Supplementary Methods S1

Stan code

```
data {
  int<lower=1> N;         //individuals
  int<lower=1> rep_d;     //number of stool samples per individual
  int<lower=1> rep_s;     //number of slides per individual
  int<lower=1> S;         //number of mixture components
  int<lower=1> A;         //number of treatment arms
  int<lower=1> n_treat[A];   //number of individuals in each treatment arm
  int<lower=-1> y_bl[N*rep_d*rep_s];   //KK measurements at baseline
  int<lower=-1> y_fu[N*rep_d*rep_s];   //KK measurements at follow-up
  int<lower=1> index[N];
  //int<lower=1> ind_study[N];
}
parameters {
  // Baseline
  real<lower=0> mean_infect[A];
  real<lower=0> M[A];      //mean worm burden at follow-up
  real<lower=0> d_var;
  real<lower=0> count_var;
  real<lower=0> count_var_fu;
  vector<lower=0>[A] pop_var;
  vector<lower=0>[A] k_fu;             //Aggregation of worms
  vector<lower=0>[N] mean_i_bl;
  vector<lower=0>[N] mean_i_fu;
  vector[N*rep_d] log_d_var_ind_raw;
  vector[N*rep_d] log_d_var_ind_raw_fu;
  real<lower=0> conv;
  real<lower=0,upper=1> z;
}
```

```
transformed parameters{
  // Baseline
  real<lower=0> k_bl;
  vector<lower=0>[N*rep_d] d_var_ind;
  vector<lower=0>[N*rep_d] mean_id_bl;
  vector[N*rep_d] log_d_var_ind;

  //Follow-up
  real<lower=0> k_egg_fu[S];
  vector<lower=0>[N*rep_d] mean_id_fu[S];
  vector<lower=0>[N*rep_d] d_var_ind_fu;
  vector[N*rep_d] log_d_var_ind_fu;



  vector<lower=0>[A] mean_infect_fu;
  vector<lower=0>[A] phi;

  vector<lower=0, upper=1>[A] p;
  vector<lower=0, upper=1>[A] p_0;
  vector<lower=0>[A] mean_female;
  vector<lower=0>[A] var_female;
  vector<lower=0>[A] s;
  vector<lower=0>[A] r;

  vector<lower=0, upper=1>[A] prev;
  real<lower=0, upper=1> CR[A];

  row_vector<upper=1>[A] ERR;
  real<upper=1> ERR_vec1;
  real<upper=1> ERR_vec2;
  real<upper=1> ERR_vec3;
  real<upper=1> ERR_vec4;
```

```
real<lower=0,upper=1> CR_vec1;
real<lower=0,upper=1> CR_vec2;
real<lower=0,upper=1> CR_vec3;
real<lower=0,upper=1> CR_vec4;

// Baseline
log_d_var_ind = -pow(d_var,2)/2+d_var*log_d_var_ind_raw;
d_var_ind=exp(log_d_var_ind);
k_bl=1/count_var; //changed to invert


for(j in 1:rep_d){
  for(l in 1:N){
    mean_id_bl[(j-1)*N+l]=mean_i_bl[l]*d_var_ind[(j-1)*N+l];
  }
}

// Follow-up
k_egg_fu[1]=1000;
k_egg_fu[2]=1/count_var_fu;   //changed to invert

for (i in 1:A){
  p[i] = (k_fu[i]/(M[i]+k_fu[i]));
  p_0[i] = p[i]^k_fu[i];
  mean_female[i] = 2^k_fu[i]*conv*M[i]*z*(1/((2+M[i]*(1-z)/k_fu[i])^(k_fu[i]+1))-1/((2+M[i]*(2-
z)/k_fu[i])^(k_fu[i]+1)));
  var_female[i] =
2^k_fu[i]*conv^2*M[i]*z^2*((2*(k_fu[i]+M[i])+k_fu[i]*M[i]*z^2)/(k_fu[i]*(2+M[i]*(1-
z^2)/k_fu[i])^(k_fu[i]+2))-(2*(k_fu[i]+M[i])+k_fu[i]*M[i]*z^2)/(k_fu[i]*(2+M[i]*(2-
z^2)/k_fu[i])^(k_fu[i]+2)));
  s[i] = mean_female[i]^2/var_female[i];
  r[i] = mean_female[i]/var_female[i];
}
```

```
  log_d_var_ind_fu = -pow(d_var,2)/2+d_var*log_d_var_ind_raw_fu;
  d_var_ind_fu=exp(log_d_var_ind_fu);

  for(j in 1:rep_d){
    for(l in 1:N){
      mean_id_fu[1,(j-1)*N+l] = 0.001;
      mean_id_fu[2,(j-1)*N+l]=mean_i_fu[l]*d_var_ind_fu[(j-1)*N+l];
    }
  }

  for(i in 1:A) prev[i]=1-2*(k_fu[i]/(k_fu[i]+M[i]/2))^k_fu[i]+p_0[i];
  for(i in 1:A) mean_infect_fu[i]=mean_female[i]*prev[i];

  for (i in 1:A) phi[i]=mean_infect_fu[i]/mean_infect[i];

  for (i in 1:A) ERR[i]=1-phi[i];
  for (i in 1:A) CR[i]=1-prev[i];



  CR_vec1 = (CR[6]*n_treat[6]+CR[16]*n_treat[16])/(n_treat[6]+n_treat[16]);       // Alb. +
Oxan. 20
  CR_vec2 = (CR[1]*n_treat[1]+CR[5]*n_treat[5])/(n_treat[1]+n_treat[5]);          // Iver. 200
  CR_vec3 =
(CR[13]*n_treat[13]+CR[17]*n_treat[17]+CR[28]*n_treat[28])/(n_treat[13]+n_treat[17]+n_trea
t[28]);   // Meb 500
  CR_vec4 = (CR[11]*n_treat[11]+CR[25]*n_treat[25])/(n_treat[11]+n_treat[25]);    // Oxan. 20



  ERR_vec1 = (ERR[6]*n_treat[6]+ERR[16]*n_treat[16])/(n_treat[6]+n_treat[16]);       // Alb. +
Oxan. 20
  ERR_vec2 = (ERR[1]*n_treat[1]+ERR[5]*n_treat[5])/(n_treat[1]+n_treat[5]);          // Iver.
200
```

```
  ERR_vec3 =
(ERR[13]*n_treat[13]+ERR[17]*n_treat[17]+ERR[28]*n_treat[28])/(n_treat[13]+n_treat[17]+n
_treat[28]);   // Meb 500
  ERR_vec4 = (ERR[11]*n_treat[11]+ERR[25]*n_treat[25])/(n_treat[11]+n_treat[25]);    //
Oxan. 20



}


model{
  real lps[S];

  // Baseline
  d_var ~ gamma(1,1);
  pop_var ~ exponential(2);
  //pop_var ~ normal(0,1);
  count_var ~ normal(0,1);
  count_var_fu ~ normal(0,1);

  conv ~ normal(1,1);

  z ~ beta(10,1);

  mean_infect ~ gamma(2,0.04);

  M ~ gamma(0.08,0.04);

  k_fu ~ normal(0.3,0.5);

  log_d_var_ind_raw ~ normal(0,1);
  log_d_var_ind_raw_fu ~ normal(0,1);
```

```
  for (i in 1:N) mean_i_bl[i] ~
gamma(mean_infect[index[i]]*pop_var[index[i]],pop_var[index[i]]);
  for (i in 1:N) mean_i_fu[i] ~ gamma(s[index[i]],r[index[i]]);


  for(i in 1:N){
    for(h in 1:rep_d){
      for(l in 1:rep_s){
        if(y_bl[i+(h-1)*rep_s*N+(l-1)*N]==-1){}
        else{
          y_bl[i+(h-1)*rep_s*N+(l-1)*N] ~ neg_binomial_2_lpmf(mean_id_bl[i+(h-1)*N], k_bl);
        }
      }
    }
  }



  for (i in 1:N) {
    lps[1] = log(1-prev[index[i]]);
    lps[2] = log(prev[index[i]]);
    for (j in 1:S){
      for(h in 1:rep_d){
        for (l in 1:rep_s){
          if(y_fu[i+(h-1)*rep_s*N+(l-1)*N]==-1){}
          else{
            lps[j] = lps[j] + neg_binomial_2_lpmf(y_fu[i+(h-1)*rep_s*N+(l-1)*N] |
mean_id_fu[j,i+(h-1)*N], k_egg_fu[j]);
          }
        }
      }
    }
    target += log_sum_exp(lps);
  }
```

```
}

generated quantities{
  real<lower=0> var_group_bl[A];
  real<lower=0, upper=1> CR_weighted[24];
  real<upper=1> ERR_weighted[24];

  CR_weighted[1] = CR_vec2;
  CR_weighted[2] = CR[2];
  CR_weighted[3] = CR[3];
  CR_weighted[4] = CR[4];
  CR_weighted[5] = CR_vec1;
  CR_weighted[6] = CR[7];
  CR_weighted[7] = CR[8];
  CR_weighted[8] = CR[9];
  CR_weighted[9] = CR[10];
  CR_weighted[10] = CR_vec4;
  CR_weighted[11] = CR[12];
  CR_weighted[12] = CR_vec3;
  CR_weighted[13] = CR[14];
  CR_weighted[14] = CR[15];
  CR_weighted[15] = CR[18];
  CR_weighted[16] = CR[19];
  CR_weighted[17] = CR[20];
  CR_weighted[18] = CR[21];
  CR_weighted[19] = CR[22];
  CR_weighted[20] = CR[23];
  CR_weighted[21] = CR[24];
  CR_weighted[22] = CR[26];
  CR_weighted[23] = CR[27];
  CR_weighted[24] = CR[29];
```

```
  ERR_weighted[1] = ERR_vec2;
  ERR_weighted[2] = ERR[2];
  ERR_weighted[3] = ERR[3];
  ERR_weighted[4] = ERR[4];
  ERR_weighted[5] = ERR_vec1;
  ERR_weighted[6] = ERR[7];
  ERR_weighted[7] = ERR[8];
  ERR_weighted[8] = ERR[9];
  ERR_weighted[9] = ERR[10];
  ERR_weighted[10] = ERR_vec4;
  ERR_weighted[11] = ERR[12];
  ERR_weighted[12] = ERR_vec3;
  ERR_weighted[13] = ERR[14];
  ERR_weighted[14] = ERR[15];
  ERR_weighted[15] = ERR[18];
  ERR_weighted[16] = ERR[19];
  ERR_weighted[17] = ERR[20];
  ERR_weighted[18] = ERR[21];
  ERR_weighted[19] = ERR[22];
  ERR_weighted[20] = ERR[23];
  ERR_weighted[21] = ERR[24];
  ERR_weighted[22] = ERR[26];
  ERR_weighted[23] = ERR[27];
  ERR_weighted[24] = ERR[29];

  for (i in 1:A) var_group_bl[i]=mean_infect[i]/pop_var[i];
}
```