Basic Functions

Florian Stijven

9-3-2022

In this file, the basic functions are defined which are further needed to implement the proposed methods.

```
library(mvtnorm)
library(purrr)
```

Likelihood

The model likelihood is defined given all parameters. This likelihood consists of two separate contributions. The contributions for patients in the treatment and control group are distinct. The unidentifiable correlation parameter don't have an influence on the likelihood and are thus not included in the function. However, the identifiable correlations do impose a restriction on what values for the unidentifiable correlations are allowed.

First, the gaussian copula likelihood is defined.

```
# likelihood contribution for one observation
# either (S_0, T_0), or (S_1, T_1) are observed
copula_loglik_comp = function(u, v, rho_comp){
  #rho_comp corresponds to the correlation between the surrogate and
  #true endpoint in the respective group
  #u and v are the cdf transformed values for the surrogate and true endpoint
  Sigma = matrix(data = c(1, rho_comp, rho_comp, 1), nrow = 2)
  q_u = qnorm(u)
  q_v = qnorm(v)
  num = dmvnorm(x = c(q_u, q_v), mean = c(0, 0), sigma = Sigma, log = TRUE)
  denom = dnorm(q_u)*dnorm(q_v)
  return(num/denom)
}

# all identifiable correlations are parameters of this function
copula_lik = function(data, rho13, rho24){
  #in data, the first column should contain u, second v, and third treatment indicator
  control_data = data[,data[,3] == 0]
  treated_data = data[,data[,3] == 0]
  log_lik = sum(map2_dbl(.x = control_data[,1], .y = control_data[,2],
               .f = copula_loglik_comp, rho_comp = rho13))
  log_lik = log_lik +
    sum(map2_dbl(.x = treated_data[,1], .y = treated_data[,2],
               .f = copula_loglik_comp, rho_comp = rho13))
  return(log_lik)
}
```

The copula likelihood multiplied with the marginal densities is the likelihood for the original data. The total loglikelihood is as a consequence the sum of the copula loglikelihood and the loglikelihood based on the marginal densities.

For now, the marginal densities are defined as weibull distributions (parameterisation see Wikipedia).

```
full_loglik_weibull = function(data, rho13, rho24,
                               lambda_s0, lambda_s1, lambda_t0, lambda_t1,
                               k_s0, k_s1, k_t0, k_t1){
  #data in cdf transformed format
  data_copula = cdf_trans_weibull_all(data,
                                      lambda_s0, lambda_s1, lambda_t0, lambda_t1,
                                      k_s0, k_s1, k_t0, k_t1)
  #copula log likelihood
  copula_loglik = copula_loglik(data, rho13, rho24)
  #sum of marginal log likelihoods
  marginal_loglik = marg_loglik_all(data,
                                    lambda_s0, lambda_s1, lambda_t0, lambda_t1,
                                    k_s0, k_s1, k_t0, k_t1)
  return(copula_loglik + marginal_loglik)
}

#marginal log likelihood for one observation
marg_loglik_obs = function(s, t,
                           lambda_s, k_s, lambda_t, k_t){
  loglik_s = dweibull(x = s, shape = k_s, scale = lambda_s, log = TRUE)
  loglik_t = dweibull(x = t, shape = k_t, scale = lambda_t, log = TRUE)
  return(loglik_t + loglik_s)
}

#marginal log likelihood for all observations
marg_loglik_all = function(data,
                           lambda_s0, lambda_s1, lambda_t0, lambda_t1,
                           k_s0, k_s1, k_t0, k_t1){
  loglik = 0
  n = nrow(data)
  for (i in 1:n){
    s = data[i, 1]
    t = data[i, 2]
    if(data[i, 3] == 0){
      loglik = loglik + marg_loglik_obs(s, t, lambda_s0, k_s0,
                                              lambda_t0, k_t0)
    }
    else{
      loglik = loglik + marg_loglik_obs(s, t, lambda_s1, k_s1,
                                              lambda_t1, k_t1)
    }
  }
  return(loglik)
}

#transform one observation
cdf_trans_weibull_obs = function(s, t,
                                 lambda_s, k_s, lambda_t, k_t){
```

```r
  return(pweibull(s, shape = k_s, scale = lambda_s),
           pweibull(t, shape = k_t, scale = lambda_t))
}

#transform all observations
cdf_trans_weibull_all = function(data,
                                 lambda_s0, lambda_s1, lambda_t0, lambda_t1,
                                 k_s0, k_s1, k_t0, k_t1){
  n = nrow(data)
  data_transformed = data
  for (i in 1:n){
    s = data[i, 1]
    t = data[i, 2]
    if(data[i, 3] == 0){
      data_transformed[i, 1:2] = cdf_trans_weibull_obs(s, t, lambda_s0, k_s0,
                                                       lambda_t0, k_t0)
    }
    else{
      data_transformed[i, 1:2] = cdf_trans_weibull_obs(s, t, lambda_s1, k_s1,
                                                       lambda_t1, k_t1)
    }
  }
}
```

Simulation