```
model {
  for (i in 1:nsub) {
    for (j in 1:4) {
      # model for outcome UPDRS
      upd[i,j] ~ dnorm(mupd[i,j], taupd)
      mupd[i,j] <- a[1] + b[1]*theta[i,j]

      # model for outcome fatigue
      fat[i,j] ~ dnorm(mu_fat[i,j], taufat)
      mufat[i,j] <- a[2] + b[2]*theta[i,j]

      # model for outcome QOL
      qol[i,j] ~ dnorm(muqol[i,j], taudy[i,j])
      taudy[i,j] <- w[i,j] * tauqol
      w[i,j] ~ dgamma(nu, nu) # student's t distribution
      # w[i,j] ~ dgamma(nu, 1)  # slash distribution
      muqol[i,j] <- a[3] + b[3]*theta[i,j]

      # model for outcome Schwab and England ADL
      sch[i,j] ~ dcat(p1[i,j,])
      # specify p1 for sch
      for (l in 1:(ncat-1)) { logit(q1[i,j,l]) <- a4[l] - b[4]*theta[i,j] }
      p1[i,j,1] <- q1[i,j,1]
      for (l in 2:(ncat-1)) {
        p1[i,j,l] <- q1[i,j,l] - q1[i,j,l-1]
      }
      p1[i,j,ncat] <- 1 - q1[i,j,(ncat-1)]

      # model for the latent disease severity
      theta[i,j] <- u[i,1] + (beta0 + beta1*trt[i] + u[i,2])*t[j]
    } # end of loop j

  # prior of u
    u[i,1:2] ~ dmnorm(zero[], precision[,])
  } # end of loop i

  # construct the covariance matrix
  precision[1:2,1:2] <- inverse(sigma[,])
  sigma[1,1] <- 1
  sigma[1,2] <- rho * sig1
  sigma[2,1] <- sigma[1,2]
  sigma[2,2] <- sig1 * sig1

  # prior for rho and sig1
  rho ~ dunif(-1, 1)
  sig1 ~ dgamma(0.001, 0.001)

  # prior for nu
  nu ~ dgamma(0.001, 0.001)
  ka <- nu * 2

  # prior for beta
  beta0 ~ dnorm(0, 0.01)
```

```
  beta1 ~ dnorm(0, 0.01)

  # prior for difficulty parameter a and discriminating parameter b
  for (i in 1:3) { a[i] ~ dnorm(0, 0.0001) }
  for (k in 1:4) { b[k] ~ dgamma(0.001, 0.001) }

  # prior for variances of the continuous outcomes
  taupd ~ dgamma(0.001, 0.001)
  sigpd <- 1/taupd
  taufat ~ dgamma(0.001, 0.001)
  sigfat <- 1/taufat
  tauqol ~ dgamma(0.001, 0.001)
  sigqol <- 1/tauqol

  # priors for the difficulty parameter a of outcome Schwab and England ADL
  a4[1] ~ dnorm(0, 0.01)
  for (l in 2:(ncat-1)) { a4[l] <- a4[l-1] + delta1[l-1] }
  for (k in 1:(ncat-2)) { delta1[k] ~ dnorm(0, 0.01)I(0,) }
}
```