Adjusting for response misclassification with the sslogit link

Accounting for imperfect sensitivity and specificity in the response

To use the `sslogit` link in INLA this needs to be enabled as follows:

```
library(INLA)
```

```
inla.models <- INLA:::inla.get.inlaEnv()$inla.models
inla.models$link$sslogit$status <- NULL
assign("inla.models", inla.models, env = INLA:::inla.get.inlaEnv())
rm(inla.models)
```

Be aware that the reason this link is not available by default is that it struggles numerically if the sensitivity and specificity is large. SO USE WITH CAUTION.

```
set.seed(1)
```

We simulate a misclassified response. `p_y` is the success probability for the correct version of the response, `y`. `s` is the misclassified response, and `p_s` is the success probability for `s`.

```
n <- 1000

p_y <- 0.1 # Success probability for y
sens <- 0.95 # P(y = 1 | s = 1)
spec <- 0.90 # P(y = 0 | s = 0)
p_s <- p_y*sens + (1-p_y)*(1-spec) # Success probability for s

df <- data.frame(s = rbinom(n = n, size = 1, prob = p_s))
```

Given this data, we can fit the naive model `r0`, which uses `s` without adjusting for misclassification, and the adjusted model `r1`, which adjusts for the misclassification by using and adjusted link function, as described in Section 4.1.

```
formula <- s ~ 1

# Model 0 (ignoring sens and spec)
r0 <- inla(formula = formula, data = df, family = "binomial",
          Ntrials = 1)

# Model 1 (adding sens and spec)
r1 <- inla(formula = formula, data = df,
           family = "binomial",
           control.family = list(
             control.link = list(
               model = "sslogit",
               hyper = list(
                 sens = list(
                   prior = "logitbeta",
                   initial = inla.link.logit(sens),
                   fixed = TRUE
                 ),
                 spec = list(
                   prior = "logitbeta",
                   initial = inla.link.logit(spec),
                   fixed = TRUE)
               )
             )
           )
         )
```

```
r0$summary.fixed
```

```
##                  mean        sd 0.025quant 0.5quant 0.975quant     mode kld
## (Intercept) -1.50499 0.0819573  -1.665623 -1.50499  -1.344356 -1.50499   0
```

```
r1$summary.fixed
```

```
##                  mean        sd 0.025quant 0.5quant 0.975quant     mode kld
## (Intercept) -2.24994 0.1225932  -2.490218 -2.24994  -2.009661 -2.24994   0
```

Alternative approach when sensitivity and specificity are not known exactly

For a third approach, we fit the model for a grid of different sensitivities and specificities, as described in the end of Section 4.1.

```
IC.sens = c(0.925, 0.975)
IC.spec = c(0.85, 0.95)

# Latin square grid
n.latin <- 50
sens.values <- seq(from = IC.sens[1],
                   to = IC.sens[2],
                   length.out = n.latin)
spec.values <- seq(from = IC.spec[1], to = IC.spec[2],
                   length.out = n.latin)

# For the Latin-square
reord = sample(1:n.latin)
SSgrid = data.frame(sens = sens.values,
                    spec = spec.values[reord])
# Weights
# Using independent normals
sd.sens = (IC.sens[1] - sens) / qnorm(0.025)
sd.spec = (IC.spec[1] - spec) / qnorm(0.025)
SS.weights = dnorm(SSgrid$sens, sens, sd.sens) * dnorm(SSgrid$spec, spec, sd.spec)
SS.weights = SS.weights / sum(SS.weights)
```

We run INLA for each pair of sensitivity and specificity of the Latin square, and keep the results in a list:

```
inla.SSlogit.grid <- function(SS, data, formula){
  r.temp = inla(data = data,
                formula = formula,
                family = "binomial",
                control.family = list(
                  control.link = list(
                    model = "sslogit",
                    hyper = list(
                      sens = list(
                        prior = "logitbeta",
                        initial = inla.link.logit(SS$sens),
                        fixed = TRUE),
                      spec = list(
                        prior = "logitbeta",
                        initial = inla.link.logit(SS$spec),
                        fixed = TRUE)
                      )
                    )
                )
  )
  r.temp
}

r.list <- lapply(X = 1:nrow(SSgrid),
                 FUN = function(x){
                   inla.SSlogit.grid(SSgrid[x,],
                                     formula = formula,
                                     data = df)
                   }
                 )

# Merging all INLA outputs
r2 <- inla.merge(loo = r.list, prob = SS.weights)
```

Comparing all approaches

The model that does no adjustment:

```
r0$summary.fixed[,c(1,3,5)]
```

```
##                 mean 0.025quant 0.975quant
## (Intercept) -1.50499  -1.665623  -1.344356
```

The model that adjusts for misclassification using fixed values for sensitivity and specificity:

```
r1$summary.fixed[,c(1,3,5)]
```

```
##                 mean 0.025quant 0.975quant
## (Intercept) -2.24994  -2.490218  -2.009661
```

The model that uses a grid of sensitivity and specificity:

```
# Approximate 95% interval
c(r2$summary.fixed$mean, r2$summary.fixed$mean + c(-2, 2)*r2$summary.fixed$sd)
```

```
## [1] -2.308922 -2.931946 -1.685898
```

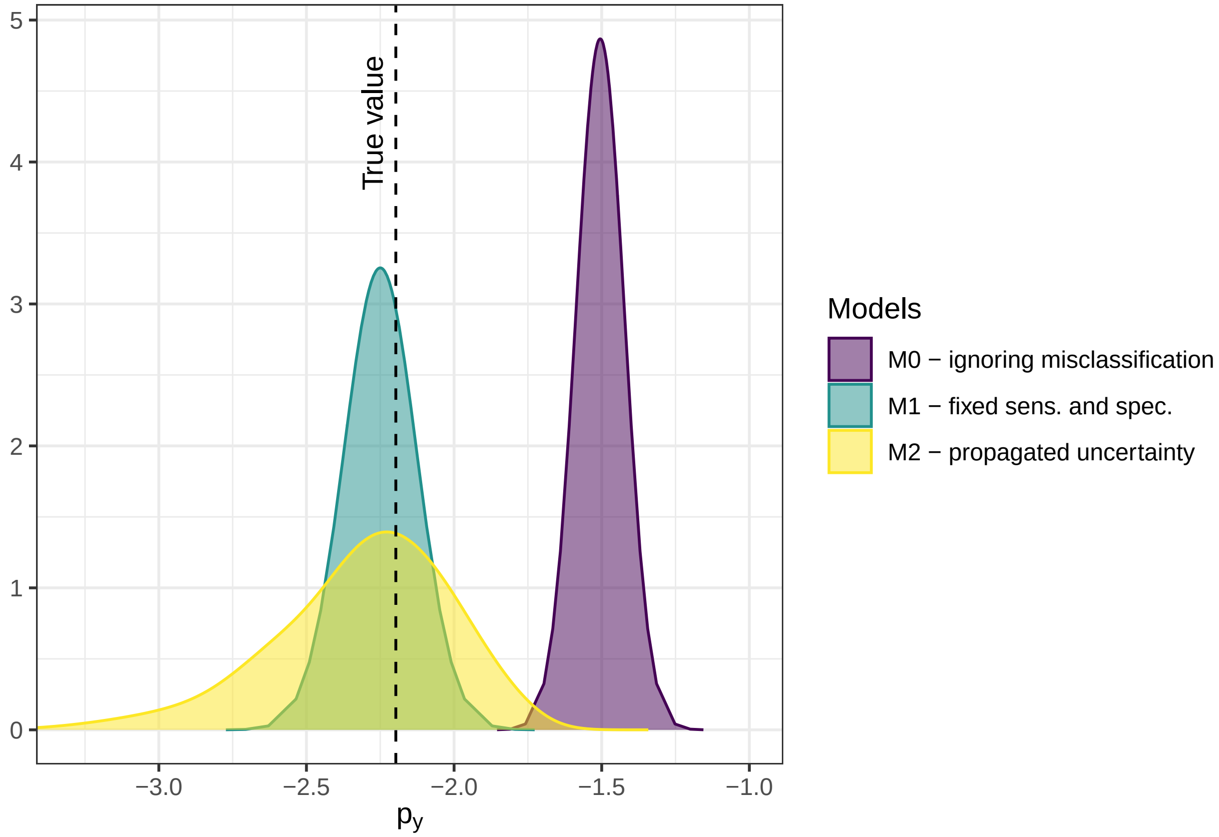
```
# Using inla.qmarginal
c(r2$summary.fixed$mean,
  inla.qmarginal(p = c(0.025, 0.975),
                 marginal = r2$marginals.fixed$`(Intercept)`))
```

```
## [1] -2.308922 -3.027396 -1.790307
```

```
marginals.df <- dplyr::bind_rows(
  'M0 - ignoring misclassification' = data.frame(r0$marginals.fixed$`(Intercept)`),
  'M1 - fixed sens. and spec.' = data.frame(r1$marginals.fixed$`(Intercept)`),
  'M2 - propagated uncertainty' = data.frame(r2$marginals.fixed$`(Intercept)`),
  .id = "Models")

library(ggplot2)

ggplot(marginals.df, aes(x = x, y = y, color = Models, fill = Models)) +
  #geom_line() +
  geom_area(alpha = 0.5, position = "identity") +
  geom_vline(xintercept = inla.link.logit(p_y), show.legend = F,
             linetype = "dashed") +
  annotate("text", x = inla.link.logit(p_y) - 0.08, y = 3.8, label="True value",
           hjust = 0, angle = 90) +
  scale_color_viridis_d() + scale_fill_viridis_d() +
  coord_cartesian(xlim = c(-3.3, -1)) +
  theme_bw() +
  xlab(bquote(p[y])) +
  theme(legend.position.inside = c(0.2, 0.8),
        axis.title.y = element_blank())
```



```
ggsave("figures/response_mc.pdf", height = 4, width = 7)
```