" , ylab="counterfactual W"  )
shade( apply(W_sim,2,PI) , sim_dat$F )
mtext( "Total counterfactual effect of F on W" )

plot( sim_dat$F , colMeans(G_sim) , ylim=c(-2,2) , type="l" ,
    xlab="manipulated F" , ylab="counterfactual G"  )
shade( apply(G_sim,2,PI) , sim_dat$F )
mtext( "counterfactual effect of F on G" )
```



The direct influence of food on weight is positive but the total influence of food is very small, showing a masking effect.