Labelling self-tracked menstrual health records with hidden semi-Markov models

Supplementary Material

Laura Symul, Susan Holmes

This is the pdf-rendering of the analyses performed in R for the manuscript "Labeling self-tracked menstrual health records with hidden semi-Markov models". The R markdown (.Rmd) files used to generate this pdf can be found on [https://github.com/lasy/semiM-Public-Repo]. Each section is a standalone Rmd file and can be run separately. On the same github repository, one can also find binary files with the specified models and the data related to each section, with the exception of the raw Kindara app data as user's time-series cannot be shared publicly to respect users' privacy and data ownership.

Contents

List of Tables

List of Figures

1 Hidden semi-Markov model for labelling fertility-app data with reproductive events

In this section, we specify a hidden-semi Markov model (HSMM) that will be used to decode time-series logged by users of a fertility/menstrual cycle tracking app (Kindara).

The hidden states of this model are tight to biological events such as the menses, the follicular phase, ovulation, the luteal phase, pregnancies (ending in a birth or a loss), breast-feeding or anovulatory cycles.

The variables (observations) are fertility-related body-signs that users can log in the tracking app. For example, users can track their bleeding flow (none, spotting, light, medium or heavy), the quality and quantity of their cervical mucus, their temperature at wake-up and results of ovulation or pregnancy tests. These tests are usually at-home kits which detect, in urine samples, either lutenizing hormones (LH), which usually surges within a day of ovulation, or human chrorionic gonadotropin (HCG), which is produced when a fertilized egg implants in the uterine wall. Note that LH tests may also detect HCG, given the similarity between the two hormones.

While these features are typically tracked by individuals using "Fertility Awareness Methods" (FAM) for their family planning (conception/contraception), there is a wide diversity in tracking habits and tracking purposes among the Kindara users. Some users only use the app to track their period, while other users take full advantage of all the tracking options offered by the app and track with the goal of increasing or decreasing their chances of pregnancy. Our modeling framework allows to model and fit the probabilities with which a user opens the app or measures/reports a given variable in each state.

1.0.1 States definition, transition and initial probabilities and sojourn distributions

Our reproductive-events model is a 19-state hidden semi-Markov model. The model parameters are initialized and specified so that these 19 states matches biological/physiological states of the reproductive cycles. Table 1.1 provides the list of these states and their description. Table 1.2 and 1.3 provides the initial and transition probabilities associated with each state while figure 1.4 provides a visualization of the state sojourn distributions.

Table 1.1: States of the hidden semi-Markov model of reproductive events

i	abbr	names	description
1	**M**	Menses	Shedding of the uterine wall, resulting in menstrual bleeding, after an ovulatory or an anovulatory cycle.
2	**lE**	Early Follicular	Estrogen levels are still low and fertile mucus is not yet observable.
3	**hE**	Late Follicular	Estrogen levels are rising and fertile mucus is observable.
4	**preO**	Ovu - 1	One-day state preceding ovulation to account for a higher probability of a positive LH test compared to previous days.
5	**O**	Ovulation	One-day state during which an egg is released from one of the ovaries.
6	**postO**	Ovu + 2	Two-day state during which the mucus becomes less transparent and more sticky and during which the wake-up body temperature increases.
7	**Lut**	Luteal	Temperature is high (from progesterone), mucus is absent, sticky or creamy and spotting may be reported towards in the last days.
8	**Ano**	Anovulatory cycle	The anovulatory cycle state follows the follicular phase if a low temperature is consistently observed until the next menses.
9	**AB**	Anovulatory with bleeding	The anovulatory with bleeding state describes anovulatory cycles in which the users experience quasi-constant bleeding. It follows the menses and the temperature is low throughout this state.
10	**P**	Implantation (Pregnancy)	The implantation state follows the post-ovulation state. Temperature is high and pregnancy tests may be positive 10-15 days after fertilization (ovulation).
11	**PL**	Pregnancy with Loss	After implantation, a pregnancy can end in a spontaneous or induced loss.
12	**L**	Loss	The embryo leaves the uterus, bleeding is usually reported.
13	**lEpL**	Post-Loss	The post-loss state is similar to the early follicular phase except that pregnancy tests may still be positive (residual HCG in the urine).
14	**PB1**	Pregnancy with Birth - 1st trimester	Pregnancies ending with a birth are divided into three trimesters. Temperature is very high in the 1st trimester. It has a fixed duration of 84 days (12 weeks).
15	**PB2**	Pregnancy with Birth - 2nd trimester	Second trimester of the pregnancy, medium temperature. Duration is exactly 12 weeks.
16	**PB3**	Pregnancy with Birth - 3rd trimester	Thirst trimester of the pregnancy, low temperature. Duration is ~12 weeks, but is variable to account for pre- and post-term births.
17	**B**	Birth	Birth state is when birth occurs. Bleeding may be reported by users.
18	**PP**	Post-Partum	After a pregnancy, a mother can either breast-feed or not. If not, the post-partum period is ~ 6-8 weeks before returning to ovulatory cycles.
19	**BF**	Breast-Feeding	Breast-feeding suppresses the return of ovulation for long period of times.

Table 1.2: Specified values of the initial probabilities for each state of the HSMM of reproductive events.

M	lE	hE	preO	O	postO	Lut	Ano	AB	P	PL	L	lEpL	PB1	PB2	PB3	B	PP	BF
0.88	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.00	0.01	0.01	0.00	0.00	0.00	0.01	0.01	0.01

Table 1.3: Specified values of the transition probabilities between states of the HSMM of reproductive events.

	M	lE	hE	preO	O	postO	Lut	Ano	AB	P	PL	L	lEpL	PB1	PB2	PB3	B	PP	BF
M	0	0.99	0	0.00	0	0	0.0	0.00	0.01	0.0	0.0	0	0	0.0	0	0	0	0.0	0.0
lE	0	0.00	1	0.00	0	0	0.0	0.00	0.00	0.0	0.0	0	0	0.0	0	0	0	0.0	0.0
hE	0	0.01	0	0.97	0	0	0.0	0.02	0.00	0.0	0.0	0	0	0.0	0	0	0	0.0	0.0
preO	0	0.00	0	0.00	1	0	0.0	0.00	0.00	0.0	0.0	0	0	0.0	0	0	0	0.0	0.0
O	0	0.00	0	0.00	0	1	0.0	0.00	0.00	0.0	0.0	0	0	0.0	0	0	0	0.0	0.0
postO	0	0.00	0	0.00	0	0	0.9	0.00	0.00	0.1	0.0	0	0	0.0	0	0	0	0.0	0.0
Lut	1	0.00	0	0.00	0	0	0.0	0.00	0.00	0.0	0.0	0	0	0.0	0	0	0	0.0	0.0
Ano	1	0.00	0	0.00	0	0	0.0	0.00	0.00	0.0	0.0	0	0	0.0	0	0	0	0.0	0.0
AB	1	0.00	0	0.00	0	0	0.0	0.00	0.00	0.0	0.0	0	0	0.0	0	0	0	0.0	0.0
P	0	0.00	0	0.00	0	0	0.0	0.00	0.00	0.0	0.3	0	0	0.7	0	0	0	0.0	0.0
PL	0	0.00	0	0.00	0	0	0.0	0.00	0.00	0.0	0.0	1	0	0.0	0	0	0	0.0	0.0
L	0	0.00	0	0.00	0	0	0.0	0.00	0.00	0.0	0.0	0	1	0.0	0	0	0	0.0	0.0
lEpL	0	0.00	1	0.00	0	0	0.0	0.00	0.00	0.0	0.0	0	0	0.0	0	0	0	0.0	0.0
PB1	0	0.00	0	0.00	0	0	0.0	0.00	0.00	0.0	0.0	0	0	0.0	1	0	0	0.0	0.0
PB2	0	0.00	0	0.00	0	0	0.0	0.00	0.00	0.0	0.0	0	0	0.0	0	1	0	0.0	0.0
PB3	0	0.00	0	0.00	0	0	0.0	0.00	0.00	0.0	0.0	0	0	0.0	0	0	1	0.0	0.0
B	0	0.00	0	0.00	0	0	0.0	0.00	0.00	0.0	0.0	0	0	0.0	0	0	0	0.2	0.8
PP	0	0.00	1	0.00	0	0	0.0	0.00	0.00	0.0	0.0	0	0	0.0	0	0	0	0.0	0.0
BF	0	0.00	1	0.00	0	0	0.0	0.00	0.00	0.0	0.0	0	0	0.0	0	0	0	0.0	0.0

The `HiddenSemiMarkov` package allows to describe sojourns distributions as parametric distributions or non-parametric distributions. If described as parametric distributions, the distribution parameters are re-estimated at each M-step of the EM procedure when fitting the model to data, ensuring that the sojourn distributions keep following the specified parametric distribution. When specified as non-parametric distribution, the shape of the distribution is unconstrained. In both case, the initial sojourn distribution can also be used as a prior when updating the distributions in the EM-procedure if this option is selected when specifying a model.

The initial (and prior) sojourn distributions of each state are specified to reflect as fairly as possible existing knowledge about these different biological states. Below, we provide references for distributions from the medical literature and from previous papers using similar data. These distributions are displayed visually in Figure 1.4.

Menses have a typical duration between 2 and 8 days.(1) The early follicular phase is characterized by low, slowly increasing, estradiol levels and medium-high FSH levels. This is the most variable phase of the menstrual cycle and has a typical duration between 3 and 8 days.(2) In the late follicular phase, estradiol levels are rising sharply, FSH levels are decreasing. This phase last 2-5 days.(2). The sojourn of "pre-O" is arbitrarily fixed to 1 day. This state is specified distinct from the late follicular phase between the probability of a positive LH test differs between these two states. The sojourn of the ovulation state is fixed to 1 day as ovulation is a brief event and that the temporal resolution of our data is of 1 day. The duration of the luteal phase, which starts after ovulation, is known to vary less inter- and intra-individually than the follicular phase.(*3–8*) In the luteal phase, the basal body temperature is higher than in the follicular phase. However, past studies have shown that it takes a few days before temperature reaches its highest plateau. Thus, we divided the luteal phases into two states. The first one ("post-O"), of fixed duration (2 days), follows the ovulation state. The second one, the "Luteal" state, lasts about 11 days and is described by a gamma distribution to allow for a slight skew towards longer durations. The state "Ano", which is associated with potential anovulatory cycles and which follows the early follicular phase, is here described with a duration of approximately 15 days with a variance of about 6 days. Anovulatory cycles are not well described in the literature, owning to the difficulty to assess the absence of ovulation. Many methods relying on progesterone levels or temperature measurements usually set a threshold, which is not universally defined (experts around the world use different values), under which ovulation is considered to not have happened. A study with longitudinal and frequent ultrasound exams would provide a better description of this phenomenon, but such study, which would be very costly given that anovulation are rare events, has not been realized yet to our knowledge. We based our estimates for the duration of anovulatory cycles

based on (*9*) and (*10*). Another observed phenomenon that may be associated with anovulation is when people experience prolonged periods of uterine bleeding. This bleeding may be light or heavy and may be continuous or intermittent but frequent. To reflect this state, we defined the "Anovulatory with bleeding" state. Its sojourn distribution is wide, ranging from height to a hundred days, given that this state is not very well characterized in duration from the existing literature but has been reported by patients to physicians. (*1*)

The states described so far correspond to states in which no conception happens: menses (a period) follow the luteal phase or an anovulatory state. However, when conception happens, *i.e.* when the egg released by the ovaries at ovulation absorbs a sperm (fertilization) and travels down one of fallopian tubes to implants in the uterine wall, another set of successive events happen. The 7-8 days following fertilization (ovulation) are very similar to a luteal phase in which no conception happens. However, once the fertilized egg is implanted, it initiates the production of the HCG hormone which can be detected in urine by pregnancy tests and the production of additional progesterone which leads to an increase or sustained plateau of high temperature. While temperature and HCG remain high in the first trimester of pregnancies, we divided the first weeks/months of pregnancies into two parts. The first part (the "P" state) is a fixed duration (17 days) state in which temperature are high and pregnancy tests likely give positive results. Usually, once pregnancy has been confirmed by a blood test in a clinic, users are less likely to keep tracking their temperature or to report pregnacy test results. This difference in behavior and the fact that early pregnancies can either continue or be interrupted (voluntarily or not) were the primary reasons to define the "P" state as a common first step before a pregnancy ending in a birth or a loss.

If the pregnancy ends in a loss, the model transitions to the state "PL" (pregnancy with loss). This state has a highly skewed sojourn distribution as losses occurring early in pregnancy are more common than late losses.(*11*, *12*) The "L" state follows the "PL" state and is associated to the moment when the loss occurs. Losses often lead to uterine bleeding for a few days which may be reported by the app users. After a loss, individuals usually return to ovulatory cycles. (*13*) Additionally, it may happen, that pregnancy test results remain positive for a few days after the loss, likely a consequence of residual presence of HCG hormone in urine. To account for that, we created an additional state, "lEpL" (for low-Estrogen post-loss) which has similar sojourn distribution and marginal emission distributions than the lE state (see below for the emission distributions), except that positive pregnancy test may be reported at a higher frequency in that state.

If the pregnancy does not end in a loss, then the model progresses through the three trimester of pregnancy. The first two trimesters have a sojourn of fixed duration so that the sojourn distribution of the third trimester embeds the whole variability in length of pregnancies. Indeed, while the average pregnancy duration is of 38 weeks, pre-term births happen in approximately 4-10% of the time depending on countries (*14*) and post-term births may also happen (although less frequently as births are usually induced when pregnancy is past term). Consequently, we describe the sojourn duration of the third trimester of pregnancy as a skew normal distribution with an heavier tail for shorter duration.

Finally, following a birth, the mother may or not breast-feed her newborn child. In the absence of breast-feeding, menses are reported to return 6-8 weeks after delivery, which means that estrogen are rising 4-6 weeks after delivery. The duration of the "post-partum" state (PP), which is associated with the post-partum period when mothers do not breastfeed, is thus described by a normal distribution of mean 5 weeks and standard deviation of 10 days. If the mother breastfeed, this usually delays the return of ovulatory cycles. Given that the breast-feeding duration is highly unpredictable, the sojourn distribution for that state is very flat and ranges from 7 weeks to over two years.

1.0.2 Marginal emission probabilities

The model observations are:

1. **bleeding**, which is reported by users as a categorical variable taking the following possible values: `none`, `spotting`, `light`, `medium`, `heavy`. Bleeding is mostly medium or heavy during menses, losses and births. Spotting may be reported in the early follicular phase, in the luteal phase and around ovulation.(*1*) The "AB" state is also characterized by frequent spotting or bleeding.(*1*)
2. **temperature**, which is a continuous variable. As explained in section 2.2, the temperature reported is reported in Farenheit by users and we center the reported values around the median temperature of that users. In addition, outliers are removed, suspiciously repeated temperatures are removed, and temperatures that users flagged as questionable are also removed. We describe the temperature as following a normal distribution whose mean is state-dependent. During the follicular phase the mean normalized temperature is negative. It is positive after ovulation and in early pregnancies.
3. **mucus** is a categorical variable which can take the following possible values: `none`, `creamy`, `fertile`, `very fertile`, `sticky`. Mucus is absent or creamy in most phases. Fertile mucus is reported when estrogen levels are high and

very fertile mucus is usually reported around ovulation.(*15*) After ovulation, under the influence of progesterone, cervical mucus becomes sticky or absent. What resembles fertile mucus may also be reported throughout pregnancies. Usually, mucus is not reported in states where bleeding is reported.

4. **pregnancy tests** is a categorical variable, that can be either pos (positive test) or neg (negative test). Positive pregnancy tests are reported in pregnancy states or just following a loss (residual HCG hormone in urine). The rest of the time, pregnancy tests results are likely negative.
5. **LH tests** is also a categorical variable, that can be either pos (positive test) or neg (negative test). Positive LH tests are reported on the day of ovulation or the preceding day. Because the α sub-unit of the HCG (pregnancy) hormone is the same as the α sub-unit of the LH hormone, LH tests may also be reported positive in pregnancy states.(*16*) Some conditions, such as PCOS, lead to high levels of LH and thus users with this condition may report positive LH tests throughout their cycle. To account for these users, we initialize the probability of a positive LH test in other states as low and expect this probability to increase when fitting the model to time-series of users that may be affected by this condition (or other conditions leading to high LH concentration).

See figure 1.5 for a visualization of the marginal distributions of these values.

1.0.3 Censoring probabilities

Our hidden semi-Markov model framework allows to model missing data. The probability of missing data can be modeled in each state by either or both of these two ways: first, one can specify the probability of all variable being missing in a given state, and second, one can specify the probability of a specific variable being reported in a given state.

In our case, the first probability can be interpreted as the probability that the user opens the app when in a given biological state while the second probability can be interpreted as the probability that the user measured/observed a given body-sign in a given state.

While we don't know what the tracking behavior of each user is, we will here specify them as the expected behavior for someone who is using the app for their family planning and who is interested in identifying their fertile window and pregnancies so that we can simulate realistic data in the next section. However, when decoding the time-series of a new users for which we don't have any tracking behavior information, we set these probabilities to have the same values accross states. When fitting the model to the time-series, the tracking behavior will be learned, just as the emission probabilities are updated.

```
## Warning: `guides(<scale> = FALSE)` is deprecated. Please use `guides(<scale> =
## "none")` instead.
```

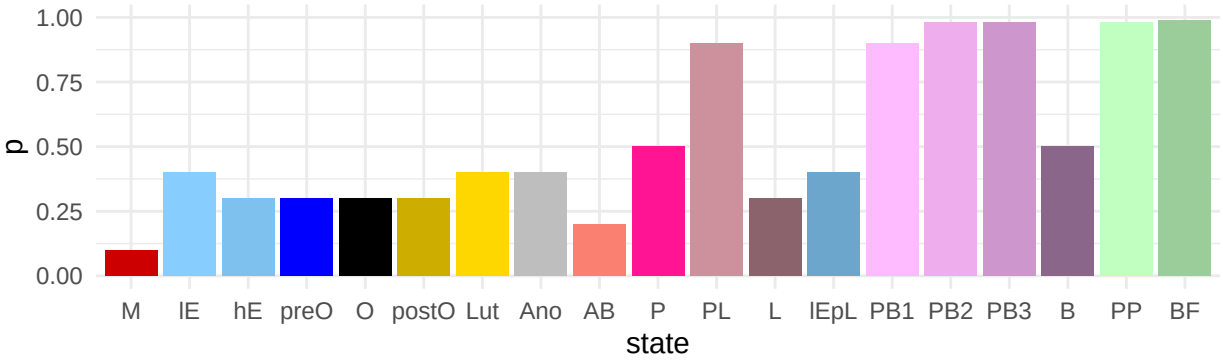


Figure 1.1: Probability of all variable beeing missing simultaneously in each state of our reproductive events HSMM for a hypothetical 'typical' user.

```
## Warning: `guides(<scale> = FALSE)` is deprecated. Please use `guides(<scale> =
## "none")` instead.
```

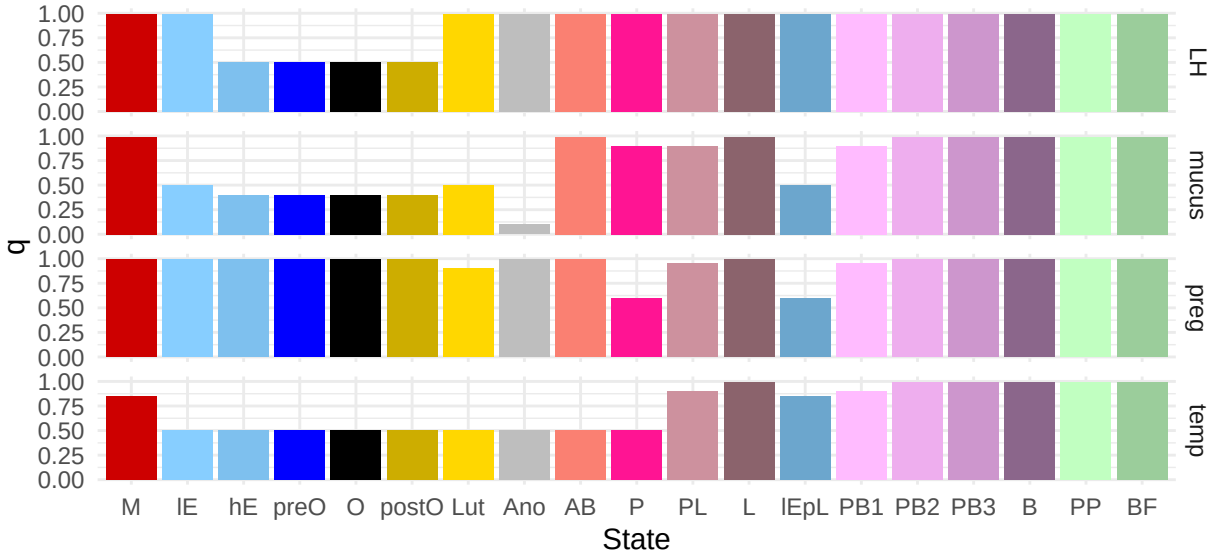


Figure 1.2: Probabilities that a hypothetical 'typical' user would not report a specific variable in each state of our reproductive events HSMM.

1.0.4 Specifying the `hsmm`

```
R_hsmm = specify_hsmm(
  J = R_model$n_states,
  state_names = R_model$states$abbr,
  state_colors = R_model$state$colors,
  init = R_model$init,
  transition = R_model$trans,
  sojourn = list(
    type = "nonparametric",
    d = t(R_model$sojourn)
  ),
  marg_em_probs = R_model$emission_par,
  censoring_probs = R_model$censoring_probs,
  verbose = FALSE
)
```

```
## Warning in .check_sojourn_all_states(sojourn, J): The provided sojourn
## distributions do not sum to 1 for at least one state. Sojourn distributions are
## normalized so that they sum to 1.
```

```
plot_hsmm_transitions(model = R_hsmm)
```

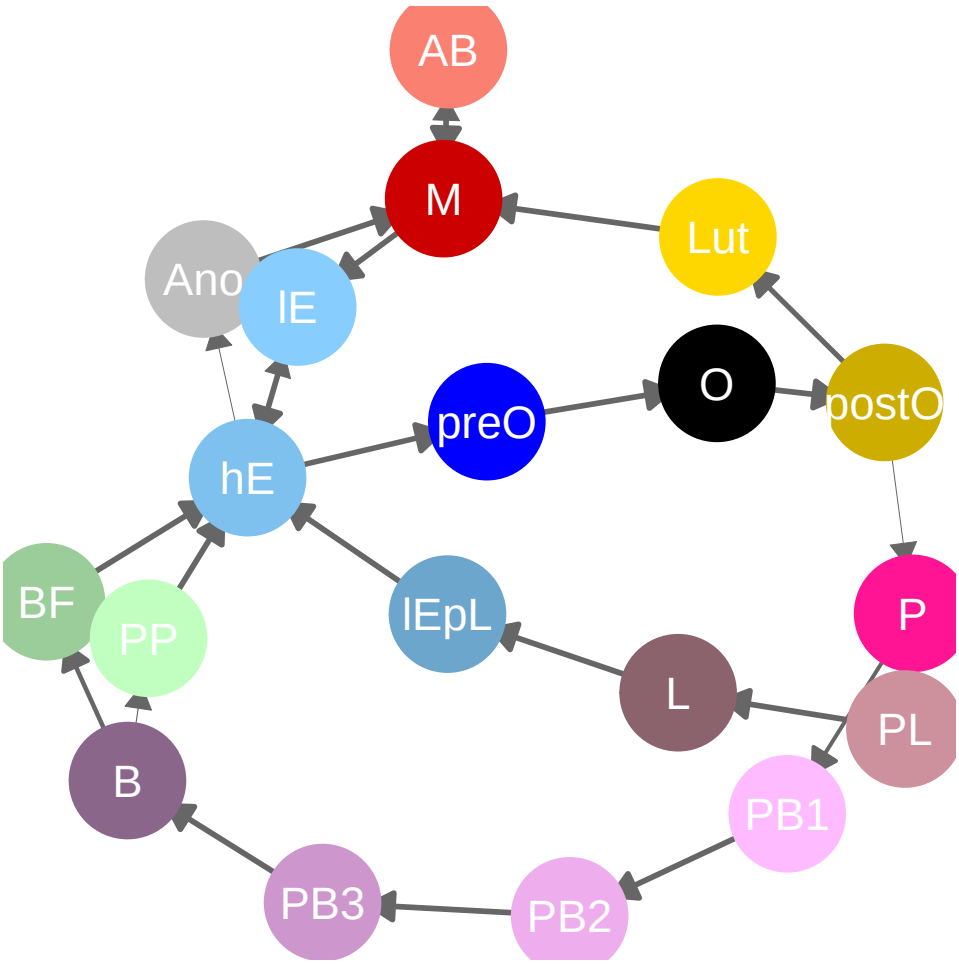


Figure 1.3: Graph of our reproductive events HSMM. The width of the edges are proportional to the transition probabilities between states.

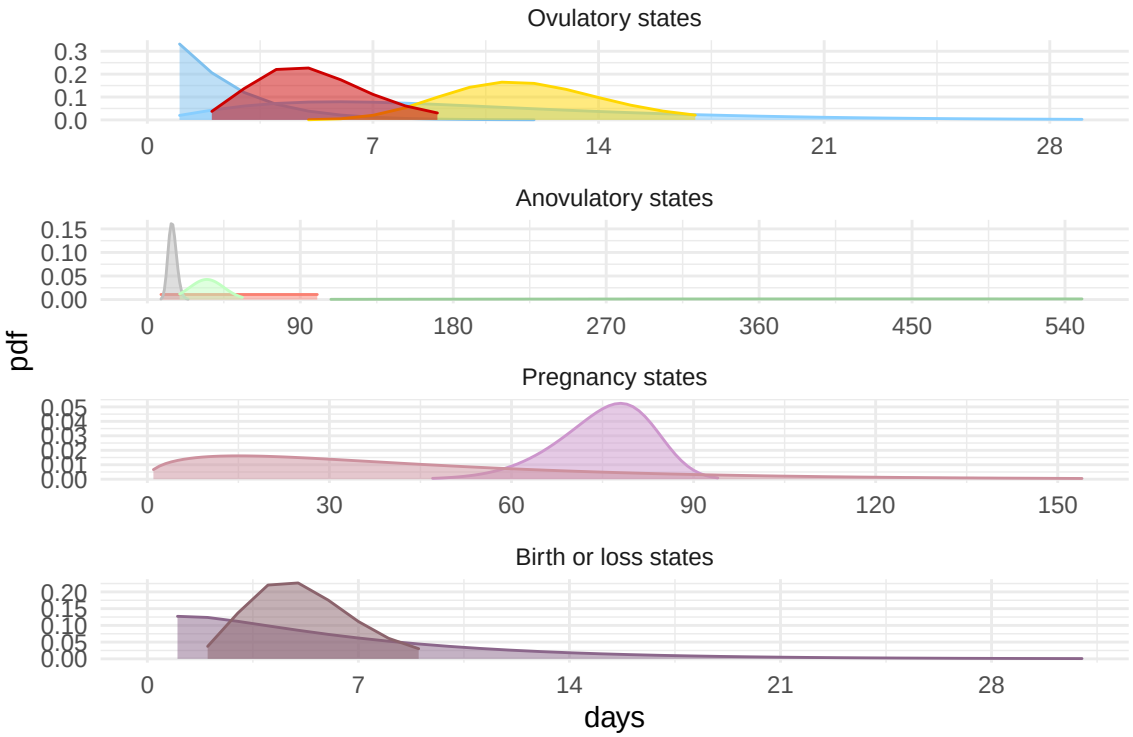


Figure 1.4: Sojourn distributions of the HSMM of reproductive events.

```
plot_hsmm_marg_dist(model = R_hsmm, show_missing_probs = TRUE)
```

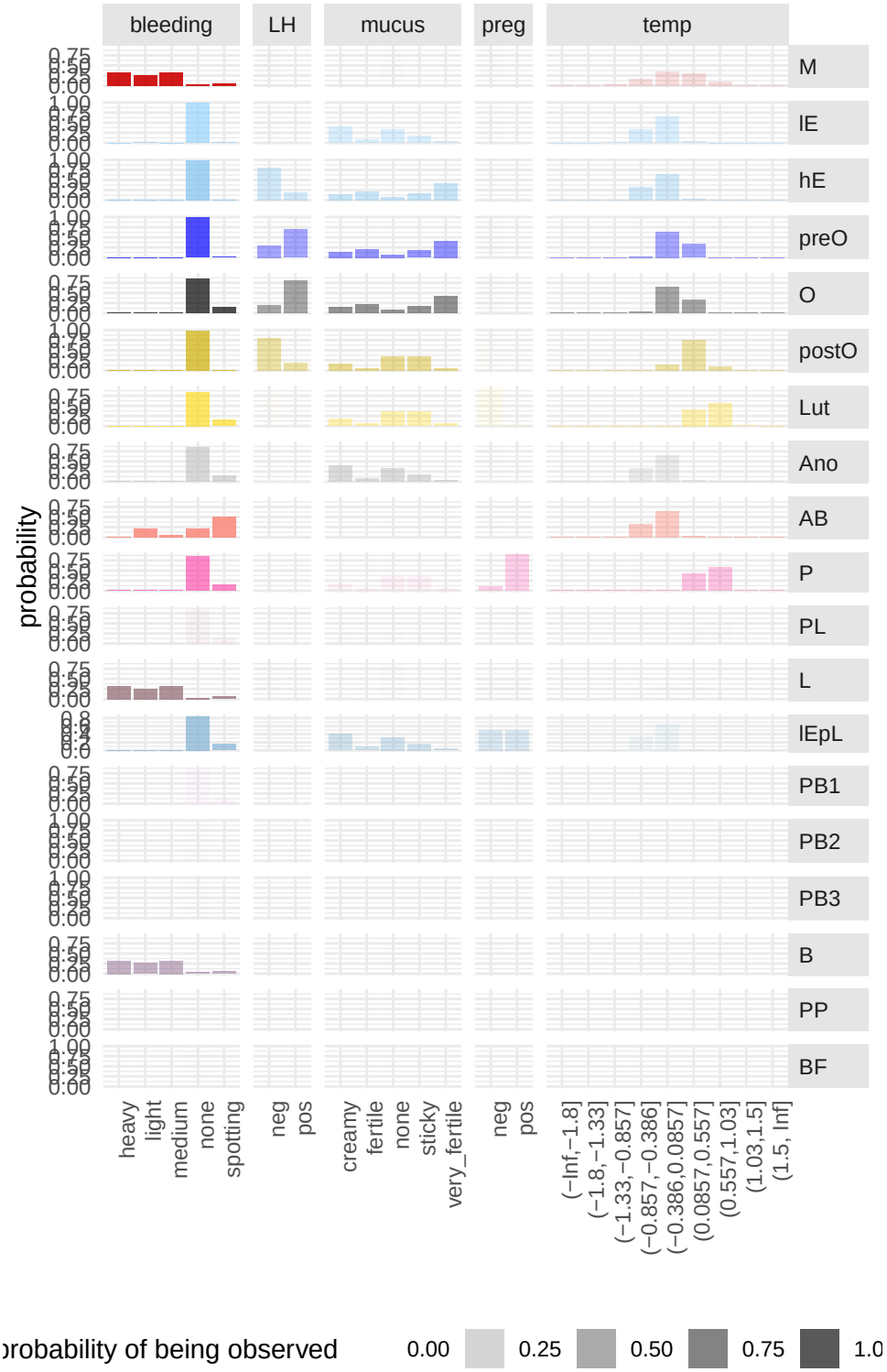


Figure 1.5: Specified marginal emission probabilities of our reproductive events HSMM.

1.0.5 Simulate a sequence

```
X = simulate_hsmm(R_hsmm, seed = 22, n_state_transitions = 150)

plot_hsmm_seq(X, model = R_hsmm)
```

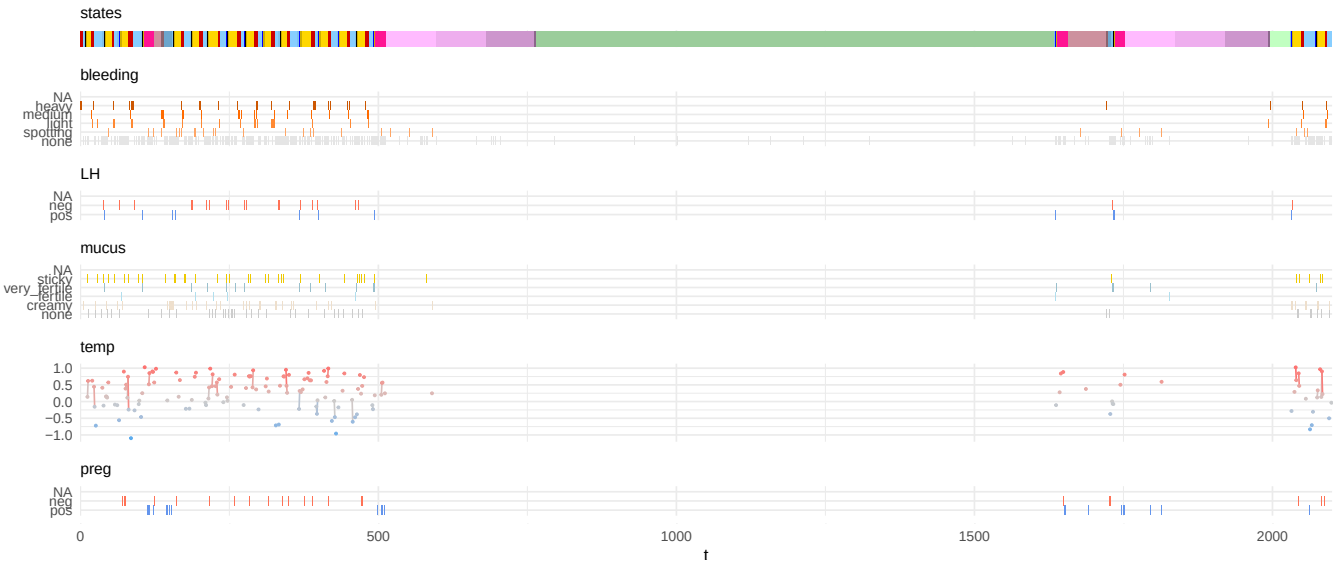


Figure 1.6: Simulated state and observation sequence with the reproductive event HSMM.

1.0.6 Save model

```
save(R_model, file = paste0("../Data/models/R_model.Rdata"))
save(R_hsmm, file = paste0("../Data/models/R_hsmm.Rdata"))
```

1.0.7 Specifying the `hmm`

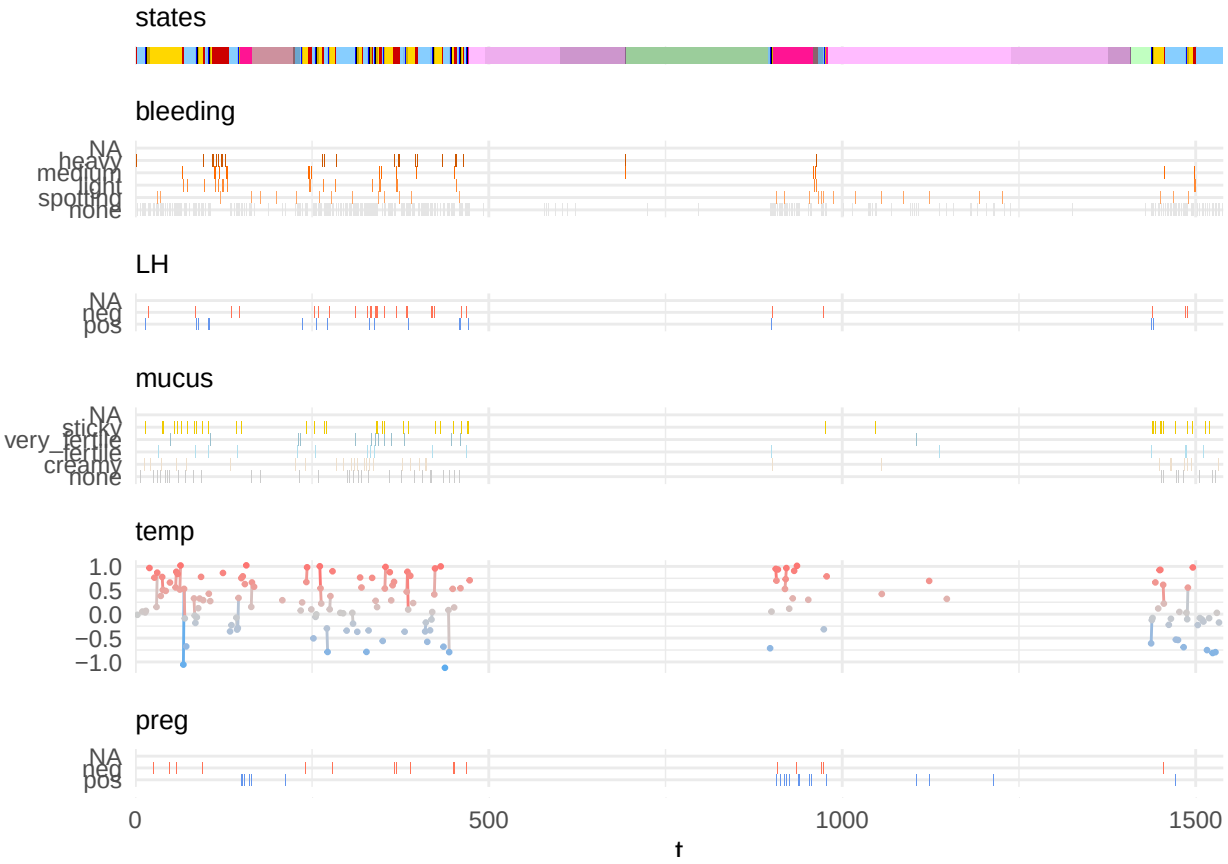
For the sake of comparing the performances of our HSMM with a simple HMM, we also define a HMM by setting the sojourn distributions as geometric. The probability parameter is estimated by fitting the sojourn distribution of the HSMM.

```
# geometric sojourns
# we fit them from randomly generating from the hsmm sojourns

S = R_model$sojourn

geom_probs = map_dbl(
  .x = 1:nrow(S),
  .f = function(j){
    d = R_model$sojourn[j,]
    X = rep((1:ncol(S))-1, times = round(d*10000))
    length(X)/(length(X) + sum(X))
  }
)
```

```
R_hmm = specify_hsmm(
  J = R_model$n_states,
  state_names = R_model$states$abbr,
  state_colors = R_model$state$colors,
  init = R_model$init,
  transition = R_model$trans,
  sojourn = list(
    type = "geometric",
    prob = geom_probs
  ),
  marg_em_probs = R_model$emission_par,
  censoring_probs = R_model$censoring_probs,
```

```
  verbose = FALSE
)
```

```
X_hmm = simulate_hsmm(R_hmm, seed = 22, n_state_transitions = 150)
plot_hsmm_seq(X_hmm, model = R_hmm)
```



Simulating a sequence with the HMM shows quite unrealistic patterns in terms of pregnancy duration or menses duration (among the most obvious unrealistic patterns).

```
save(R_hmm, file = paste0("../Data/models/R_hmm.Rdata"))
```

1.0.8 Specifying a hsmm with weak sojourn priors

Similarly, to compare the performances of our HSMM, we weaken the sojourn distribution priors.

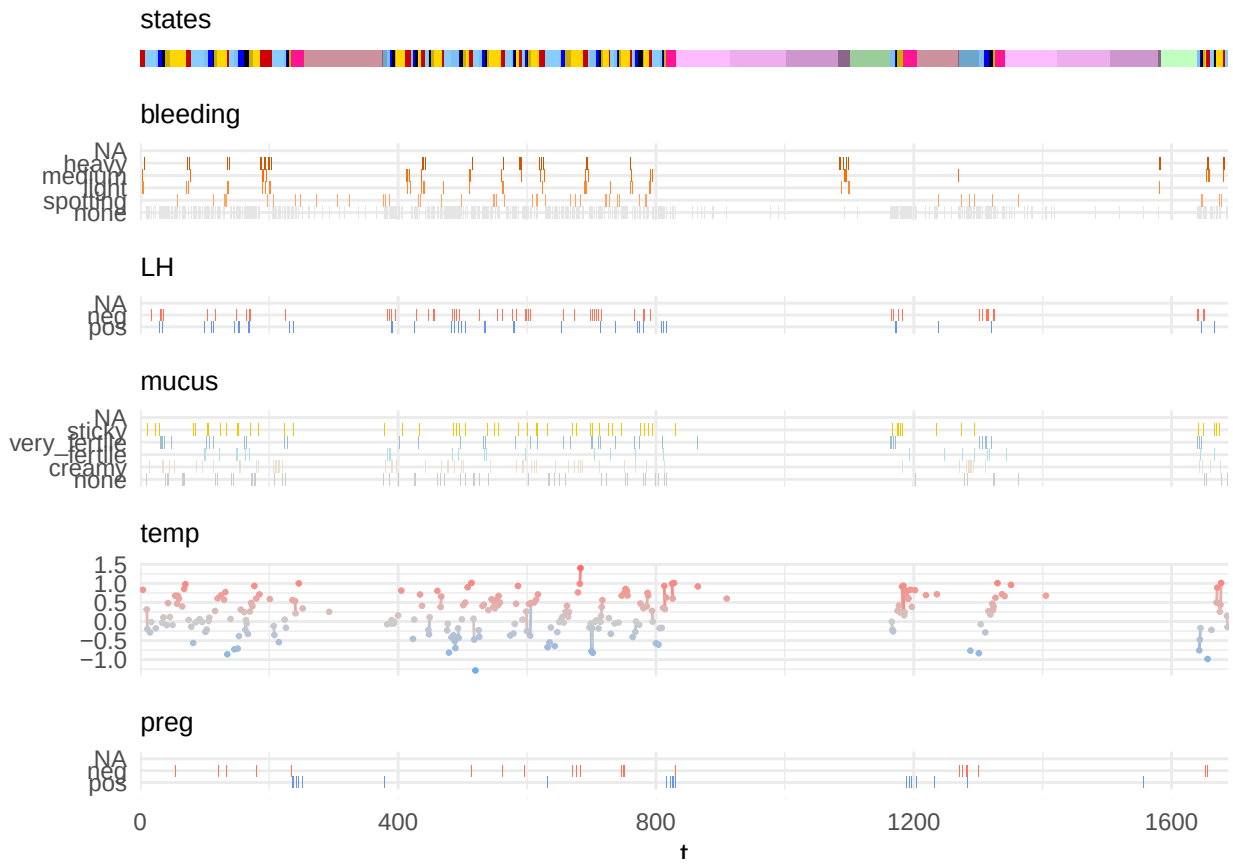
The sojourn distributions are altered such that the distributions are centered approximately around the same mean duration but have wider variability.

```
sojourns = R_hsmm$sojourn

for(j in 1:J){
  this_d_o = sojourns[[j]]$d
  rsamp = sample(1:length(this_d_o), prob = this_d_o, size = 10000, replace = TRUE)
  v = (sd(rsamp)+0.1) %>% ceiling() %>%  add(3)
  this_d_f = dnorm(-round(3*v):round(3*v), mean = 0, sd = v)
  V = (length(this_d_f)/2) %>% floor
  this_d = stats::filter(c(rep(0,V),this_d_o, rep(0,V)), filter = this_d_f, sides = 2)
  this_d = this_d[!is.na(this_d)]
  this_d = this_d/sum(this_d)
  # plot(this_d_o[1:30], ylim = range(c(this_d_o, this_d)))
  # points(this_d[1:30], col = "red")
  sojourns[[j]]$d = this_d
```

```
}
```

```
R_weak_hsmm = specify_hsmm(
  J = R_model$n_states,
  state_names = R_model$states$abbr,
  state_colors = R_model$state$colors,
  init = R_model$init,
  transition = R_model$trans,
  sojourn = sojourns,
  marg_em_probs = R_model$emission_par,
  censoring_probs = R_model$censoring_probs,
  verbose = FALSE
)
```

```
X_weak_hsmm = simulate_hsmm(R_weak_hsmm, seed = 22, n_state_transitions = 150)
plot_hsmm_seq(X_weak_hsmm, model = R_weak_hsmm)
```




```
save(R_weak_hsmm, file = paste0("../Data/models/R_weak_hsmm.Rdata"))
```

2 Datasets

2.1 Synthetic data generation

To generate synthetic data for N fictive users, we first define some key characteristics of these users. First, their highest tracking frequency is defined via a parameter α. Then, we randomly sample their menstrual characteristics such as the length and regularity of their follicular and luteal phases or the noise in their temperature measurements from realistic priors.

2.1.1 Simulating realistic tracking behavior

Because we want to evaluate the performance of our framework given different level of missing data in a realistic context, we first define censoring probabilities for each state and variable that reflect the tracking behavior of a dedicated user whose purpose is to achieve or avoid pregnancy. A dedicated user would not track all the features every day but would instead systematically report the relevant features at the relevant moments of their cycle. For example, they would report ovulation test in the days leading and following ovulation, not during their menses.

Once these censoring probabilities are defined, we can modulate them with a parameter α reflecting that some users are less dedicated than others.

We specify realistic censoring probabilities of a dedicated user as follow:

```
## Loading objects:
##   R_hsmm
```

Table 2.1: Expected censoring probabilities p.

M	lE	hE	preO	O	postO	Lut	Ano	AB	P	PL	L	lEpL	PB1	PB2	PB3	B	PP	BF
0.01	0.05	0.025	0.025	0.025	0.025	0.05	0.05	0.05	0.05	0.9	0.05	0.05	0.9	0.98	0.98	0.2	0.9	0.98

Table 2.2: Expected marginal censoring probabilities q.

	M	lE	hE	preO	O	postO	Lut	Ano	AB	P	PL	L	lEpL	PB1	PB2	PB3	B	PP	BF
bleeding	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
LH	0.99	0.99	0.30	0.10	0.10	0.10	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99
mucus	0.99	0.50	0.10	0.10	0.10	0.10	0.50	0.10	0.99	0.90	0.90	0.99	0.50	0.90	0.99	0.99	0.99	0.99	0.99
temp	0.85	0.50	0.10	0.10	0.10	0.10	0.30	0.10	0.50	0.30	0.75	0.99	0.85	0.75	0.99	0.99	0.99	0.99	0.99
preg	0.99	0.99	0.99	0.99	0.99	0.99	0.90	0.99	0.99	0.60	0.95	0.99	0.60	0.95	0.99	0.99	0.99	0.99	0.99

We now modulate these probabilities with the parameter α:

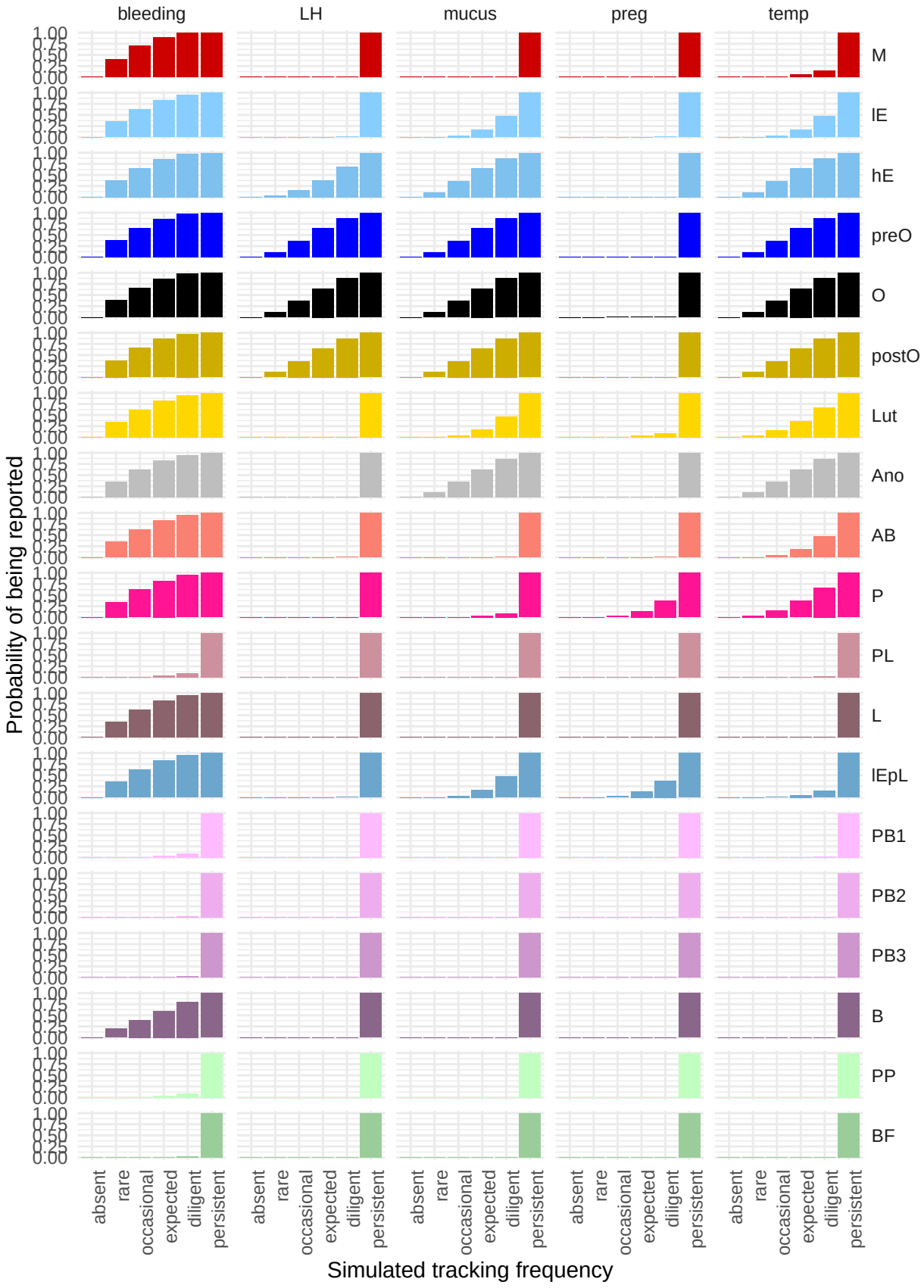


Figure 2.1: Probabilities for a variable to be reported given different tracking behaviors.

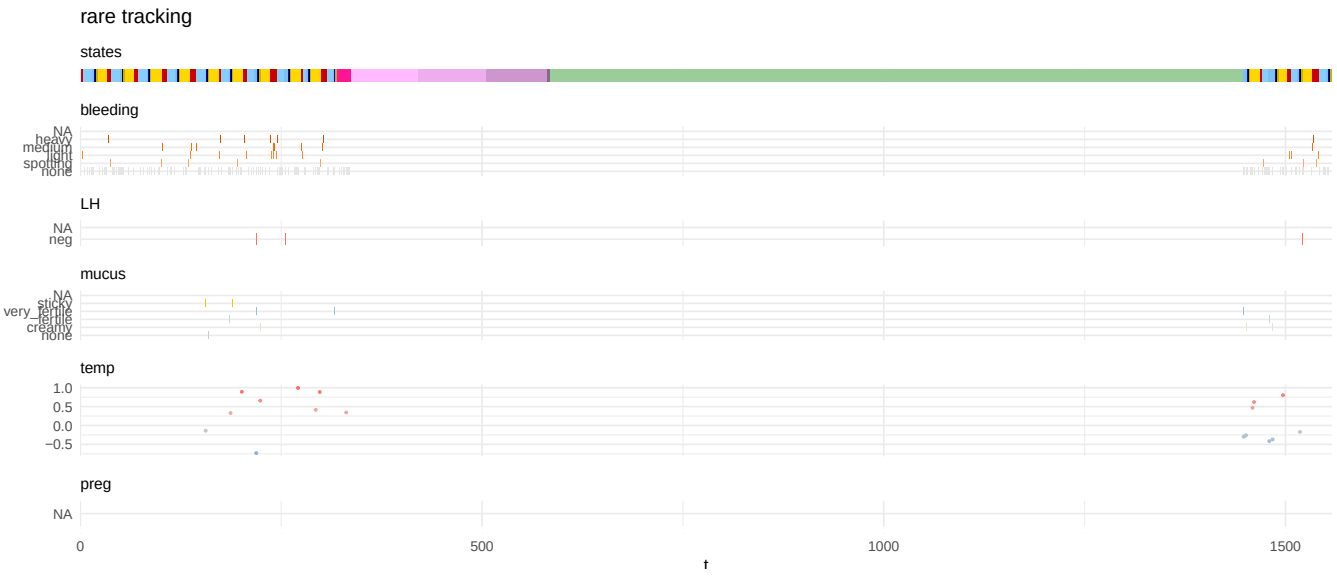
Generating time-series

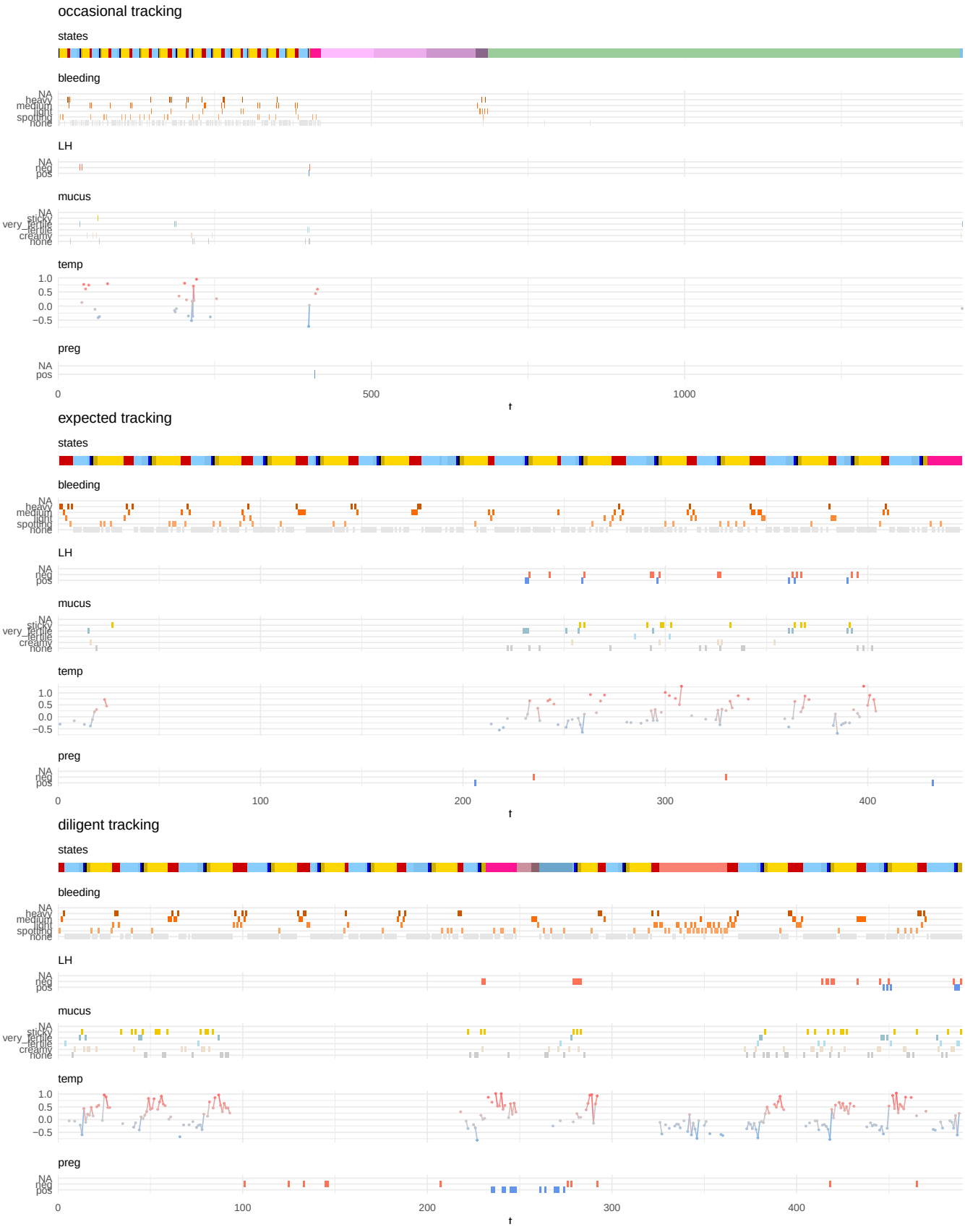
```
Xsim_file = "../Data/synthetic_data/Xsim.feather"
if(!file.exists(Xsim_file)){
  set.seed(0)
  s_users =
    simulate_individual_characteristics(
      N_per_alpha = 50,
      missing_probs = missing_probs %>%  filter(alpha != 0)
    )
  Xsim = generate_time_series(s_users, missing_probs, model = R_hsmm)
  Xsim = modify_set_of_tracked_variables(Xsim)
  write_feather(Xsim, Xsim_file)
}else{
  Xsim = read_feather(path = Xsim_file)
}
```

Visualization of synthetic time-series examples

```
examples =
  Xsim %>%
  select(seq_id, tracking_category, alpha_level) %>%
  group_by(seq_id, alpha_level) %>%
  summarize(has_full_tracking = sum(tracking_category == "full") >= 100, .groups = "drop") %>%
  filter(has_full_tracking) %>%
  group_by(alpha_level) %>%
  slice_head(n = 1)
```

```
for(i in 1:nrow(examples)){
  ex_id = examples$seq_id[i]
  plot_hsmm_seq(X = Xsim %>%  filter(seq_id == ex_id), model = R_hsmm,
                title = str_c(examples$alpha_level[i], " tracking")) %>%
    print()
}
```




occasional tracking
states
bleeding
NA
heavy
medium
light
spotting
none
LH
NA
neg
pos
mucus
NA
sticky
very_fertile
fertile
creamy
none
temp
1.0
0.5
0.0
−0.5
preg
NA
pos
0
500
1000
t
expected tracking
states
bleeding
NA
heavy
medium
light
spotting
none
LH
NA
neg
pos
mucus
NA
sticky
very_fertile
fertile
creamy
none
temp
1.0
0.5
0.0
−0.5
preg
NA
neg
pos
0
100
200
300
400
t
diligent tracking
states
bleeding
NA
heavy
medium
light
spotting
none
LH
NA
neg
pos
mucus
NA
sticky
very_fertile
fertile
creamy
none
temp
1.0
0.5
0.0
−0.5
preg
NA
neg
pos
0
100
200
300
400
t

Synthetic dataset characteristics

```
table(tracking_category = Xsim$tracking_category)  %>%
  kable(., format = "latex", booktabs = T,
        caption = "Number of data-point in each tracking category.") %>%
  kable_styling(latex_options = c("hold_position"))
```

Table 2.3: Number of data-point in each tracking category.

tracking_category	Freq
b	65288
bp	63876
btm	68836
full	72147

```
table(seq_id = Xsim$seq_id) %>%
  data.frame() %>%
  set_colnames(c("seq_id","sequence_length")) %>%
  ggplot(., aes(x = sequence_length)) +
  geom_histogram(bins = 50) + expand_limits(x = 0)
```

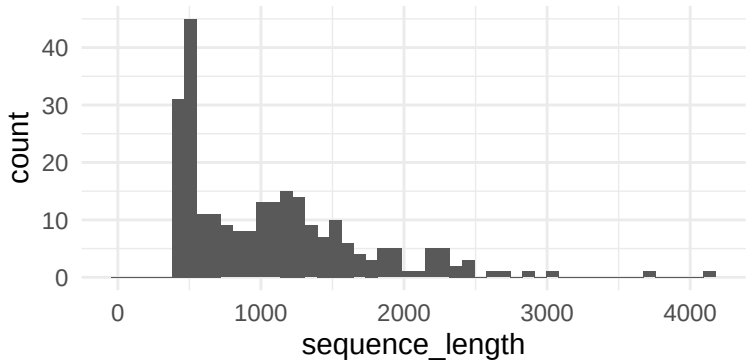


Figure 2.2: Histogram of simulated sequence length.

```
tracking_freq_table_synth_data =
  Xsim %>%
  filter(tracking_category == "full") %>%
  group_by(alpha_level) %>%
```

Table 2.4: Effective tracking frequency (fraction of reported variables) and frequency of opening the app (at least one variable is logged at a given time-point) per simulated tracking behaviors.

Tracking_category	Effective_tracking_frequency	Frequency_of_opening_the_app
rare	0.03	0.14
occasional	0.06	0.22
expected	0.16	0.44
diligent	0.18	0.40
persistent	1.00	1.00

```
  mutate(any_log = !is.na(bleeding) | !is.na(LH) | !is.na(preg) | !is.na(mucus) | !is.na(temp),
         any_log = ifelse(!any_log,NA, 1)) %>%
  summarize(across(c(bleeding, LH, preg, mucus, temp, any_log), .fns = function(x) sum(!is.na(x))),
            n = n(),
            .groups = "drop") %>%
  mutate(Effective_tracking_frequency = (bleeding+LH+preg+mucus+temp)/5/n,
         Frequency_of_opening_the_app = any_log/n) %>%
  select(alpha_level, Effective_tracking_frequency, Frequency_of_opening_the_app ) %>%
  arrange(alpha_level)

tracking_freq_table_synth_data %>%
  rename(Tracking_category = alpha_level) %>%
  kable(.,
        booktab = T,
        digits = 2,
        caption = "Effective tracking frequency (fraction of reported variables) and frequency of opening the
```

2.2 Kindara app data pre-processing and partial manual labeling.

pre-processing function

```
source("Scripts/00_function_prepare_obs.R")
```

pre-processing

```
days = read_feather(path = paste0(IO$tmp_data, "days_selected_users.feather"))

X = purrr::map_dfr(
  .x = unique(days$user_id),
  .f = function(uid) prepare_obs(d = days %>% filter(user_id == uid))
) %>%
  rename(seq_id = user_id, t = rel_date)

write_feather(X, path = paste0(IO$output_data, "processed_app_data.feather"))
```

manual labeling

```
ML = read_feather(path = paste0(IO$output_data, "ML.feather"))
```

```
load("../Data/models/R_hsmm.Rdata", verbose = TRUE)

ML = label_sequences(model = R_hsmm, X = X, ground_truth = ML)

write_feather(ML, path = paste0(IO$output_data, "ML.feather"))
```

```
breaks_tracking_freq = c(0, tracking_freq_table_synth_data$Effective_tracking_frequency)
breaks_tracking_freq = (breaks_tracking_freq + lag(breaks_tracking_freq))/2
breaks_tracking_freq = c(0, breaks_tracking_freq[-1],1)


breaks_app_open = c(0, tracking_freq_table_synth_data$Frequency_of_opening_the_app)
breaks_app_open = (breaks_app_open + lag(breaks_app_open))/2
breaks_app_open = c(0, breaks_app_open[-1],1)


users_tracking_freq_cat =
  X %>%
  group_by(seq_id) %>%
  mutate(any_log = !is.na(bleeding) | !is.na(LH) | !is.na(preg) | !is.na(mucus) | !is.na(temp),
         any_log = ifelse(!any_log,NA, 1)) %>%
  summarize(across(c(bleeding, LH, preg, mucus, temp, any_log), .fns = function(x) sum(!is.na(x))),
            n = n(),
            .groups = "drop") %>%
  mutate(Effective_tracking_frequency = (bleeding+LH+preg+mucus+temp)/5/n,
         Frequency_of_opening_the_app = any_log/n,
         tracking_freq_cat = Effective_tracking_frequency %>%
           cut(breaks = breaks_tracking_freq,
               labels = c("none",tracking_freq_table_synth_data$alpha_level %>% as.character())),
         open_app_cat = Frequency_of_opening_the_app %>%
           cut(breaks = breaks_app_open,
               labels = c("none",tracking_freq_table_synth_data$alpha_level %>% as.character()))
  )


users_tracking_freq_cat %>%
  group_by(tracking_freq_cat) %>%
  summarize(n = n(),
            perc = n/length(unique(X$seq_id)) * 100,
```

Table 2.5: Number and percentage of users in our real-world dataset according to the tracking categories defined for our simulated data, based on the frequency with which they log features.

tracking_freq_cat	n	perc
rare	12	11
occasional	33	29
expected	13	11
diligent	55	48
persistent	1	1

Table 2.6: Number and percentage of users in our real-world dataset according to the tracking categories defined for our simulated data, based on the frequency with which they open the app.

open_app_cat	n	perc
rare	4	4
occasional	29	25
expected	14	12
diligent	27	24
persistent	40	35

```
            .groups = "drop") %>%
  kable(.,
        booktab = T,
        digits = 0,
        caption = "Number and percentage of users in our real-world dataset according to the tracking categor
```

```
users_tracking_freq_cat %>%
  group_by(open_app_cat) %>%
  summarize(n = n(),
            perc = n/length(unique(X$seq_id)) * 100,
            .groups = "drop") %>%
  kable(.,
        booktab = T,
        digits = 0,
        caption = "Number and percentage of users in our real-world dataset according to the tracking categor
```

3 Decoding time-series

3.1 Decoding functions

Functions are implemented in the file `Scripts/00_FAM_decoding_functions.R` which is available on the same github repository as this file.

3.2 Loading models

```
load("../Data/models/R_hsmm.Rdata", verbose = TRUE)
load("../Data/models/R_hmm.Rdata", verbose = TRUE)
load("../Data/models/R_weak_hsmm.Rdata", verbose = TRUE)
```

3.3 Synthetic data

```
X = read_feather("../Data/synthetic_data/Xsim.feather")

RES.file = "../Data/decodings/RES_synthetic_data.feather"

if(rerun_from_here | !file.exists(RES.file)){

  RES = purrr::map_dfr(.x = unique(X$seq_id),
                       .f = function(sid){
                         # cat(sid, "\n")  # prints the subject IDs
                         get_most_likely_sequence_with_prob(
                           X = X %>%
                             filter(seq_id == sid) %>%
                             select(-alpha, -alpha_level,
                                    -state, -tracking_category,
                                    -temp_sd,
                                    -mean_lE_sojourn, -sd_lE_sojourn,
                                    -mean_Lut_sojourn, -sd_Lut_sojourn),
                           verbose = FALSE,
                           model = R_hsmm
                         )
                       }
  )

  write_feather(RES, path = RES.file)
}
```

3.4 App data

We decode the app data using several approaches to compare their accuracy. We use our proposed approach to adapt for tracking behavior with the proposed HSMM as well as with a simple HMM and a HSMM with broader sojourn distributions.

We also decode the app data with the proposed HSMM, without adjusting for tracking behavior.

```
models =
  expand_grid(
    tibble(model = c("R_hmm","R_weak_hsmm","R_hsmm", "R_hsmm"),
           approach = c("adaptative","adaptative","adaptative","non-adaptative")),
    fit_model = c(FALSE, TRUE)) %>%
  mutate(
    file_name =
```

```
      paste0(approach, "_", model, "_", ifelse(fit_model, "fitted","specified"), ".feather")
  )

write_feather(models, path = paste0(IO$output_data, "models.feather"))
```

```
X = read_feather(path = paste0(IO$output_data, "processed_app_data.feather"))
ML = read_feather(path = paste0(IO$output_data, "ML.feather"))
X_all_users = X %>% filter(seq_id %in% unique(ML$seq_id))
```

```
output_dir = paste0(IO$output_data,"decodings/")
if (!dir.exists(output_dir)) dir.create(output_dir)

exec_times =
  purrr::map_dfr(
    .x = 1:nrow(models),
    .f = decode_with_model,
    verbose = TRUE
  )
```

```
## 1
## 2
## 3
## 4
## 5
## 6
## 7
## 8
```

```
# X = X_all_users %>%  filter(seq_id == sid)
```

4 Performances

4.1 On Synthetic data

Loading and formatting results

```
X = read_feather("../Data/synthetic_data/Xsim.feather")
RES = read_feather("../Data/decodings/RES_synthetic_data.feather")
```

```
XP = full_join(RES,
               X %>%  select(seq_id, t, state, tracking_category, alpha, alpha_level) %>%
                 rename(state_GT = state), by = c("seq_id", "t"))
```

Overall accuracy

```
XP =
  XP %>%
  mutate(
    tracking_category_str =
      case_when(tracking_category == "b" ~ "bleeding only (B)",
                tracking_category == "bp" ~ "bleeding & preg. tests (BP)",
                tracking_category == "btm" ~ "bleeding, mucus & tº (BTM)",
                tracking_category == "full" ~ "all"
      ) %>%
      factor(., levels = c("bleeding only (B)","bleeding & preg. tests (BP)","bleeding, mucus & tº (BTM)","al
  )
```

```
Accuracies =
  XP %>%
  group_by(alpha, alpha_level, tracking_category_str) %>%
  summarize(Accuracy = mean(state == state_GT, na.rm = TRUE),
            Weighted_mean_of_sample_accuracy = weighted.mean(x = state == state_GT, w = prob, na.rm = TRUE),
            .groups = "drop") %>%
  pivot_longer(cols = c("Accuracy","Weighted_mean_of_sample_accuracy"),
               names_to = "accuracy_type", values_to = "Accuracy") %>%
  mutate(accuracy_type = accuracy_type %>%  str_replace_all(., "_"," "))
```

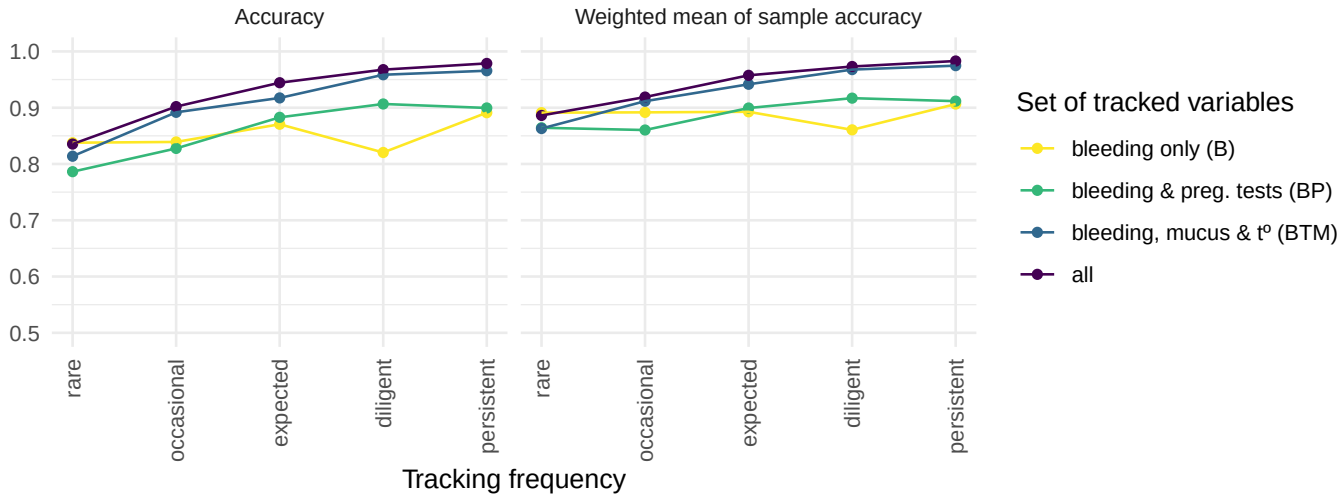


Figure 4.1: Accuracy and weighted mean of sample accuracy on synthetic data.

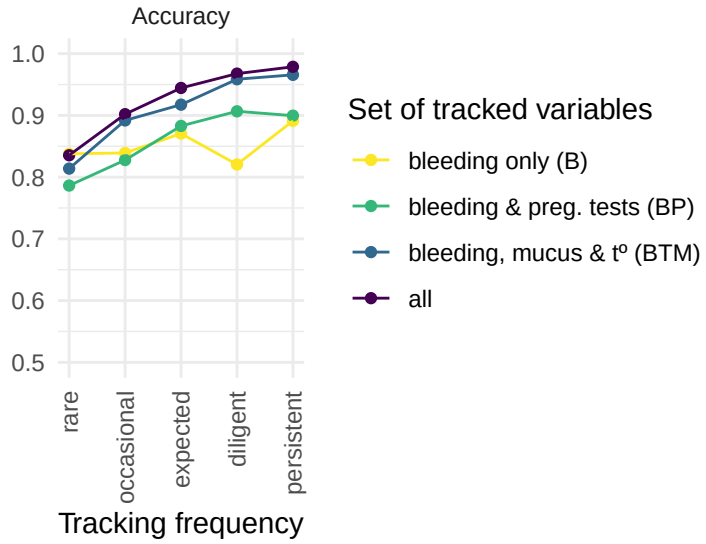


Figure 4.2: Accuracy on synthetic data.

State-specific accuracy

```
Accuracies_per_state = XP %>%
  group_by(alpha, alpha_level, tracking_category_str, state_GT) %>%
  summarize(Accuracy = mean(state == state_GT, na.rm = TRUE),
            Weighted_mean_of_sample_accuracy = weighted.mean(x = state == state_GT, w = prob, na.rm = TRUE),
            .groups = "drop") %>%
  mutate(state_name = R_hsmm$state_names[state_GT] %>% factor(., levels = R_hsmm$state_names),
         state_col = R_hsmm$state_colors[state_GT]) %>%
  pivot_longer(cols = c("Accuracy","Weighted_mean_of_sample_accuracy"),
               names_to = "accuracy_type", values_to = "accuracy") %>%
  mutate(accuracy_type = accuracy_type %>%  str_replace_all(., "_"," "))
```

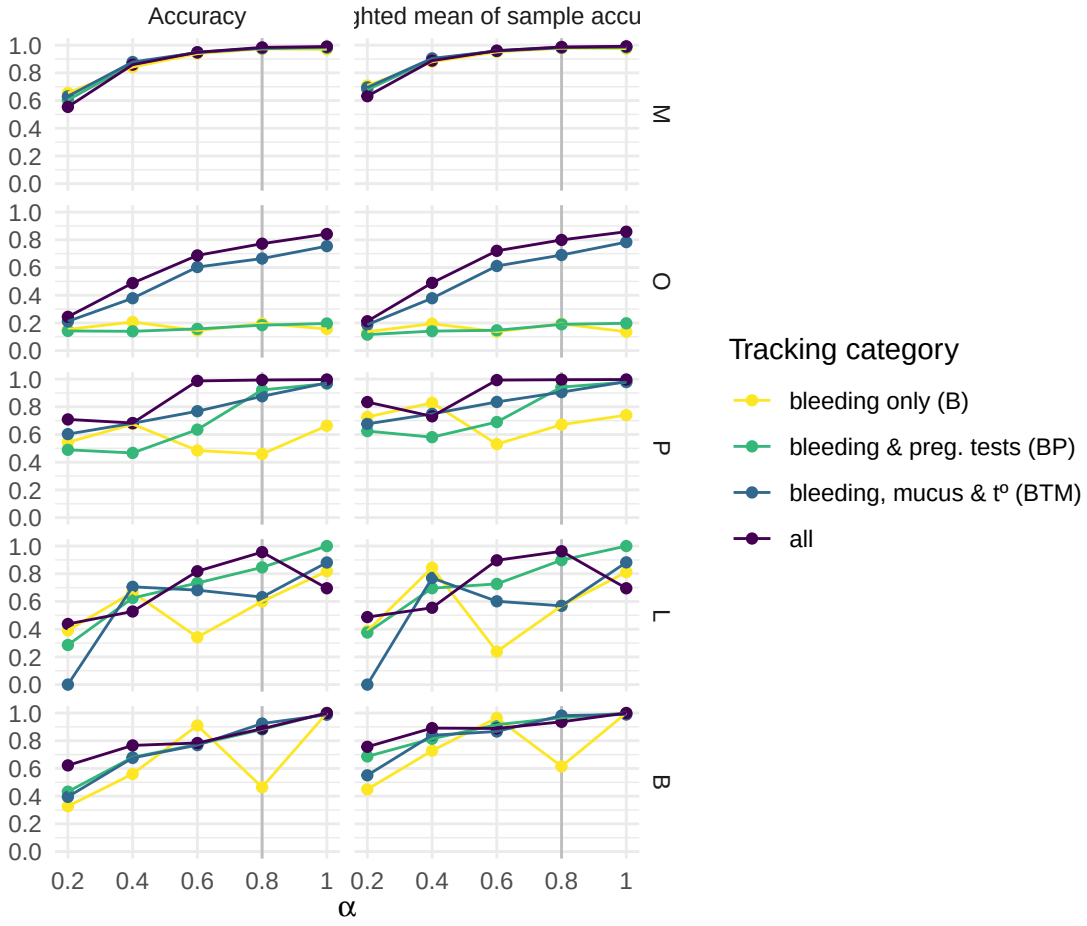


Figure 4.3: Per state accuracy on synthetic data.

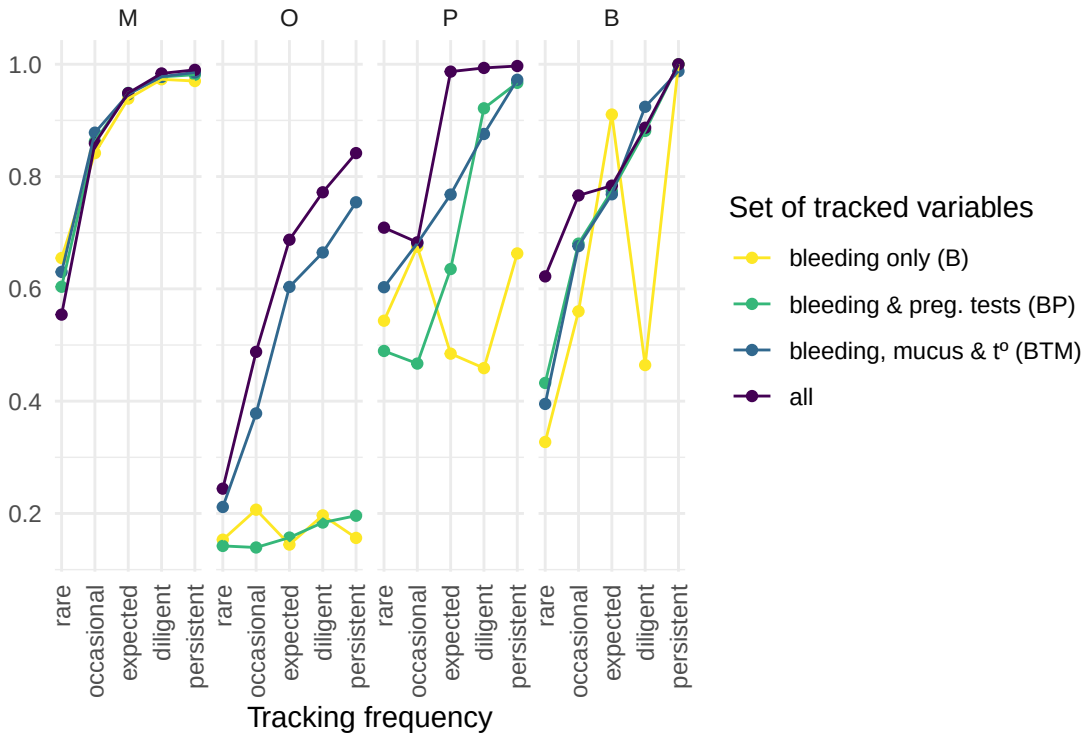


Figure 4.4: Per state accuracy on synthetic data.

Confusion matrices

```
confusion_matrix_df = XP %>%
  filter(!is.na(state)) %>%
  group_by(alpha, alpha_level, tracking_category, state_GT, state) %>%
  summarize(n = n(),
            wn = sum(prob),
            .groups = "drop") %>%
  group_by(alpha, alpha_level, tracking_category, state_GT) %>%
  mutate(tot = sum(n),
         wtot = sum(wn),
         .groups = "drop") %>%
  ungroup() %>%
  mutate(perc = n/tot,
         wperc = wn/wtot) %>%
  select(-n, -wn, -tot, -wtot) %>%
  pivot_longer(cols = c("perc","wperc"), names_to = "type", values_to = "fraction") %>%
  mutate(type = ifelse(type == "perc", "Accuracy","Weighted Accuracy"),
         GT_state_name = R_hsmm$state_names[state_GT] %>%  factor(., levels = R_hsmm$state_names),
         decoded_state_name = R_hsmm$state_names[state] %>%  factor(., levels = R_hsmm$state_names))
```

```
g_synth_conf_mat =
  ggplot(confusion_matrix_df %>%  filter(type == "Accuracy"),
         aes(x = decoded_state_name, y = GT_state_name, fill = fraction))+
  geom_tile()+
  scale_y_discrete(drop = FALSE) +
  facet_grid(tracking_category ~ alpha_level) +
  scale_fill_gradient(low = "white", high = "steelblue4", limits = c(0,1)) +
  coord_fixed() +
  ylab("Simulated states (ground truth)") + xlab("Decoded states") +
  theme(axis.text.x = element_text(angle = 90, hjust = 1))

g_synth_conf_mat
```

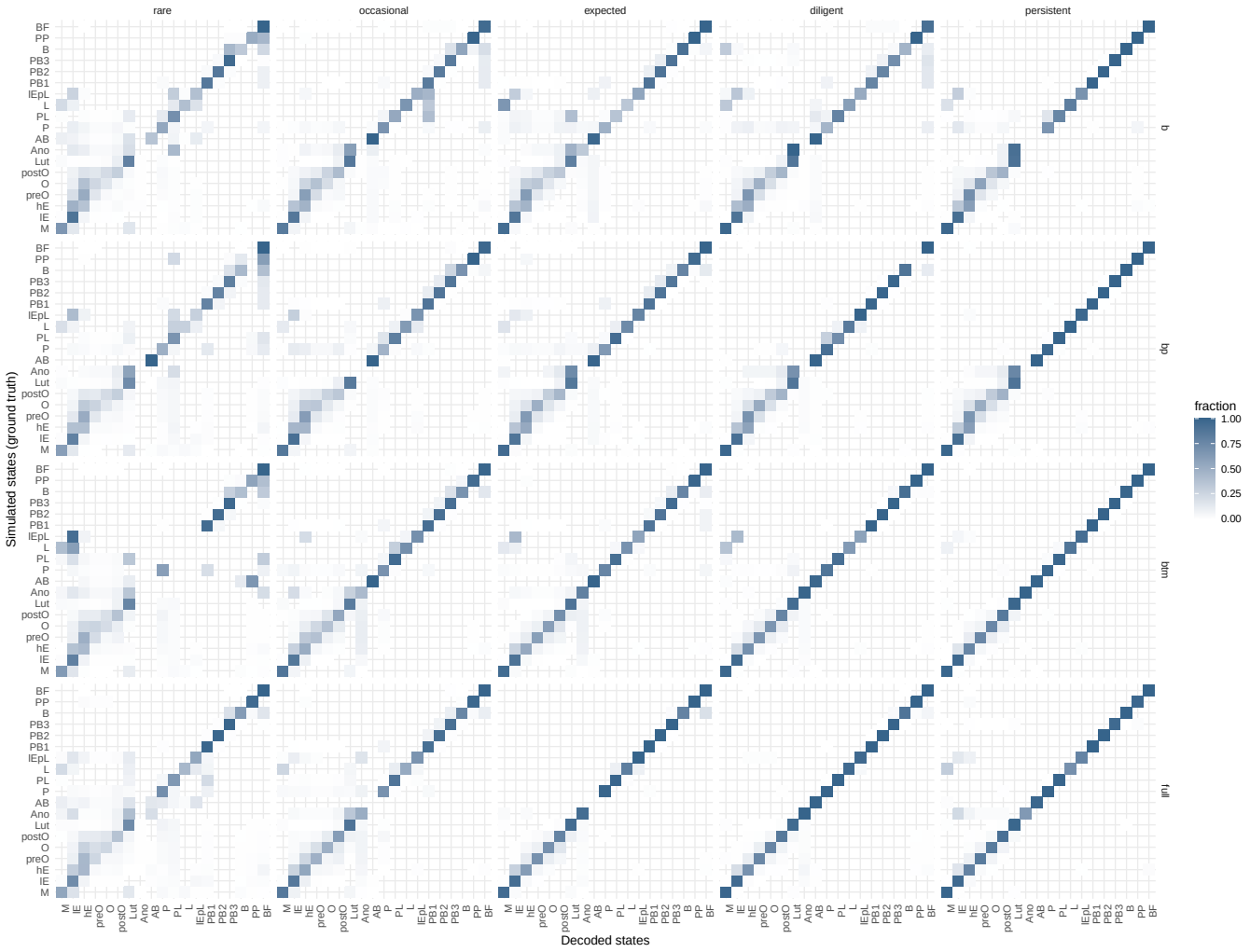


Figure 4.5: Confusion matrix for the decoding on synthetic data.

```
save(g_synth_conf_mat, file = "../Data/plots/g_synth_conf_mat.Rdata")
```

4.2 On Kindara data

We compare the performances of decoding with different models: a simple HMM, a weakly specified HSMM, the proposed HSMM. We also evaluate the impact of our hierarchical approach by comparing the performances of our HSMM with hierarchical decoding (h-HSMM) and of our HSMM without the hierarchical decoding (HSMM).

```
models = read_feather(paste0(IO$output_data, "models.feather"))

models =
  models %>%
  mutate(
    model_name =
      str_c(
        ifelse(approach == "adaptative","h-",""),
        model %>%
          str_remove("R_") %>%
          str_replace("hmm","HMM") %>%
          str_replace("hsmm", "HSMM") %>%
          str_replace("_"," ")
      ),
    model_name = model_name %>%
      factor(., levels = unique(model_name)),
    type = ifelse(fit_model, "fitted","specified"),
    type = type %>% factor(., levels = unique(type))
  )
```

4.2.1 Accuracies

```
source("Scripts/00_performances_analysis.R")

tic()
accuracies =
  map_dfr(
    .x = 1:nrow(models),
    .f = function(i){
      this_decoding_res =
        compute_accuracy(
          data_file = paste0(IO$output_data, "processed_app_data.feather"),
          GT_file =  paste0(IO$output_data, "ML.feather"),
          decoding_file = str_c(IO$output_data, "decodings/", models$file_name[i])
        )
      bind_cols(
        models[rep(i,2),],
        data.frame(
          accuracy_type = c("accuracy","weighted mean of sample accuracy"),
          value = c(this_decoding_res$Accuracy,
                    this_decoding_res$Weighted_mean_of_sample_accuracy)
        )
      )
    }
  )
toc() # 3 sec
```

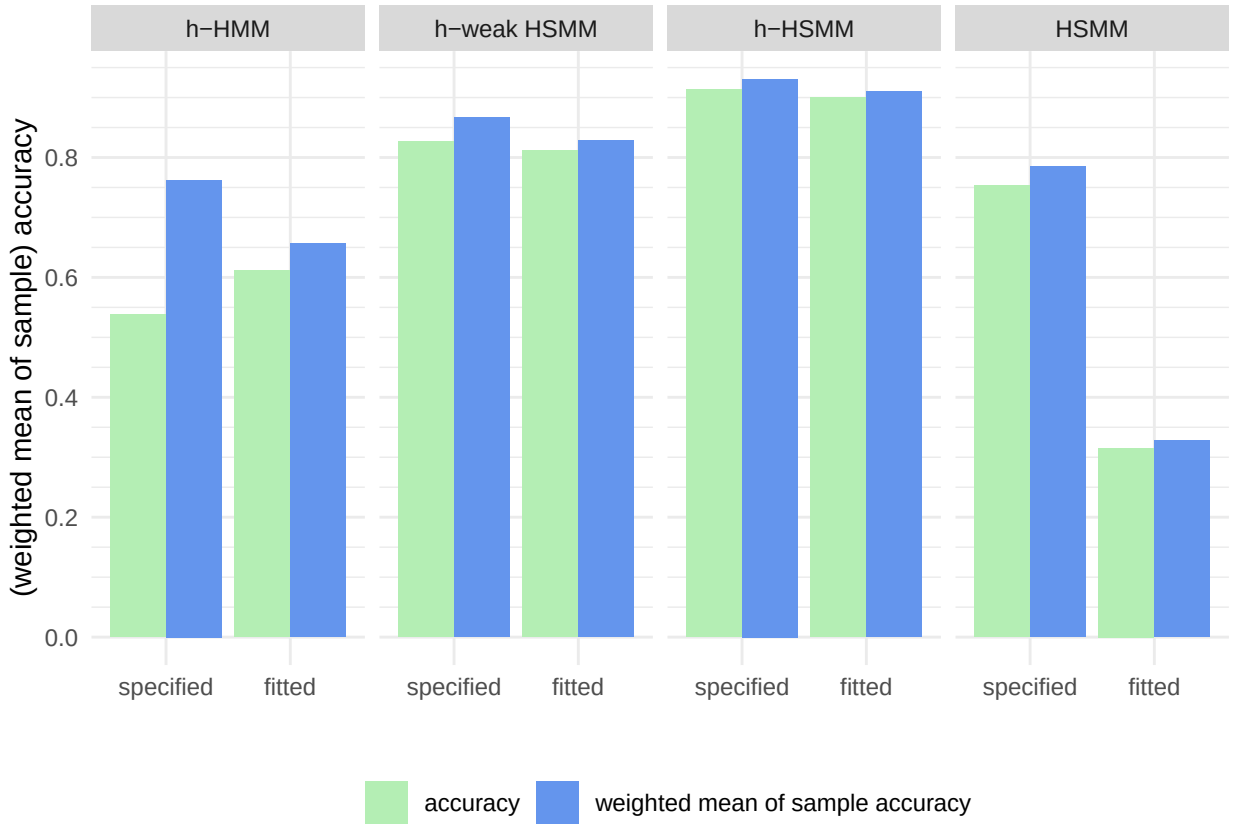
```
## 4.515 sec elapsed
```

```
accuracies
```

model	approach	fit_model	file_name	model_name	type	accura
R_hmm	adaptative	FALSE	adaptative_R_hmm_specified.feather	h-HMM	specified	accura
R_hmm	adaptative	FALSE	adaptative_R_hmm_specified.feather	h-HMM	specified	weigh
R_hmm	adaptative	TRUE	adaptative_R_hmm_fitted.feather	h-HMM	fitted	accura
R_hmm	adaptative	TRUE	adaptative_R_hmm_fitted.feather	h-HMM	fitted	weigh
R_weak_hsmm	adaptative	FALSE	adaptative_R_weak_hsmm_specified.feather	h-weak HSMM	specified	accura
R_weak_hsmm	adaptative	FALSE	adaptative_R_weak_hsmm_specified.feather	h-weak HSMM	specified	weigh
R_weak_hsmm	adaptative	TRUE	adaptative_R_weak_hsmm_fitted.feather	h-weak HSMM	fitted	accura
R_weak_hsmm	adaptative	TRUE	adaptative_R_weak_hsmm_fitted.feather	h-weak HSMM	fitted	weigh
R_hsmm	adaptative	FALSE	adaptative_R_hsmm_specified.feather	h-HSMM	specified	accura
R_hsmm	adaptative	FALSE	adaptative_R_hsmm_specified.feather	h-HSMM	specified	weigh
R_hsmm	adaptative	TRUE	adaptative_R_hsmm_fitted.feather	h-HSMM	fitted	accura
R_hsmm	adaptative	TRUE	adaptative_R_hsmm_fitted.feather	h-HSMM	fitted	weigh
R_hsmm	non-adaptative	FALSE	non-adaptative_R_hsmm_specified.feather	HSMM	specified	accura
R_hsmm	non-adaptative	FALSE	non-adaptative_R_hsmm_specified.feather	HSMM	specified	weigh
R_hsmm	non-adaptative	TRUE	non-adaptative_R_hsmm_fitted.feather	HSMM	fitted	accura
R_hsmm	non-adaptative	TRUE	non-adaptative_R_hsmm_fitted.feather	HSMM	fitted	weigh

```
write_feather(accuracies, path = "../Data/decodings/rw_accuracies.feather")
```

```
ggplot(accuracies, aes(y = value, x = type, fill = accuracy_type))+
  geom_bar(stat = "identity", position = "dodge") +
  facet_grid(. ~ model_name) +
  xlab("") +
  ylab("(weighted mean of sample) accuracy") +
  scale_y_continuous(breaks = seq(0,1,by = 0.2), minor_breaks = seq(0,1,by = 0.05)) +
  scale_fill_manual("",values = c("darkseagreen2","cornflowerblue")) +
  theme(strip.background = element_rect(fill = "gray85", colour = NA),
        legend.position = "bottom")
```



4.2.2 Confusion matrices

```
tic()
conf_matrices =
  map_dfr(
    .x = 1:nrow(models),
    .f = function(i){
      this_decoding_res =
        compute_accuracy(
          data_file = paste0(IO$output_data, "processed_app_data.feather"),
          GT_file =  paste0(IO$output_data, "ML.feather"),
          decoding_file = str_c(IO$output_data, "decodings/", models$file_name[i])
        )
      this_decoding_res$Conf_Matrix %>%
        mutate(model_name = models$model_name[i],
               type = models$type[i])

    }
  )
toc() # 2 sec
```

```
## 3.393 sec elapsed
```

```
write_feather(conf_matrices, path = "../Data/decodings/rw_conf_matrices.feather")
```

```
ggplot(
  conf_matrices,
  aes(x = decoded_state_name, y = GT_state_name, fill = proportion)) +
  geom_tile() +
  facet_grid( type + proportion_type   ~  model_name) +
  scale_fill_gradient(
    "Proportion of time-points    \nwith a given manual label    ",
    low = "white", high = "steelblue4", limits = c(0,1)) +
  coord_fixed() +
  ylab("Manually labelled states") + xlab("Decoded states")+
  theme(axis.text.x = element_text(angle = 90, hjust = 1),
        strip.background = element_rect(fill = "gray85", colour = "white"))
```

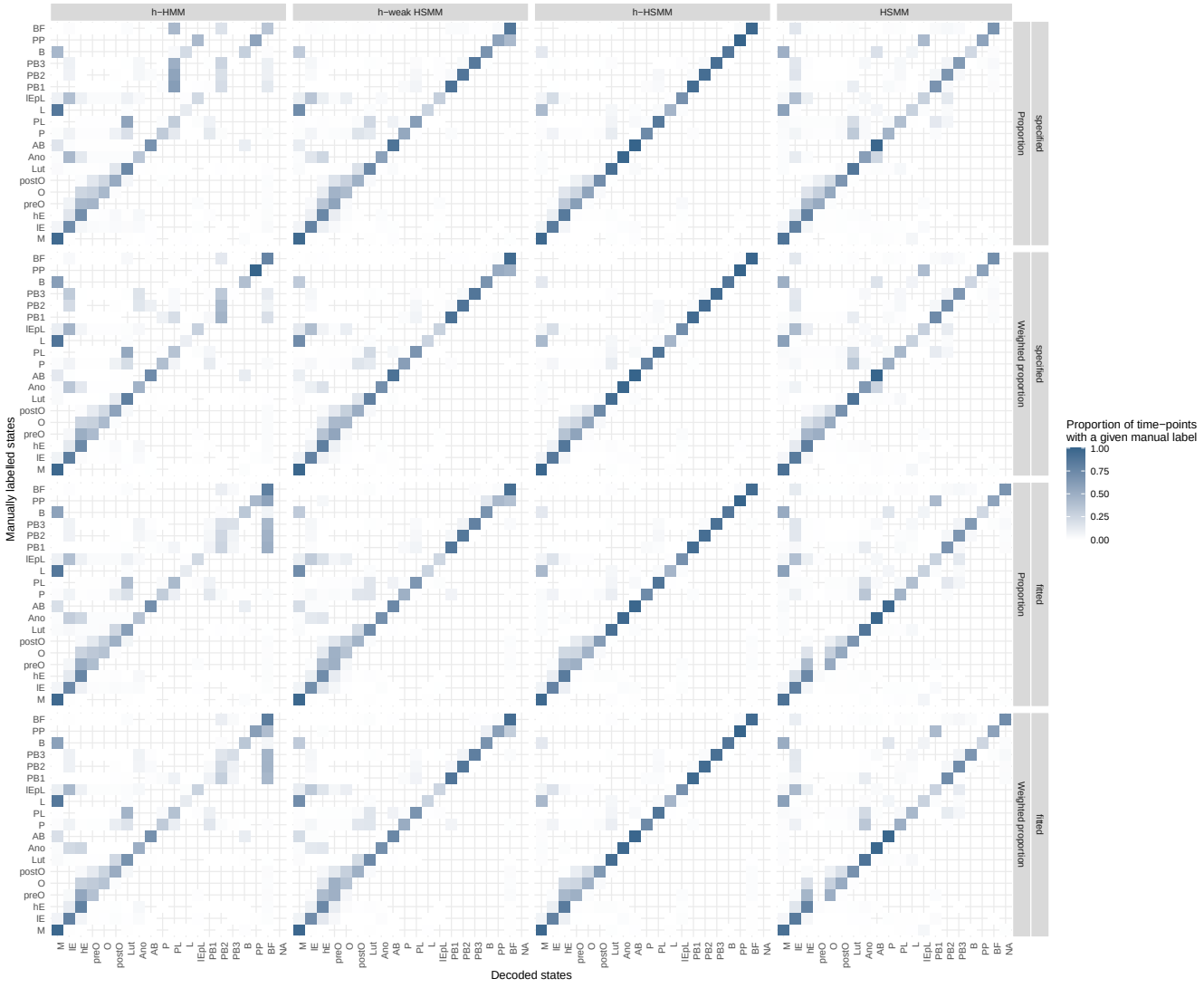


Figure 4.6: Confusion matrices for the decoding on our real-world data for all models.

```
ggplot(
  conf_matrices %>% filter(type == "specified", proportion_type != "Proportion"),
  aes(x = decoded_state_name, y = GT_state_name, fill = proportion)) +
  geom_tile() +
  facet_grid(. ~ model_name) +
  scale_fill_gradient(
    "Proportion of time-points    \nwith a given manual label    ",
    low = "white", high = "steelblue4", limits = c(0,1)) +
  coord_fixed() +
  ylab("Manually labelled states") + xlab("Decoded states")+
  theme(axis.text.x = element_text(angle = 90, hjust = 1),
        strip.background = element_rect(fill = "gray85", colour = NA),
        legend.position = "bottom")
```

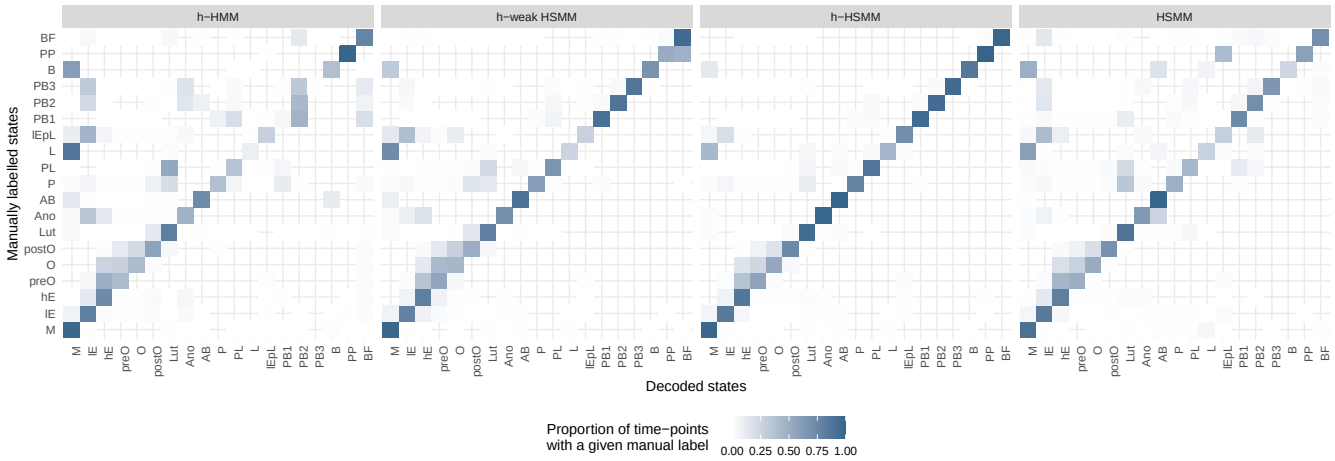


Figure 4.7: Confusion matrices for the decoding on our real-world data for all specified models.

```
ggplot(
  conf_matrices %>% filter(model_name == "h-HSMM"),
  aes(x = decoded_state_name, y = GT_state_name, fill = proportion)) +
  geom_tile() +
  facet_grid(proportion_type ~ type) +
  scale_fill_gradient(
    "Proportion of time-points    \nwith a given manual label    ",
    low = "white", high = "steelblue4", limits = c(0,1)) +
  coord_fixed() +
  ylab("Manually labelled states") + xlab("Decoded states")+
  theme(axis.text.x = element_text(angle = 90, hjust = 1),
        strip.background = element_rect(fill = "gray85", colour = NA)) +
  ggtitle("Confusion matrix for our h-HSMM")
```

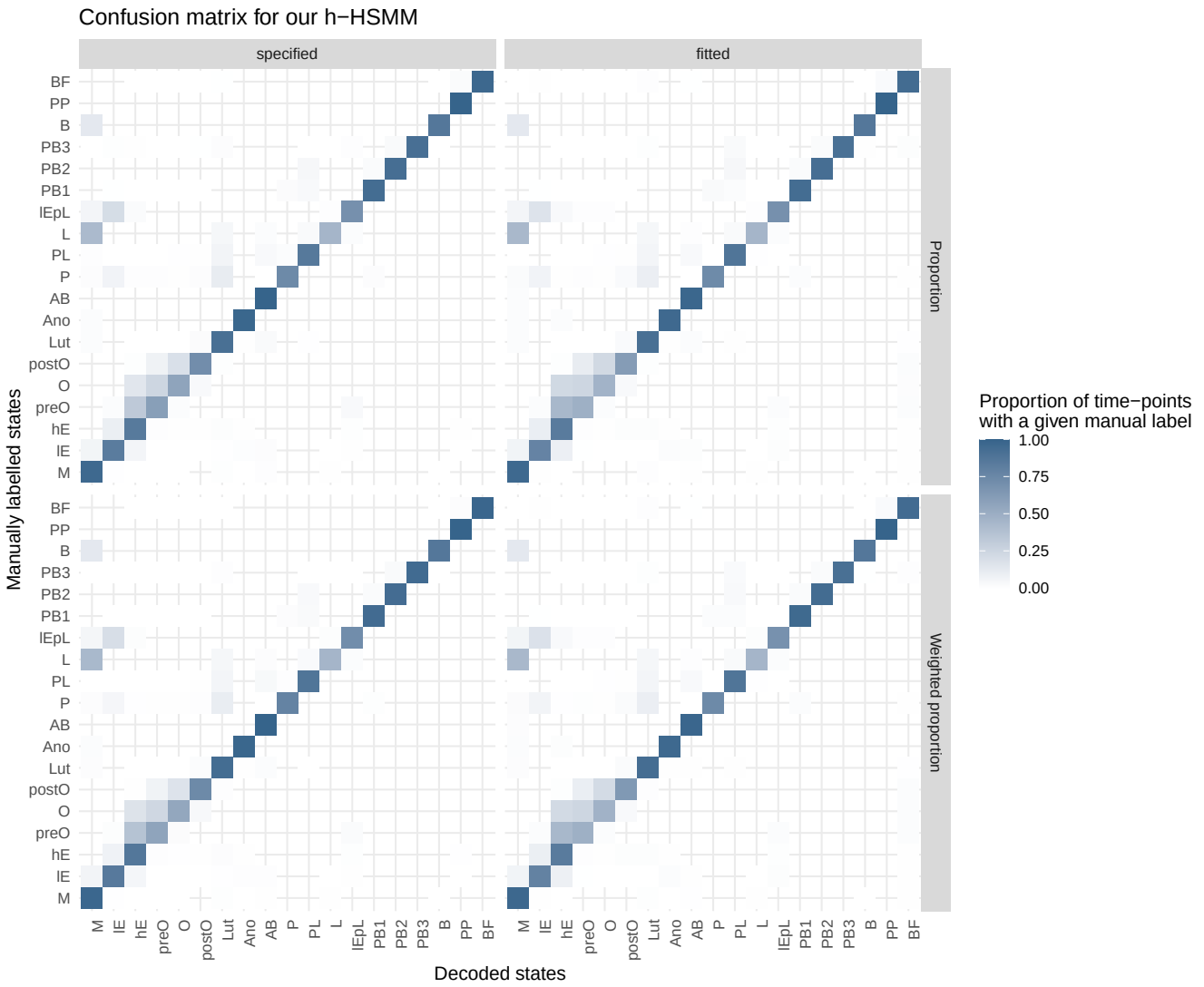


Figure 4.8: Confusion matrices (accuracy and weighted accuracy) for the decoding on our real-world data for our proposed h-HSMM.

4.2.3 Predicted state probabilities

```
tic()
XP_all =
  map_dfr(
    .x = 1:nrow(models),
    .f = function(i){
      this_decoding_res =
        compute_accuracy(
          data_file = paste0(IO$output_data, "processed_app_data.feather"),
          GT_file =  paste0(IO$output_data, "ML.feather"),
          decoding_file = str_c(IO$output_data, "decodings/", models$file_name[i])
        )
      this_decoding_res$XP %>%
        mutate(model_name = models$model_name[i],
               type = models$type[i])
    }
  )
toc() # 3 sec
```

```
## 3.089 sec elapsed
```

```
write_feather(XP_all, path = "../Data/decodings/XP_all.feather")
```

```
ggplot(XP_all,
       aes(x = correct_prediction, y = prob,
           col = correct_prediction, fill = correct_prediction)) +
  geom_boxplot(outlier.size = 0.5, alpha = 0.5, outlier.alpha = 0.5) +
  facet_grid(type ~ model_name) +
  ylab("predicted state probability") +
  xlab("") +
  scale_fill_discrete("state prediction",
                      breaks = c(FALSE, TRUE),
                      labels = c("incorrect", "correct")) +
  scale_color_discrete("state prediction",
                      breaks = c(FALSE, TRUE),
                      labels = c("incorrect", "correct")) +
  theme(axis.text.x = element_blank(),
        strip.background = element_rect(fill = "gray90", color = "transparent"))
```

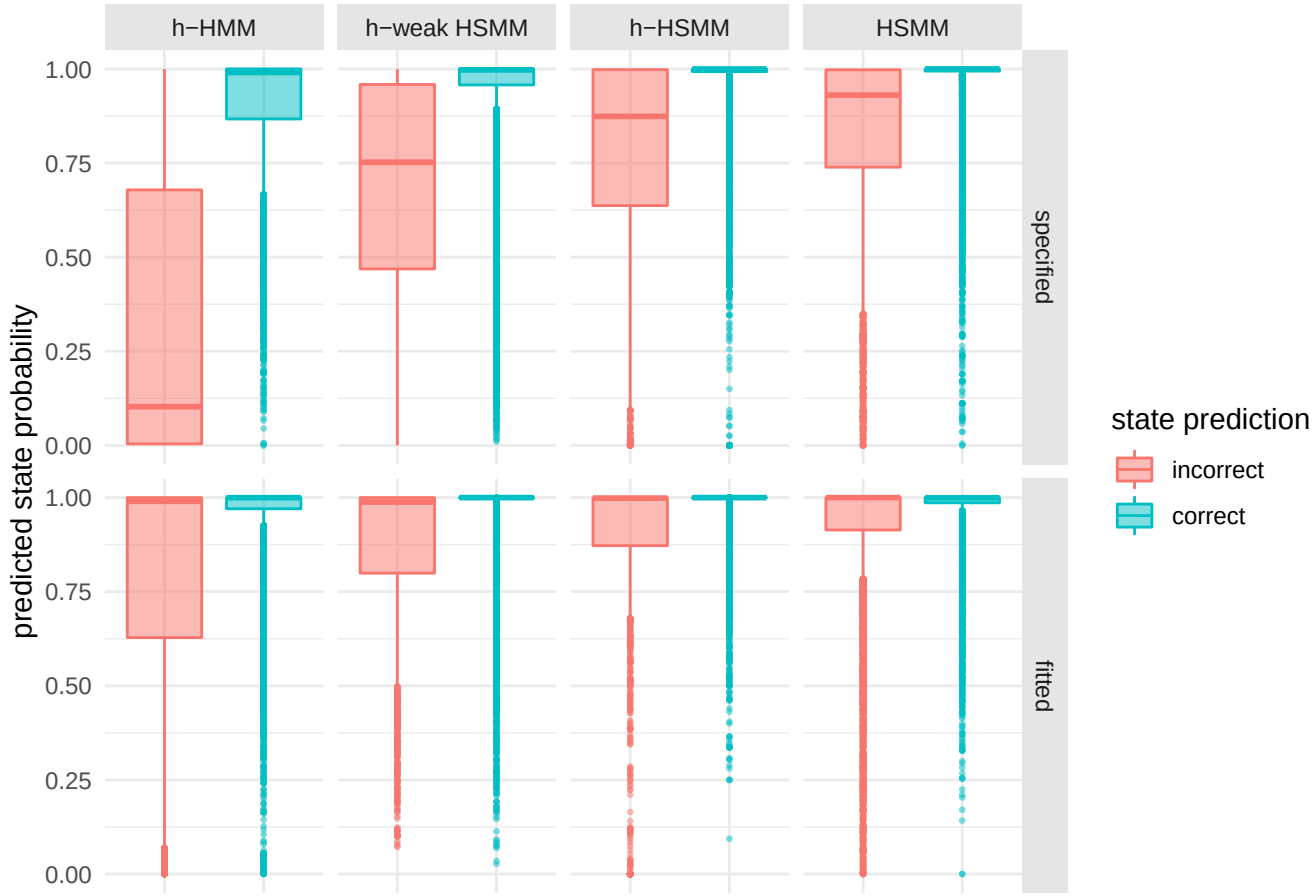


Figure 4.9: Predicted state probabilities when prediction are correct (match the ground truth) or incorrect (different from ground truth).

4.2.4 Pregnancy durations

Besides the accuracy computed from our sparse manual labeling, we also evaluate the performance of each model by looking at the distribution of pregnancy duration.

```
tic()
pregnancies =
```

```
  map_dfr(
    .x = 1:nrow(models),
    .f = function(i){
      this_pregnancy_res =
        compute_pregnancy_duration(
          decoding_file = str_c(IO$output_data, "decodings/", models$file_name[i])
        )
      this_pregnancy_res %>%
        mutate(model_name = models$model_name[i],
               model_type = models$type[i])

    }
  )
toc() # 2 sec
```

```
## 2.651 sec elapsed
```

```
pregnancies =
  pregnancies %>%
  mutate(type = type %>% factor(., levels = c("pregnancy with loss", "pregnancy with birth"))) %>%
  filter(!is.na(type))

write_feather(pregnancies, path = "../Data/decodings/pregnancies.feather")
```

```
ggplot(pregnancies %>% filter(str_detect(type, "pregnancy"), model_type == "specified"),
       aes(x = duration/7, fill = type)) +
  geom_histogram(position = "identity", binwidth = 1, alpha = 0.75) +
  facet_grid(model_name ~ ., scale = "free_x") +
  xlab("duration (weeks)") +
  theme(strip.background = element_rect(fill = "gray85", colour = NA),
        legend.position = "bottom")
```

```
## Warning: Removed 17 rows containing non-finite values (stat_bin).
```

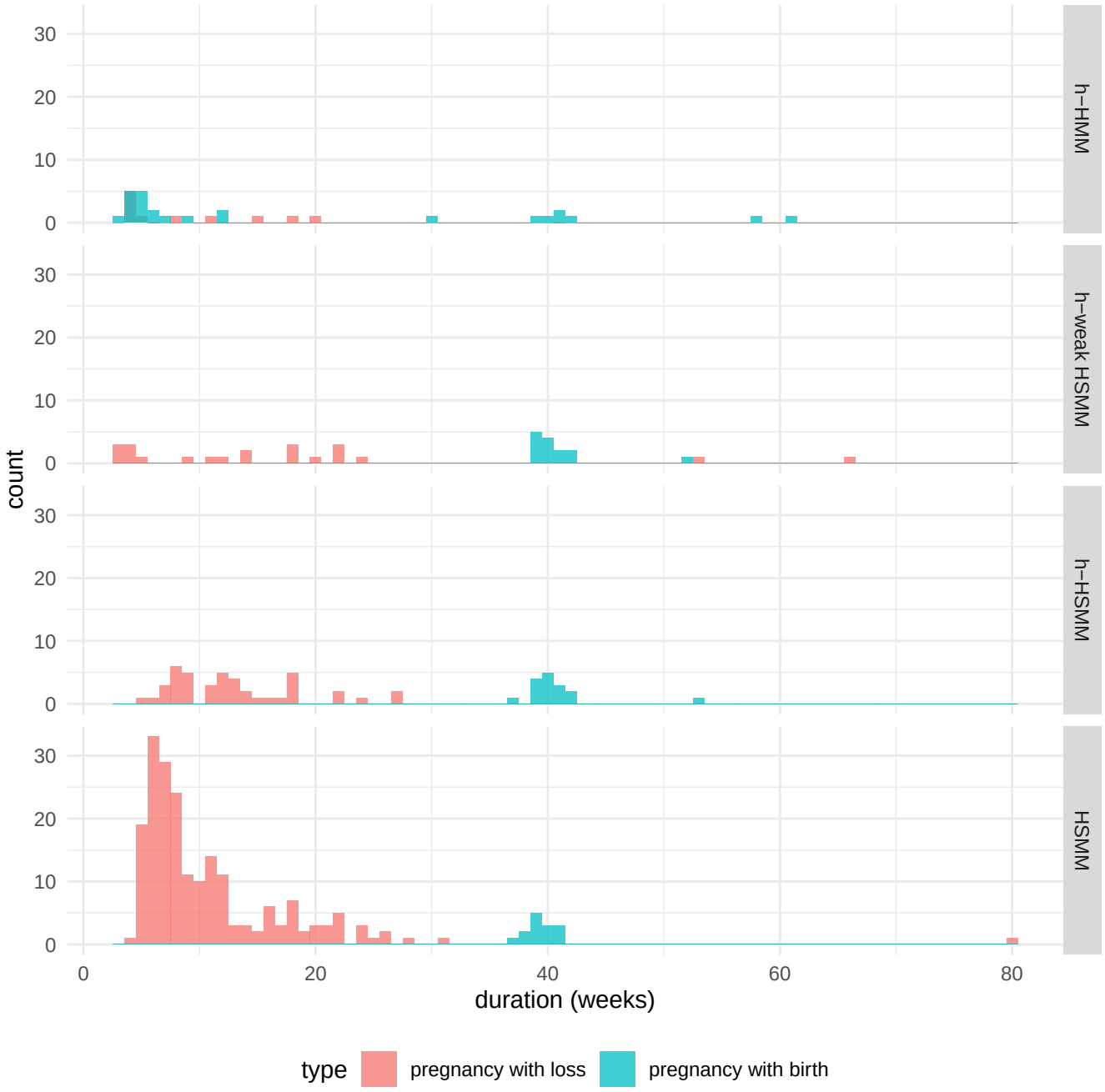


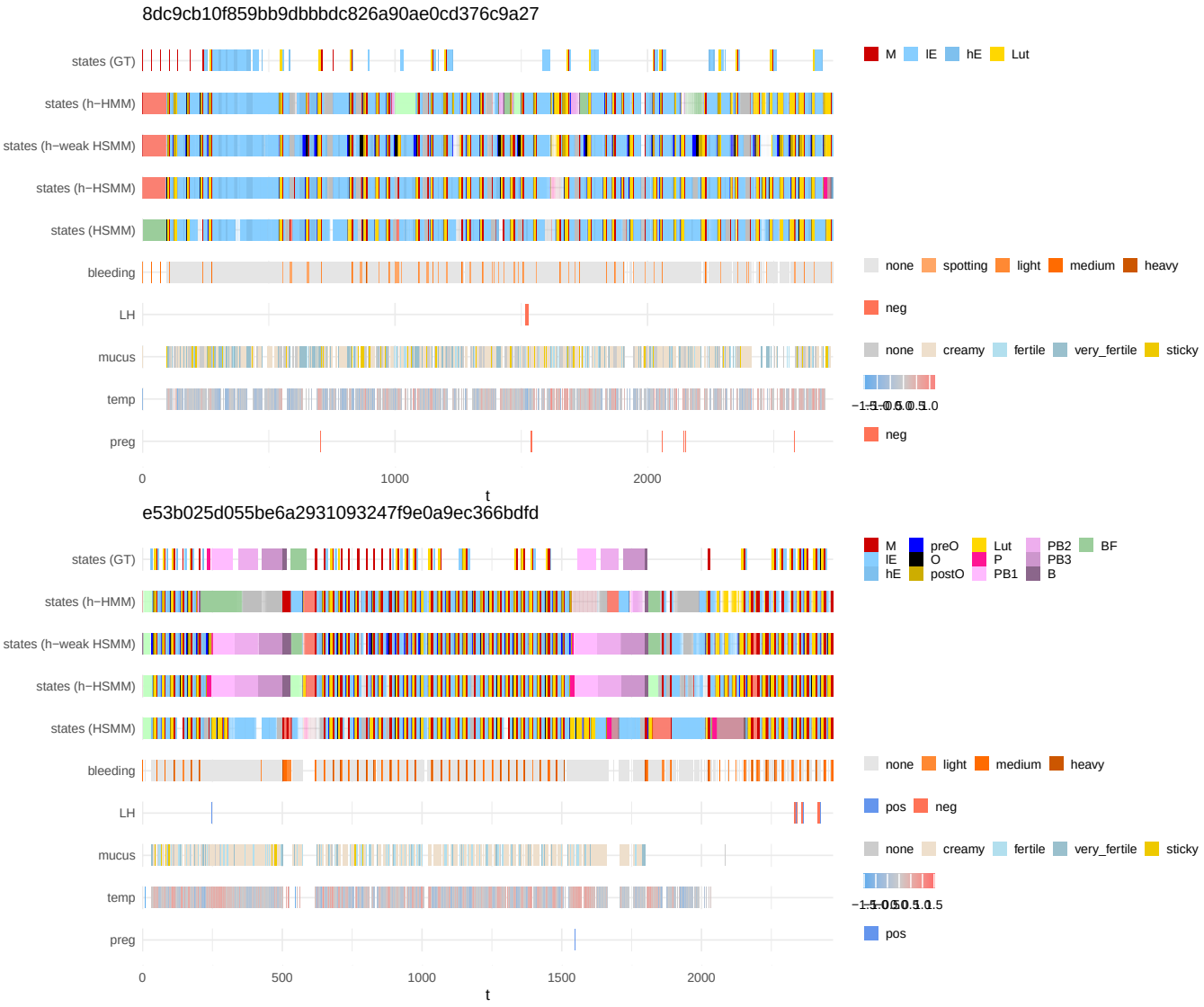
Figure 4.10: Distribution of pregnancy durations as identified by the different models.

4.2.5 Sequences

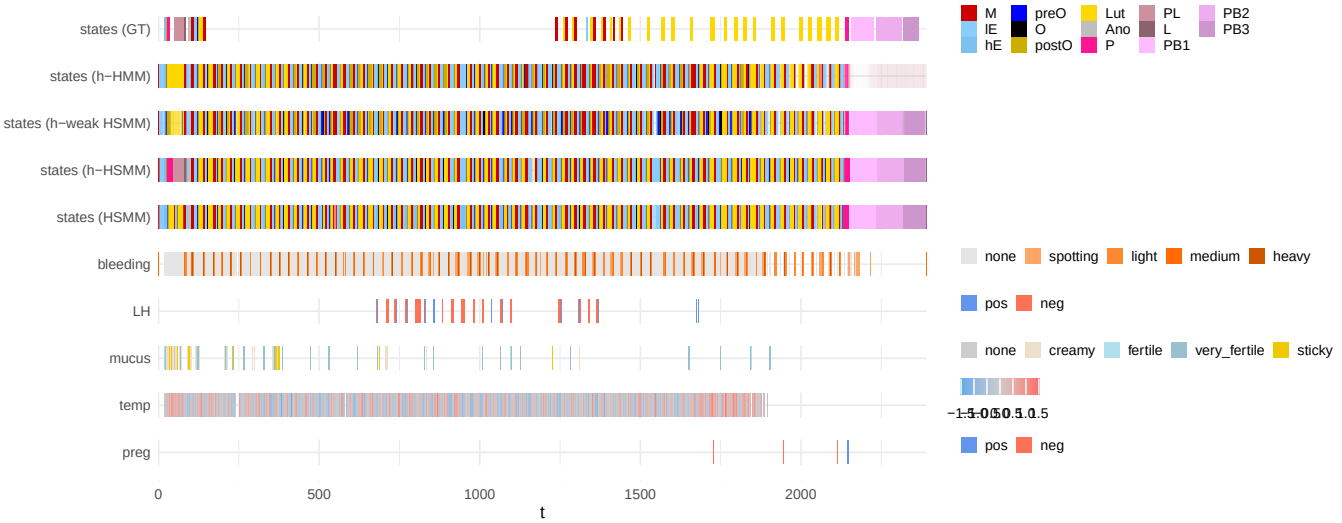
All sequences with specified models decodings

```
X_all_seq_4_viz =
  create_X_for_decoding_viz(
    data_file = paste0(IO$output_data, "processed_app_data.feather"),
    GT_file = paste0(IO$output_data, "ML.feather"),
    decoding_files = models %>% filter(type == "specified") %>% mutate(name = model_name)
  )

# X_all_seq_4_viz %>%
#   group_by(seq_id) %>%
#   summarize(n_wrong = sum(`state_h-HSMM` != state_GT, na.rm = TRUE)) %>%
```

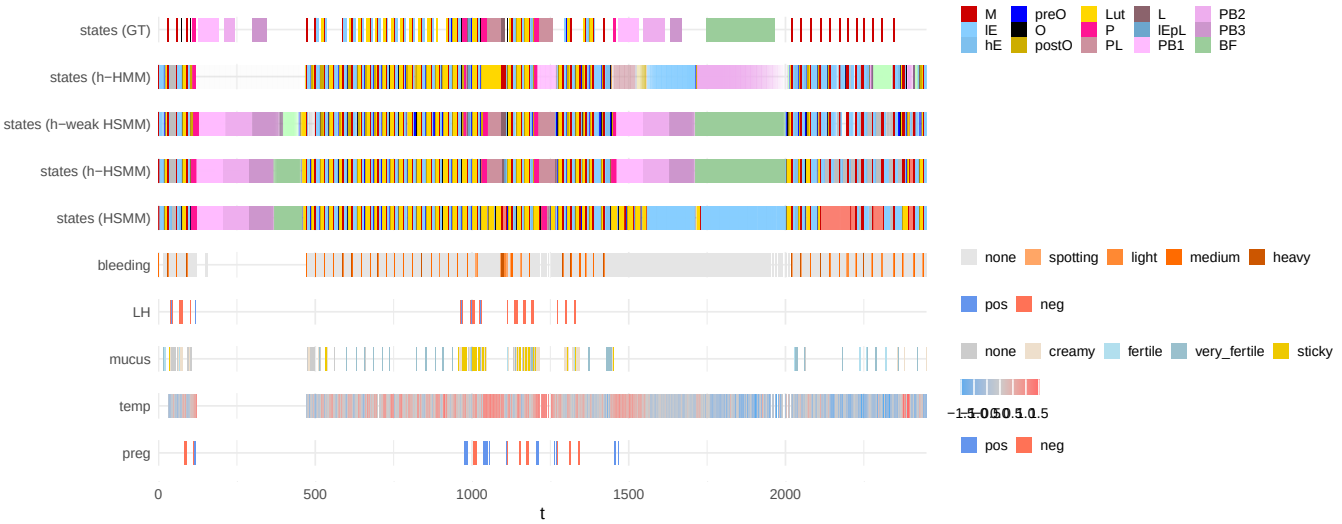
```r
#   arrange(-n_wrong)

write_feather(X_all_seq_4_viz, path = str_c(IO$output_data,"X_with_decoding_specified_models.feather"))

for(sid in unique(X_all_seq_4_viz$seq_id)[1:10]){
  this_X = X_all_seq_4_viz %>% filter(seq_id == sid)
  plot_hsmm_seq(X = this_X, model = R_hsmm, title = sid, compact_view = TRUE) %>%
    print()
}
```



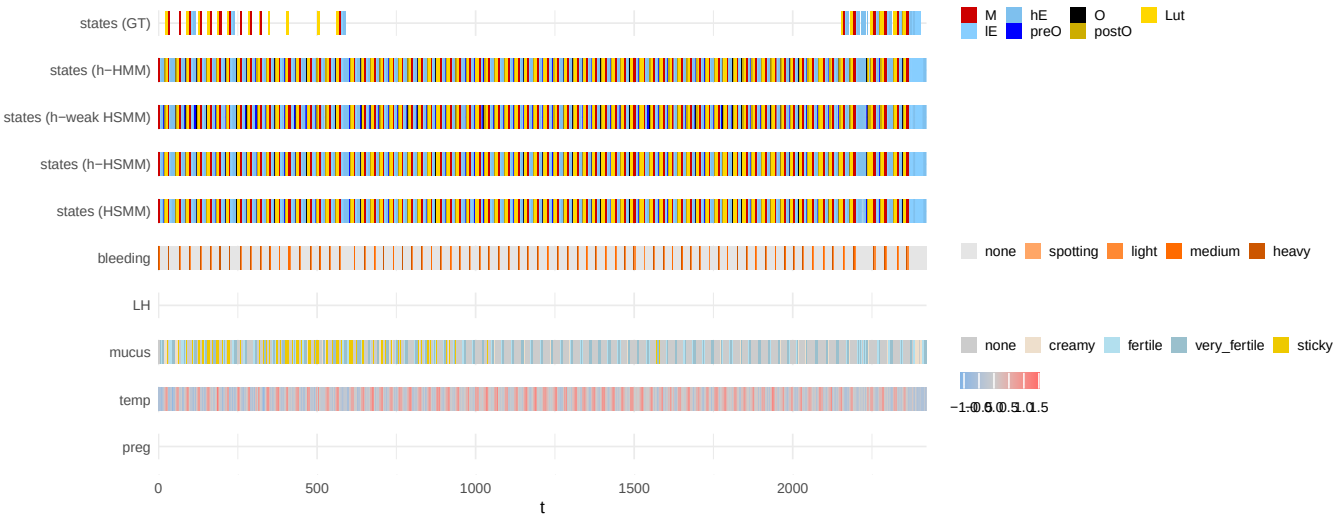
0b0ff00a3bd310ab401d371981471f46e96c7364



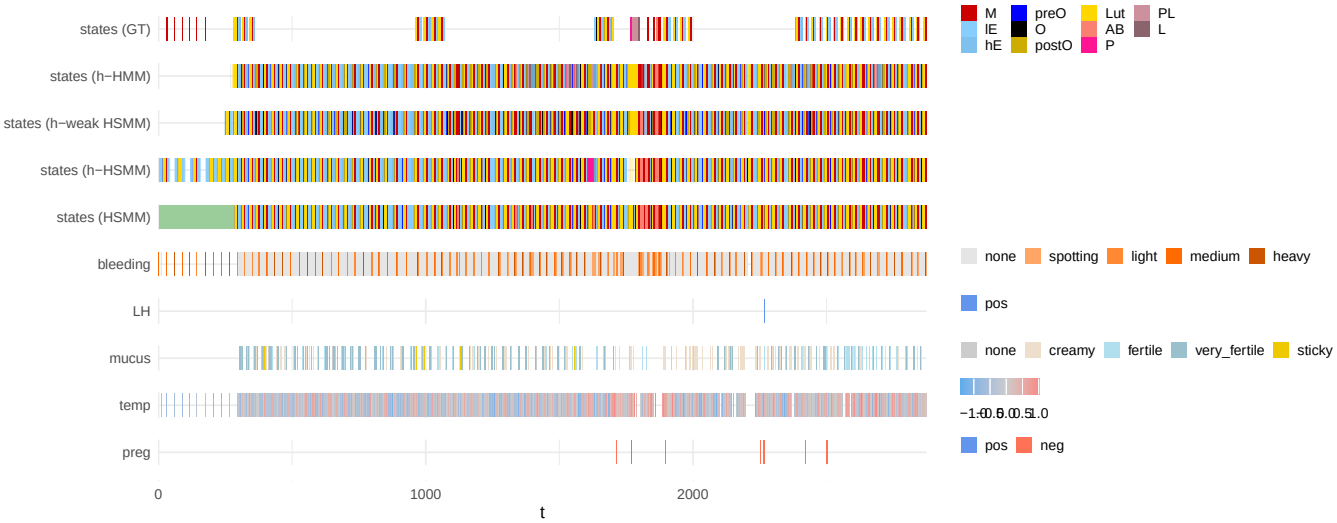
3ec8d9a84ecb1efd9640201e69e458d148b8b118



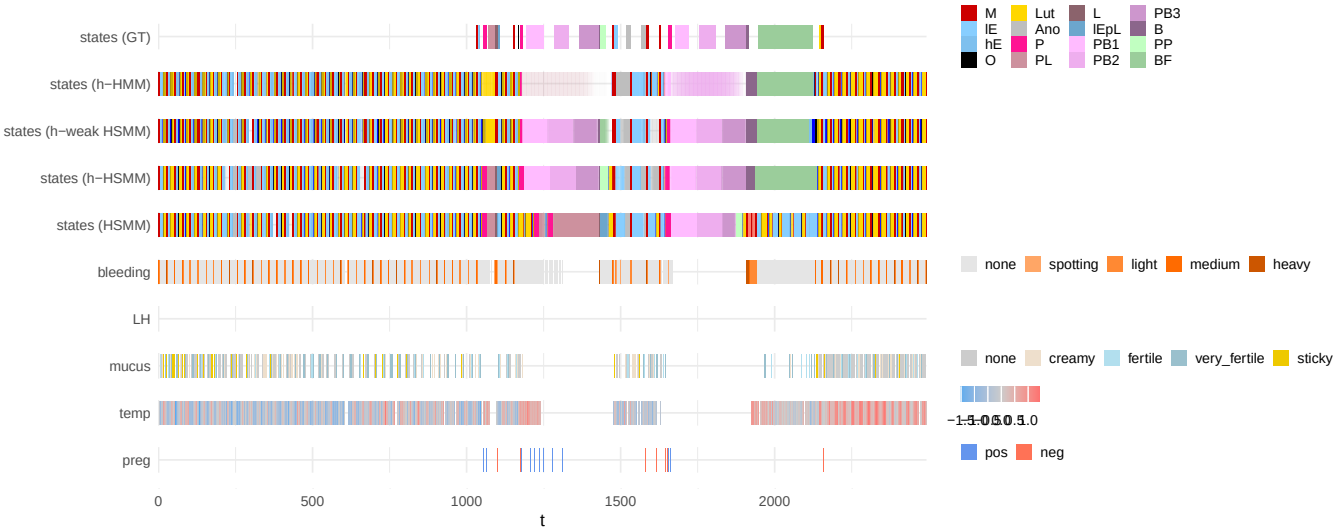
db0bfa84dcff70b46e928c9b9d4569a4fbf91416



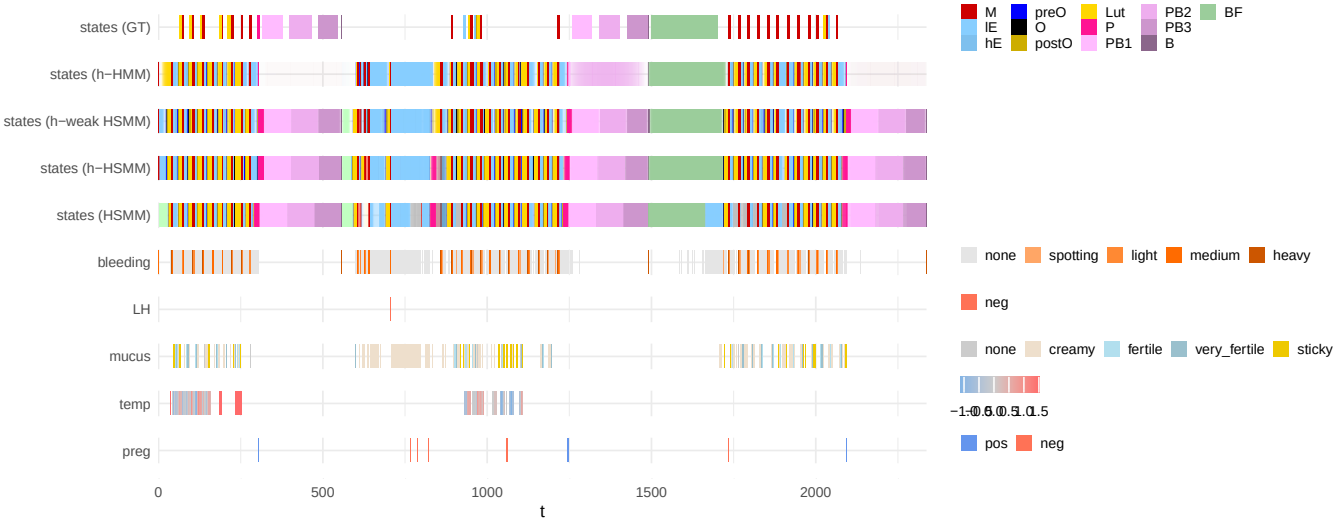
f0f8e3a18fba177fcaf580381c47f39038cf06bf

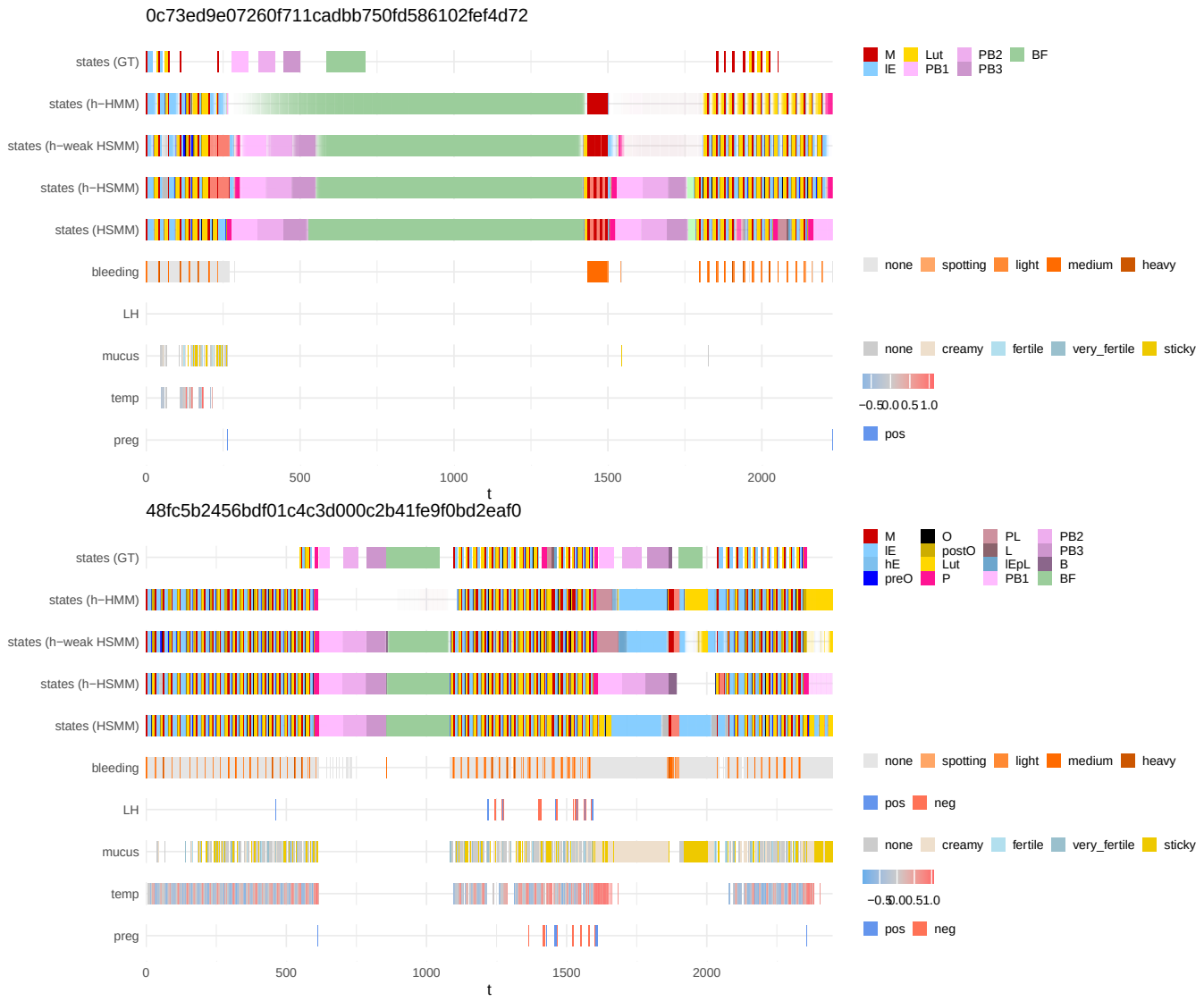


4517215d2a1f256e6077db5e4ecb359804b9c99e



02a0954f67de02f2e4930de01160b203508e1df8

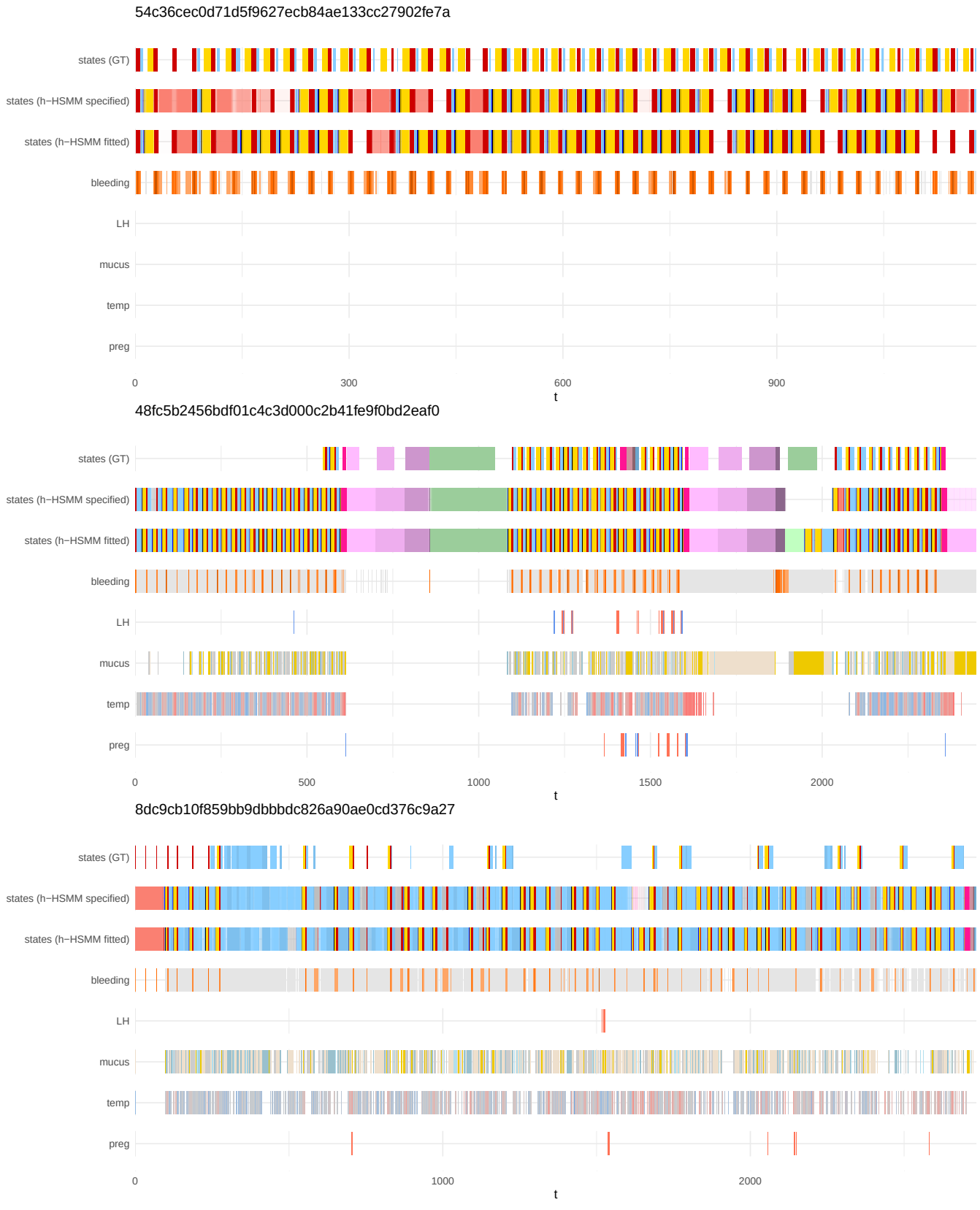




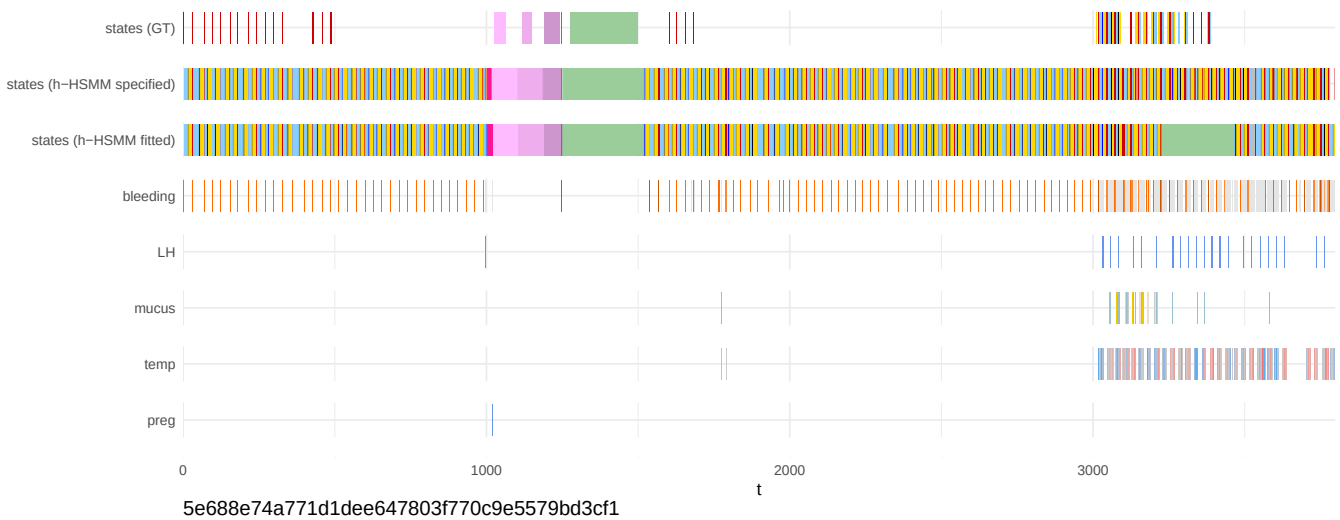
Sequences with differences between the specified and fitted models

```
X =
  create_X_for_decoding_viz(
    data_file = paste0(IO$output_data, "processed_app_data.feather"),
    GT_file = paste0(IO$output_data, "ML.feather"),
    decoding_files =
      models %>%
      filter(model_name == "h-HSMM") %>%
      mutate(name = str_c(model_name, " ", type))
  )


selected_sequences =
  X %>%
  filter(!is.na(state_GT),
         `state_h-HSMM specified` != `state_h-HSMM fitted`) %>%
  group_by(seq_id) %>%
  summarize(n = n(), .groups = "drop") %>%
  arrange(-n) %>%
  slice_head(n = 10)


for(sid in unique(selected_sequences$seq_id)){
```

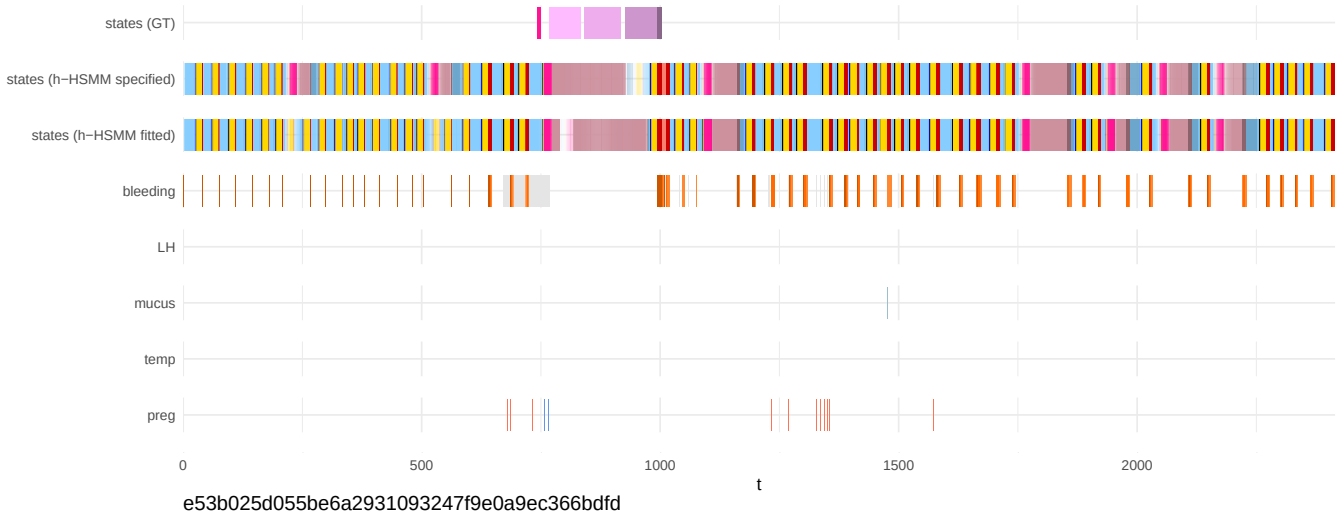
```
  this_X = X %>% filter(seq_id == sid)
  plot_hsmm_seq(
    X = this_X, model = R_hsmm, title = sid,
    compact_view = TRUE, add_color_legend_in_compact_view = FALSE) %>%  print()
}
```



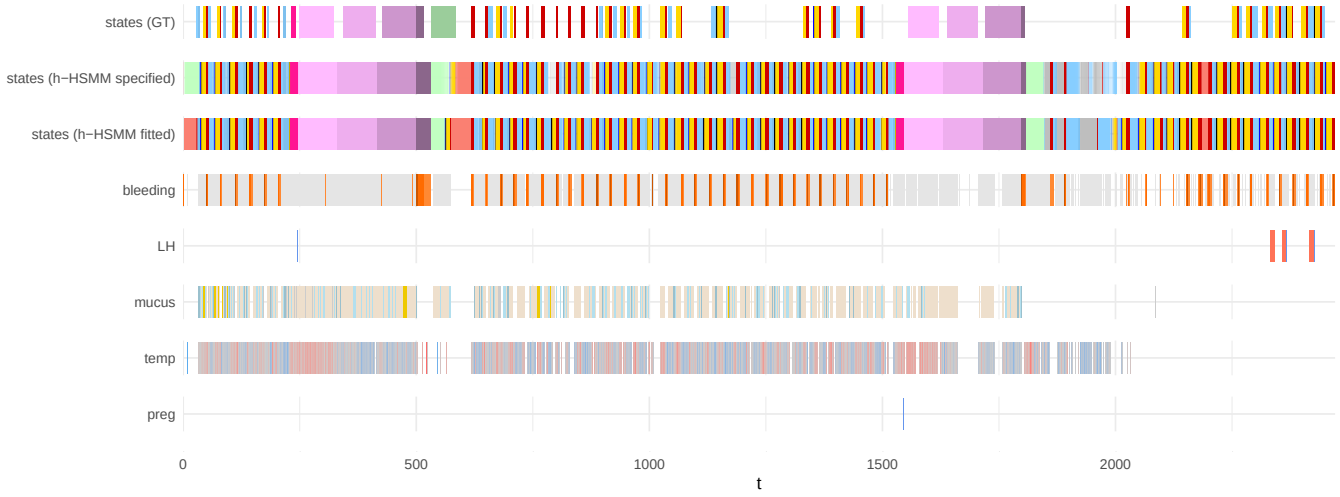
008a7d168d2ae8ff5fafc410f31cedbff386dcba



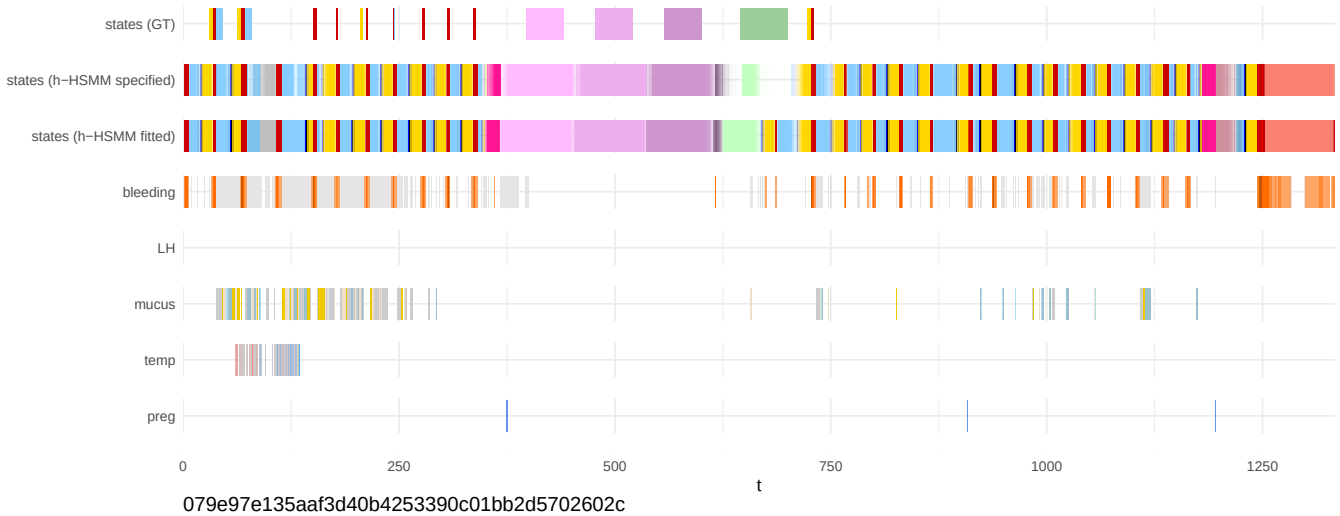
5e688e74a771d1dee647803f770c9e5579bd3cf1



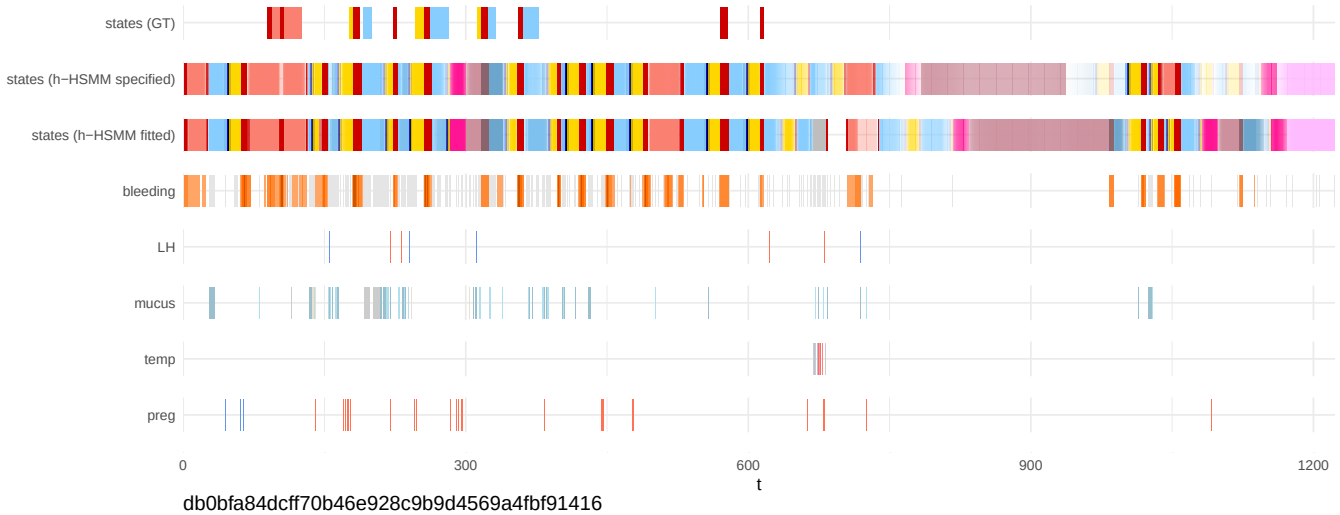
e53b025d055be6a2931093247f9e0a9ec366bdfd



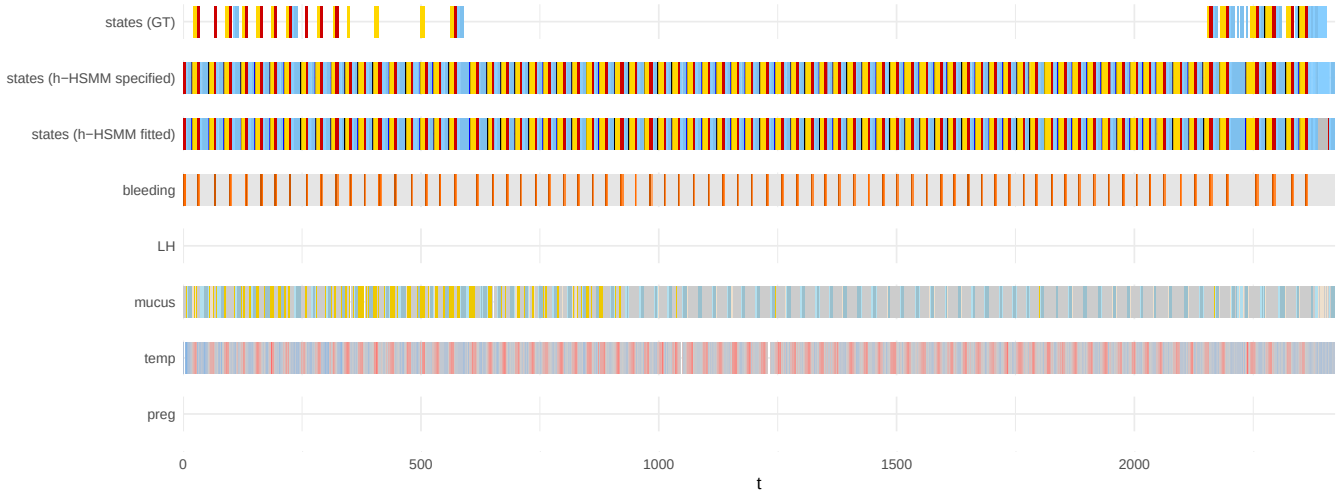
0da615fab9908566273bee16f0ef2ec9b55a47db



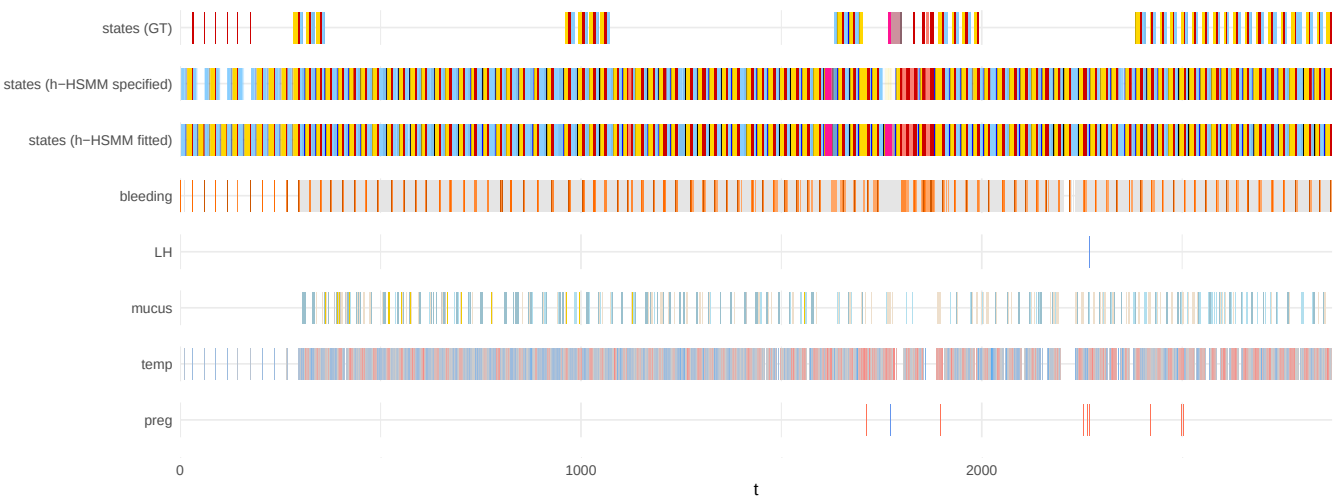
079e97e135aaf3d40b4253390c01bb2d5702602c



db0bfa84dcff70b46e928c9b9d4569a4fbf91416



f0f8e3a18fba177fcaf580381c47f39038cf06bf



5 Predicting cycle length and next-day observations

In this section, we evaluate the ability of our model to learn the cycle characteristics of users and to predict the next period of irregular users (*i.e.* users with irregular cycle length).

To do so, we filter users (or part of their time-series) to keep those with irregular cycles and who log with a tracking frequency of at least 1/3 days.

We compare the predictions from our model with the baseline prediction which is based on the mean or median cycle length of previous cycles.

5.1 Loading models and decoding results

```
load("../Data/models/R_hsmm.Rdata", verbose = TRUE)
```

```
RES.file = paste0(IO$output_data, "decodings/adaptative_R_hsmm_specified.feather")
RES = read_feather(path = RES.file)
rm(RES.file)

X = read_feather(path = paste0(IO$output_data, "processed_app_data.feather"))
```

5.2 Selecting users and cycles

```
ans = identify_cycles(RES = RES, X = X) ; RES = ans$RES ; cycles = ans$cycles
users = cycle_characteristics_per_user(cycles = cycles)
```

We select cycles - which were found to be ovulatory cycles with our hsmm-based labeling - in which the states were estimated with a probability over 30% - in which users have at least tracked their mucus, temperature or ovulation tests - which were part of a stretch of at least 5 consecutive cycles fulfilling the above conditions and which had a cumulative absolute differences in cycle length of at least 5 days over the consecutive cycles.

```
cycles =
  cycles %>%
  arrange(seq_id, cycle_number) %>%
  group_by(seq_id) %>%
  mutate(
    ovulatory_cycles = frollsum(ovulatory_cycle, n = 5, align = "left") == 5,
    ovu_prob_ok = frollsum(ovu_prob >= 0.5, n = 5, align = "left") == 5,
    state_prob_ok = frollsum(min_prob >= 0.3, n = 5, align = "left") == 5,
    sufficient_tracking = frollsum(tracking_behavior %in% c("BLH","BM","BT","BTM"), n = 5, align = "left") ==
    variability =
      (frollapply(cycle_length, n = 5, align = "left",
                  FUN = function(x) sum(abs(diff(x)))) >= 8) %>%
      replace_na(FALSE),
    ok_cycles = ovulatory_cycles & ovu_prob_ok & state_prob_ok & sufficient_tracking  & variability) %>%
  ungroup()


selected_consecutive_cycles =
  cycles %>%
  filter(ok_cycles) %>%
  select(seq_id, cycle_number) %>%
  group_by(seq_id) %>%
  mutate(stretch_number = row_number(),
         stretch_id = str_c(seq_id, "_", stretch_number)) %>%
  ungroup() %>%
  filter(stretch_number %in% c(1,6,11,16,21)) %>%
```

```
  expand_grid(data.frame(cycle_number_increment = 0:4)) %>%
  mutate(cycle_number = cycle_number + cycle_number_increment,
         rel_cycle_nb = cycle_number_increment + 1) %>%
  select(-cycle_number_increment) %>%
  left_join(
    .,
    cycles %>% select(seq_id, cycle_number, cycle_start, cycle_end, cycle_length),
    by = c("seq_id","cycle_number")
    )


selected_consecutive_cycles %>% select(seq_id, stretch_id, cycle_number) %>% distinct() %>% nrow()
```

```
## [1] 155
```

```
selected_consecutive_cycles %>% select(seq_id, stretch_id) %>% distinct() %>% nrow()
```

```
## [1] 31
```

```
selected_consecutive_cycles %>% select(seq_id) %>% distinct() %>% nrow()
```

```
## [1] 21
```

5.3 Fitting the model to each user time series

5.3.1 Building the input dataset

```
# for each user
# for each consecutive stretch
# we keep the 4 first cycles (= training cycles)

input = data.frame()
input_baseline = data.frame()
output = data.frame() # for cycle length prediction
fifth_cycles = data.frame()

for(str_id in unique(selected_consecutive_cycles$stretch_id)){
  cat(str_id, "\t")
  cycles_this_stretch = selected_consecutive_cycles %>% filter(stretch_id == str_id)

  # X input

  cycles_for_input = cycles_this_stretch %>% filter(rel_cycle_nb %in% 1:4)
  X_input =
    X %>%
    filter(seq_id == cycles_this_stretch$seq_id[1],
           t >= min(cycles_for_input$cycle_start),
           t <= max(cycles_for_input$cycle_end)+5) %>%  # we add 5 days so that the next period is included
    rename(user_id = seq_id) %>%
    mutate(seq_id = str_id)

  input = bind_rows(input, X_input)


  # baseline input
  this_input_baseline =
    cycles_for_input %>%
    select(seq_id, cycle_number, cycle_length) %>%
    rename(user_id = seq_id) %>%
```

```
    mutate(seq_id = str_id)

  input_baseline = bind_rows(input_baseline, this_input_baseline)

  # X fifth cycle
  cycle_5 = cycles_this_stretch %>% filter(rel_cycle_nb == 5)
  X_cycle_5 =
    X %>%
    filter(seq_id == cycles_this_stretch$seq_id[1],
           t >= min(cycle_5$cycle_start),
           t <= max(cycle_5$cycle_end)) %>%
    rename(user_id = seq_id) %>%
    mutate(seq_id = str_id)

  fifth_cycles = bind_rows(fifth_cycles, X_cycle_5)

  # output to predict
  this_output =
    cycle_5 %>%
    select(seq_id, cycle_number, cycle_length, cycle_start, cycle_end) %>%
    rename(user_id = seq_id) %>%
    mutate(seq_id = str_id)
  output = bind_rows(output, this_output)
}
```

```
## 008a7d168d2ae8ff5fafc410f31cedbff386dcba_1  015f4654d5b79b3c23dc89bf2b2742348956259f_1  0b0ff00a3bd310ab401
```

```
cat("\n")
```

5.3.2 Ovulatory cycle model

First, since all of our predictions are made for ovulatory cycles, we reduce our model to only include the states related to ovulatory cycles.

```
J = 7
marg_em_probs = R_hsmm$marg_em_probs
for(var in names(marg_em_probs)){
  if(marg_em_probs[[var]]$type == "non-par")
    marg_em_probs[[var]]$params$probs = marg_em_probs[[var]]$params$probs[,1:J]
  if(marg_em_probs[[var]]$type == "binom"){
    marg_em_probs[[var]]$params$size = marg_em_probs[[var]]$params$size[1:J]
    marg_em_probs[[var]]$params$prob = marg_em_probs[[var]]$params$prob[1:J]
  }
  if(marg_em_probs[[var]]$type == "norm"){
    marg_em_probs[[var]]$params$mean = marg_em_probs[[var]]$params$mean[1:J]
    marg_em_probs[[var]]$params$sd = marg_em_probs[[var]]$params$sd[1:J]
  }
}

# modify R_hsmm >> become P_hsmm (P for predictions)
P_hsmm = specify_hsmm(
  J = J ,
  state_names = R_hsmm$state_names[1:J],
  state_colors = R_hsmm$state_colors[1:J],
  init = R_hsmm$init[1:J],
  transition = R_hsmm$transition[1:J, 1:J],
  sojourn = R_hsmm$sojourn[1:J],
  marg_em_probs = marg_em_probs,
  censoring_probs = list(p = R_hsmm$censoring_probs$p[1:J],
```

```
                          q = R_hsmm$censoring_probs$q[, 1:J])
)
```

```
## Warning in .check_init(init, J): Initial probabilities did not sum to one. Values are normalized such that the i
```

```
## Warning in .check_transitions(transition, J): Transition matrix rows do not sum
## to 1. Values will be normalized such that the transition probabilities from any
## state sum to 1.
```

5.3.3 Training the model

For each user, we fit the model to their first four cycles.

```
fitted_model_dir = "../Data/fitted_models/"
# unlink(fitted_model_dir, recursive = TRUE)

#tic()
if(!file.exists(fitted_model_dir)){
  dir.create(fitted_model_dir)
  done = purrr::map(
    .x = 1:length(unique(input$seq_id)),
    .f = function(i){
      cat(i, "\t")
      sid = unique(input$seq_id)[i]
      trained_model =
        fit_hsmm(model = P_hsmm,
                 X = input %>% filter(seq_id == sid),
                 N0_emission = 10,
                 N0_sojourn = 0)
      save(trained_model, file = str_c(fitted_model_dir,sid,".Rdata"))
    }
  )
}
#toc() # 35 sec
```

5.4 Predicting cycle length

```
pred_file = "../Data/cycle_length_pred.feather"

if(!file.exists(pred_file)){

  # for each seq_id
  pred = purrr::map_dfr(
    .x = 1:length(unique(input$seq_id)),
    .f = function(i){
      cat(i, "\t")
      sid = unique(input$seq_id)[i]
      # we retrieve the output for this user
      this_output = output %>% filter(seq_id == sid)
      # we retrieve the fitted model for this user
      load(file = str_c(fitted_model_dir,sid,".Rdata"))
      # we retrieve the fifth cycle of this user
      this_fifth_cycle = fifth_cycles %>% filter(seq_id == sid)

      # Baseline
      # The predicted cycle length is the mean of the cycle length of the 4 cycles.
      # Uncertainty is the standard deviation over these 4 cycles.
```

```r
baseline_pred =
  input_baseline %>%
  filter(seq_id == sid) %>%
  summarize(uncertainty = sd(cycle_length),
            cycle_length = mean(cycle_length)) %>%
  mutate(diff = cycle_length - this_output$cycle_length) %>%
  full_join(., data.frame(cycleday = 1:this_output$cycle_length),
            by = character()) %>%
  mutate(model = "Baseline",
         cycleday_backward = cycleday - this_output$cycle_length - 1)

# Predict from each day of the next cycle
t_cycle_start = min(this_fifth_cycle$t)
hsmm_pred = purrr::map_dfr(
  .x = 1:this_output$cycle_length,
  .f = function(cd){
    # first we decode the 5th cycle until day cd
    dec =
      predict_states_hsmm(
      model = trained_model$model,
      X = this_fifth_cycle %>% filter(t <= t_cycle_start + cd - 1),
      ground_truth = data.frame(seq_id = sid,
                                t = t_cycle_start,
                                state = 1),
      trust_in_ground_truth = 1,
      method = "Viterbi")
    # we detect the last state transition
    last_state_transition =
      dec$state_seq %>%
      group_by(state) %>%
      summarize(t_last_transition = min(t), .groups = "drop") %>%
      arrange(-t_last_transition) %>%
      slice_head()
    # we simulate X times and we compute the mean and sd of the predicted cycle lengths
    # (we don't really simulate, we sample the sojourns from the last state transitions)
    Ns = 100
    hsmm_pred_this_cd =
      purrr::map_dfr(
        .x = last_state_transition$state:7,
        .f = function(s){
          data.frame(state = s, i = 1:Ns,
                     sojourn = sample(1:length(trained_model$model$sojourn[[s]]$d),
                                      Ns, prob = trained_model$model$sojourn[[s]]$d,
                                      replace = TRUE))
        }
      ) %>%
      group_by(i) %>%
      summarize(cycle_length_from_last_transition = sum(sojourn), .groups = "drop") %>%
      mutate(cycle_length =
               cycle_length_from_last_transition +
               last_state_transition$t_last_transition -
               t_cycle_start
      ) %>%
      summarize(uncertainty = sd(cycle_length),
                cycle_length = mean(cycle_length)) %>%
      mutate(cycleday = cd, diff = cycle_length - this_output$cycle_length)

    hsmm_pred_this_cd
```

```
        }
      ) %>%
        mutate(model = "HSMM",
               cycleday_backward = cycleday - this_output$cycle_length - 1)


      this_seq_pred = bind_rows(baseline_pred, hsmm_pred) %>%
        mutate(seq_id = sid)
      this_seq_pred
    }
  )

  write_feather(pred, path = pred_file)

}else{
  pred = read_feather(path = pred_file)
}


# ggplot(this_seq_pred, aes(x = cycleday_backward, y = diff, col = model)) +
#   geom_hline(yintercept = 0, col = "black")+
#   geom_line() +
#   geom_ribbon(aes(ymin = diff - uncertainty, ymax = diff + uncertainty, fill = model),
# col = NA, alpha = 0.5)
```

5.4.1 Results

```
# ggplot(pred %>% filter(model == "HSMM", cycleday_backward >= -25),
# aes(x = cycleday_backward, y = diff, col = model, group = seq_id))+
#   geom_line(alpha = 0.1) + geom_hline(yintercept = 0, linetype = 2)

ggplot(pred %>%
         filter(model == "HSMM", cycleday_backward >= -25) %>%
         mutate(diff_bin = round(diff)) %>%
         group_by(cycleday_backward, diff_bin, model) %>%
         summarize(n = n(), .groups = "drop") ,
       aes(x = cycleday_backward, y = diff_bin, fill = n))+
  geom_tile() +
  scale_fill_gradient(low = "white", high = "steelblue") +
  expand_limits(fill = 0) +
  ylim(c(-12,12)) + geom_hline(yintercept = 0, linetype = 2)
```

```
## Warning: Removed 7 rows containing missing values (geom_tile).
```

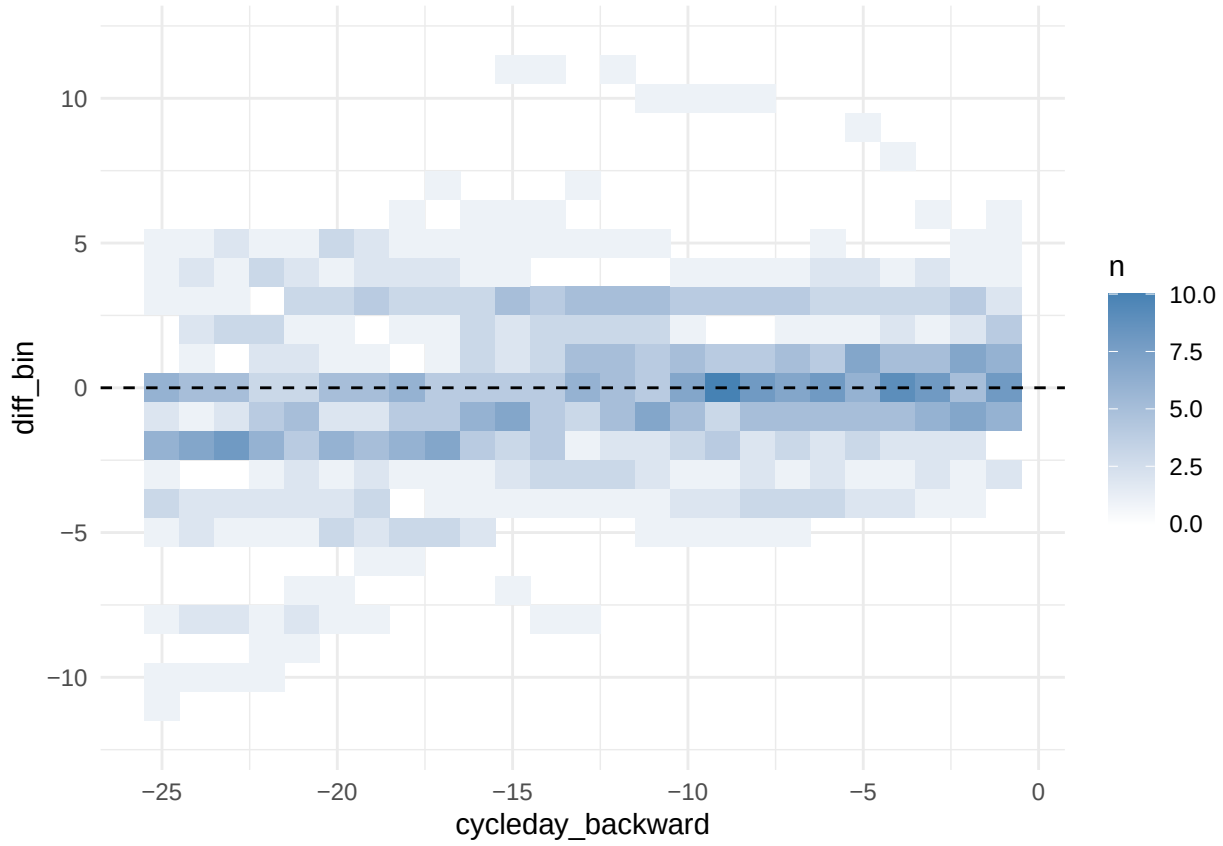


Figure 5.1: Distribution of the errors on the cycle length prediction from the HSMM simulations as one progresses through the cycles.

```
ggplot(pred %>% filter(cycleday_backward %in% c(-25, -10)),
       aes(x = diff, fill = model)) +
  geom_density(alpha = 0.3, col = NA)+
  facet_grid(cycleday_backward ~ .)
```

Table 5.1: MSE on the cycle length prediction for both model when made at the cycle start or 10 days before the next period.

Prediction day	MSE Baseline	MSE HSMM
At cycle start	18.91	17.89
After ovulation (10 days before next cycle)	18.91	7.96

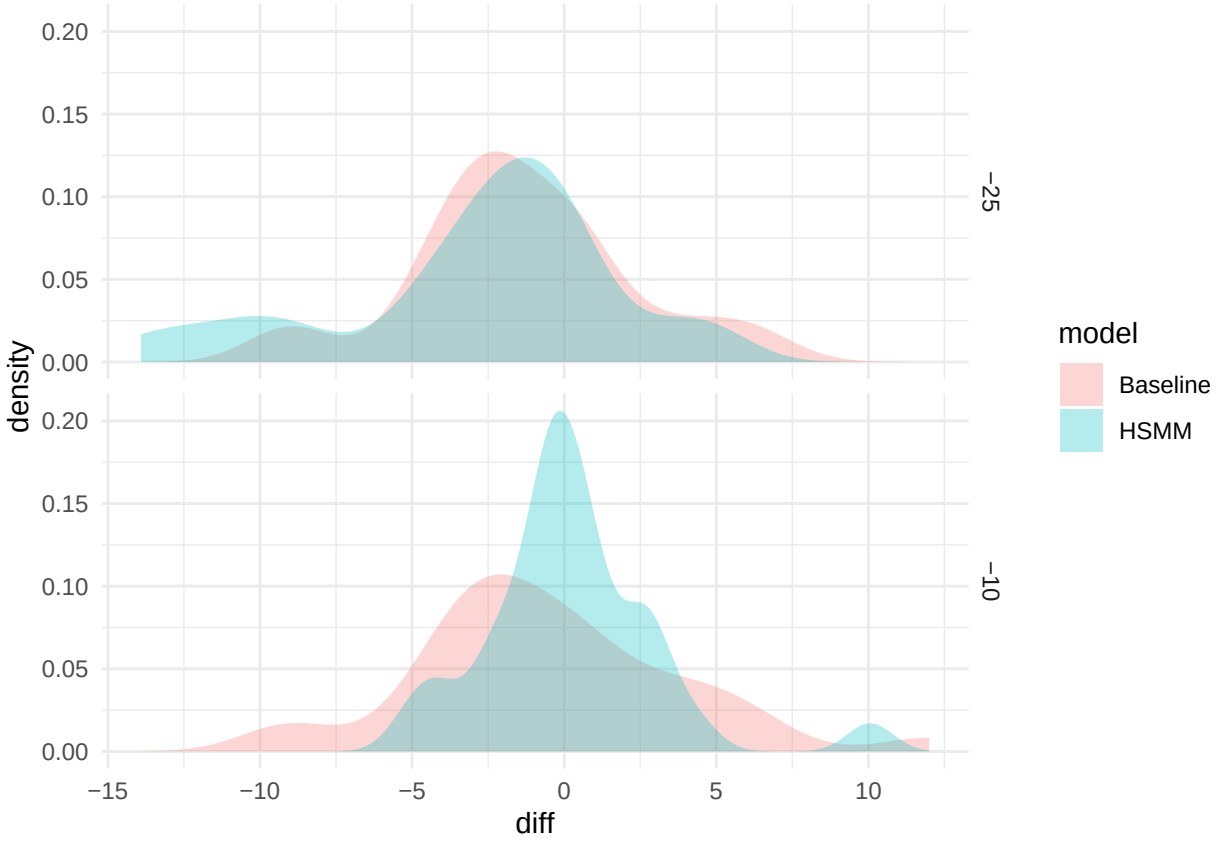


Figure 5.2: Comparison of the distributions of the error on the cycle length prediction between the HSMM and the baseline method.

```
pred_cycleday = c("At cycle start", "After ovulation\n(10 days before next cycle)")
pred_cycleday = factor(pred_cycleday, levels = pred_cycleday)

pred_res_table =
  pred %>%
  filter(cycleday_backward == -10 | cycleday == 1) %>%
  mutate(pred_cycleday = pred_cycleday[(cycleday != 1)+1]) %>%
  group_by(pred_cycleday, model) %>%
  summarize(MSE = mean(diff^2), .groups = "drop") %>%
  pivot_wider(names_from = model, values_from = MSE, names_prefix = "MSE ") %>%
  rename(`Prediction day` = pred_cycleday)

save(pred_res_table, file = "../Data/cycle_length_pred_summary_table.Rdata")

kable(pred_res_table, digits = 2, caption = "MSE on the cycle length prediction for both model when made at t
```

5.5 Predicting observations over the next days

```
predictions_dir = "../Data/obs_predictions/"
if(!dir.exists(predictions_dir)) dir.create(predictions_dir)

preds_with_fitted_model =
  predict_obs(input = input, fifth_cycle = fifth_cycles, model = "fitted",
              file = str_c(predictions_dir,"predictions_with_fitted_model.feather"),
              reset = FALSE)
# 2 min

preds_with_specified_model =
  predict_obs(input = input, fifth_cycle = fifth_cycles, model = "specified",
              file = str_c(predictions_dir,"predictions_with_specified_model.feather"),
              reset = FALSE)
# 2 min

preds_baseline =
  predict_obs(input = input, fifth_cycle = fifth_cycles, model = "baseline",
              file = str_c(predictions_dir,"predictions_baseline.feather"),
              reset = FALSE)
# 4 sec
```

5.5.1 Comparing predictions from the three models

```
preds =
  bind_rows(
    preds_baseline %>%
      select(seq_id, t, bleeding, mucus, temp, LH, preg) %>%
      mutate(model = "baseline"),
    preds_with_fitted_model %>%
      select(seq_id, t, bleeding, mucus, temp, LH, preg) %>%
      mutate(model = "fitted HSMM"),
    preds_with_specified_model %>%
      select(seq_id, t, bleeding, mucus, temp, LH, preg) %>%
      mutate(model = "specified HSMM")
  )
```

Missingness

```
fifth_cycles =
  fifth_cycles %>%
  arrange(seq_id, t) %>%
  group_by(seq_id) %>%
  mutate(cycle_length = n(),
         cycleday_fw = row_number(),
         cycleday_bw = row_number() - cycle_length - 1,
         cycleday = ifelse(cycleday_bw >= -18, cycleday_bw, cycleday_fw)) %>%
  ungroup()
```

```
preds_missing =
  preds %>%
  mutate(across(.cols = c(bleeding, mucus, temp, LH, preg), .fns = is.na)) %>%
  pivot_longer(
    cols = c(bleeding, mucus, temp, LH, preg),
    names_to = "var",
    values_to = "is_missing"
  ) %>%
  group_by(seq_id, t, model, var) %>%
  summarize(p_missing = mean(is_missing), .groups = "drop") %>%
```

```
  left_join(fifth_cycles %>%
              select(-user_id) %>%
              mutate(across(.cols = c(bleeding, mucus, temp, LH, preg), .fns = is.na)) %>%
              pivot_longer(
                cols = c(bleeding, mucus, temp, LH, preg),
                names_to = "var",
                values_to = "is_missing"
              ),
            by = c("seq_id", "t","var")) %>%
  mutate(error = ifelse(is_missing, 1-p_missing, p_missing))


ggplot(preds_missing %>% filter( cycleday %in% -18:10),
       aes(x = cycleday %>% factor(., levels = c(1:10,-18:-1)),
           fill = model, col = model, y = error)) +
  geom_boxplot(alpha = 0.5) +
  facet_grid(var ~ .) +
  xlab("cycleday") + ylab("error on missingness")
```

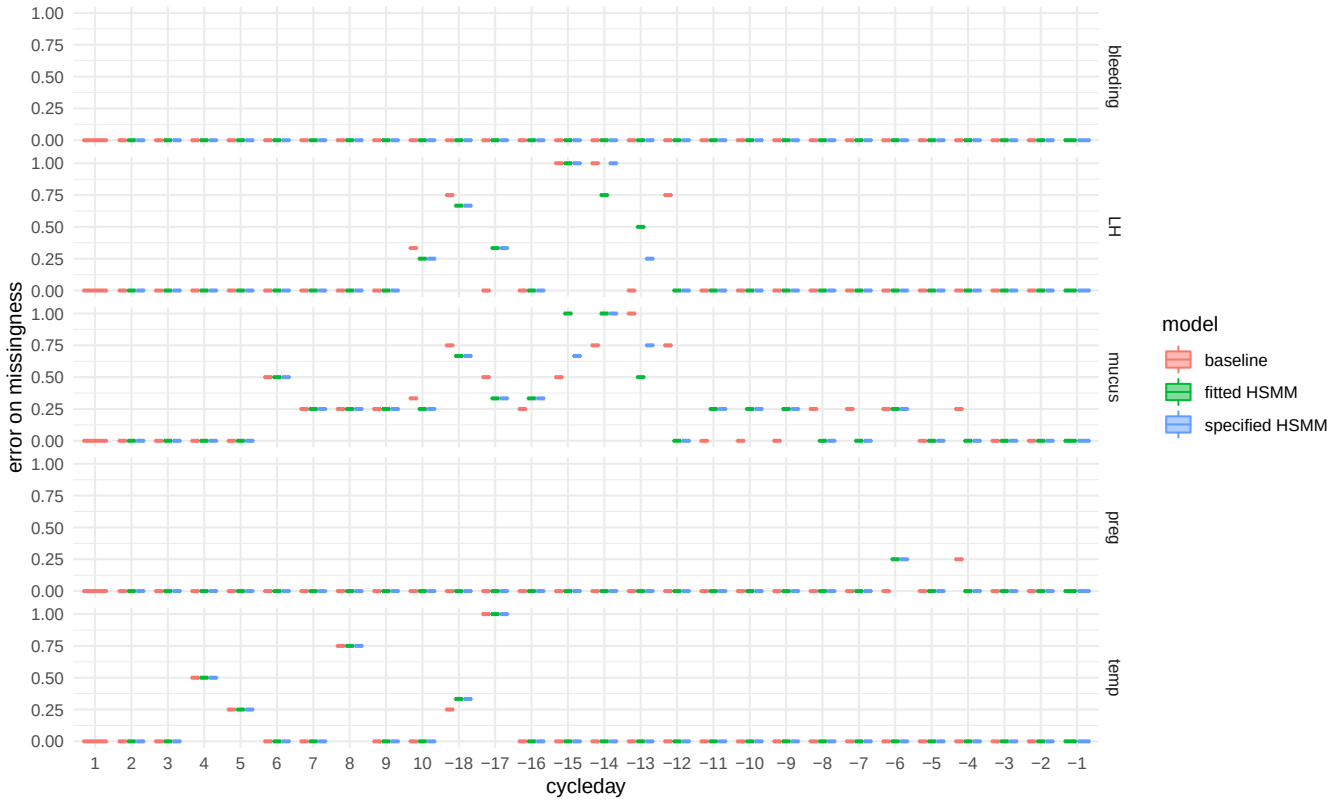


Figure 5.3: Predicting next day observations: error on missingness prediction.

Values

```
preds_values =
  preds %>%
  mutate(bleeding = factor(bleeding, levels = c("none","spotting","light","medium","heavy")) %>% as.numeric()
  rename(pred_bleeding = bleeding,
         pred_mucus = mucus,
         pred_temp = temp,
         pred_LH = LH,
         pred_preg = preg) %>%
```

```r
  left_join(
    .,
    fifth_cycles %>%
      mutate(bleeding = factor(bleeding, levels = c("none","spotting","light","medium","heavy")) %>% as.numer
      rename(
        actual_bleeding = bleeding,
        actual_mucus = mucus,
        actual_temp = temp,
        actual_LH = LH,
        actual_preg = preg),
    by = c("seq_id","t")
  ) %>%
  mutate(error_bleeding = abs(actual_bleeding - pred_bleeding),
         error_mucus = (actual_mucus != pred_mucus)*1,
         error_temp = (actual_temp - pred_temp)^2,
         error_LH = (actual_LH != pred_LH)*1,
         error_preg = (actual_preg != pred_preg)*1
  ) %>%
  select(seq_id, t, cycleday, model, starts_with("error_")) %>%
  pivot_longer(
    cols =  starts_with("error_"),
    names_to = "var",
    values_to = "error",
    names_prefix = "error_"
  ) %>%
  filter(!is.na(error)) %>%
  group_by(seq_id, t, cycleday, model, var) %>%
  mutate(MSE = mean(error)) %>%
  ungroup()


ggplot(preds_values %>% filter( cycleday %in% -16:12, var != "preg") %>%
         mutate(
           cycle_phase =
             case_when(cycleday %in% 1:6 ~ "menses",
                       cycleday %in% 7:12 ~ "follicular phase",
                       cycleday %in% -16:-12 ~ "peri-ovulatory phase",
                       cycleday %in% -12:-1 ~ "luteal phase") %>%
             factor(., levels = c("menses","follicular phase","peri-ovulatory phase","luteal phase")),
           cycleday = cycleday %>% factor(., levels = c(1:12,-16:-1))
         ),
       aes(x = cycleday, col = model, fill = model, y = MSE)) +
  geom_boxplot(varwidth = FALSE, outlier.size = 0.5, alpha = 0.5) +
  facet_grid(var ~ cycle_phase, scales = "free", space = "free")
```

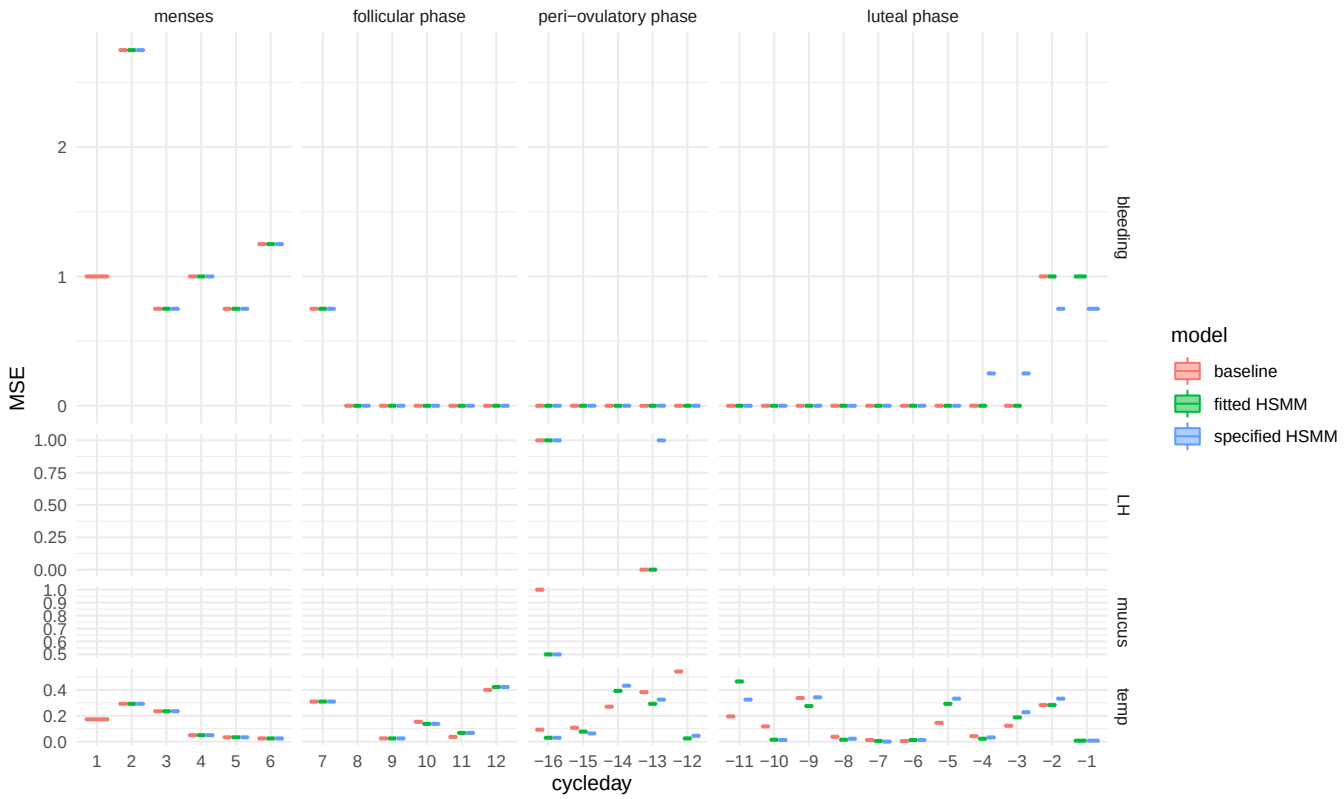


Figure 5.4: Predicting next day observations: MSE on value prediction

```
preds_values %>% group_by(var, model) %>% summarize(mean_MSE = mean(MSE)) %>%
  pivot_wider(id_cols = var, names_from = model, values_from = mean_MSE)  %>% kable(.)
```

```
## `summarise()` has grouped output by 'var'. You can override using the `.groups` argument.
```

var	baseline	fitted HSMM	specified HSMM
bleeding	0.3189655	0.3217391	0.3245614
LH	0.3333333	0.4285714	0.7500000
mucus	1.0000000	0.7500000	0.7500000
temp	0.1638460	0.1591345	0.1658252

6 Learning within-state dependencies

In this section, we show how our model is able to learn within-state dependencies between variables and to accurately decode the sequence of hidden states when the only difference between the two states is the direction of the correlation between the variables.

To show this, we proceed as follows. We first specify a two-state HSMM decoding time-series of two continuous variables whose marginal emission probabilities are identical in the two states. At specification, these two variables are independent. We then simulate a time-series from this specified model. We call this time-series X1. The next step consists in introducing within-state correlation between the variables in this time-series. The modified time-series is named X2. We can now train the specified model on this modified time-series. Finally, we decode the modified time-series with the fitted model and compare the decoding accuracy with the decoding with the specified (not fitted) model.

Model specification

```
m = specify_hsmm(J = 2,
                 init = c(1,0),
                 transition = matrix(c(0,1,1,0), nrow = 2, ncol = 2),
                 sojourn = list(type = "gamma", shape = c(10,10), scale = c(1,1)),
                 marg_em_probs = list(
                   var1 = list(type = "norm", params = list(mean = c(0,0), sd = c(1,1))),
                   var2 = list(type = "norm", params = list(mean = c(0,0), sd = c(5,5)))
                 ),
                 state_names = c("A","B"),
                 state_colors = c("lightskyblue2", "lightskyblue4"))
```

```
X1 = simulate_hsmm(model = m, n_state_transitions = 20)
plot_hsmm_seq(X1, model = m, add_state_color_legend = TRUE)
```

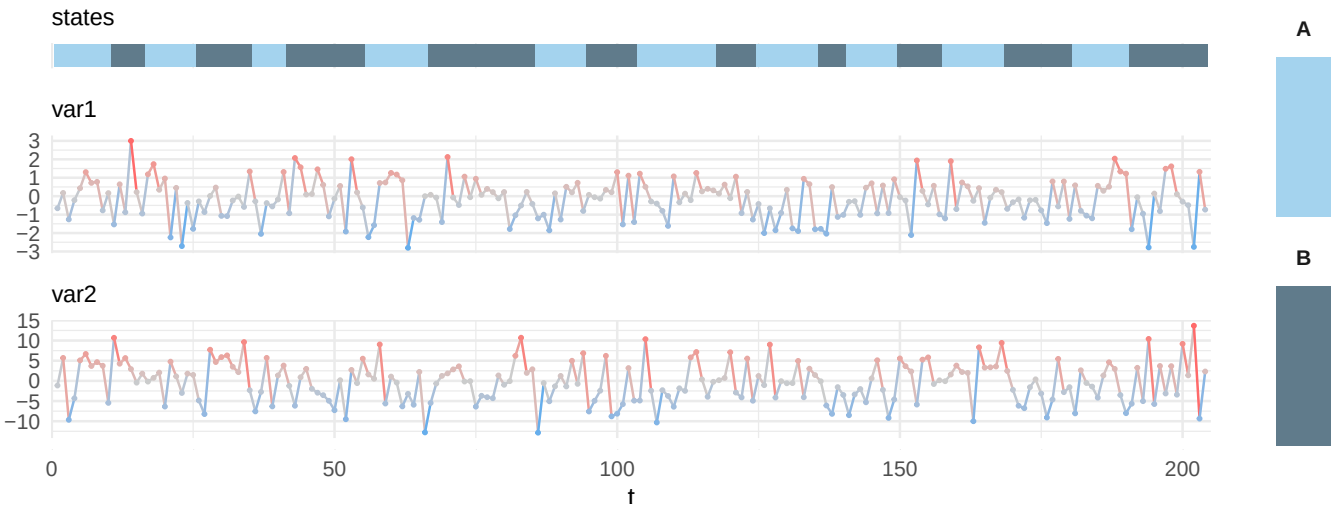


Figure 6.1: Simulated data with the specified model.

Modifying time-series to introduce within-state dependencies

```
X2 = X1
var1_state1 = sort(X2$var1[X2$state == 1])
var2_state1 = sort(X2$var2[X2$state == 1])
j = sample(1:length(var1_state1))
var1_state1 = var1_state1[j]
var2_state1 = var2_state1[j]

var1_state2 = sort(X2$var1[X2$state == 2])
var2_state2 = sort(X2$var2[X2$state == 2], decreasing = TRUE)
```

```
j = sample(1:length(var1_state2))
var1_state2 = var1_state2[j]
var2_state2 = var2_state2[j]

X2$var1[X2$state == 1] = var1_state1
X2$var2[X2$state == 1] = var2_state1
X2$var1[X2$state == 2] = var1_state2
X2$var2[X2$state == 2] = var2_state2
```

```
plot_hsmm_seq(X2, model = m, add_state_color_legend = TRUE)
```

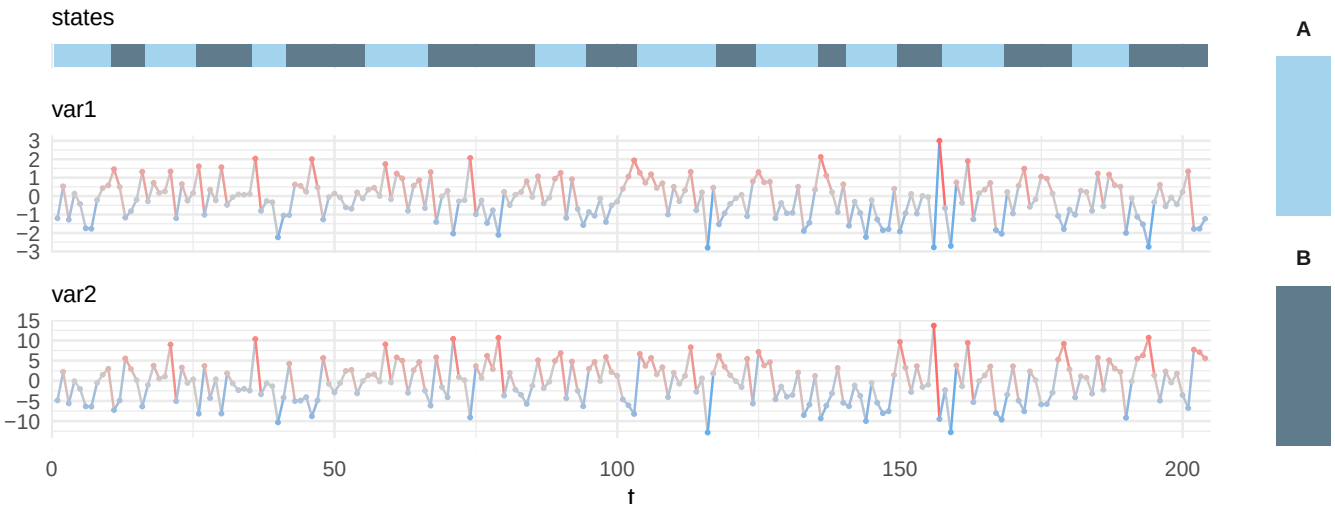


Figure 6.2: Modified data to introduce within-state correlations.

```
Xb = bind_rows(
  X1 %>%  mutate(data_type = "simulated data"),
  X2 %>%  mutate(data_type = "modified data")
) %>%
  mutate(state = factor(state),
         data_type = data_type %>% factor(., levels = c("simulated data","modified data")))


ggplot(Xb, aes(x = var1, y = var2, col = state))+
  geom_point() + scale_color_manual(values = m$state_colors) +
  facet_grid( . ~ data_type)
```

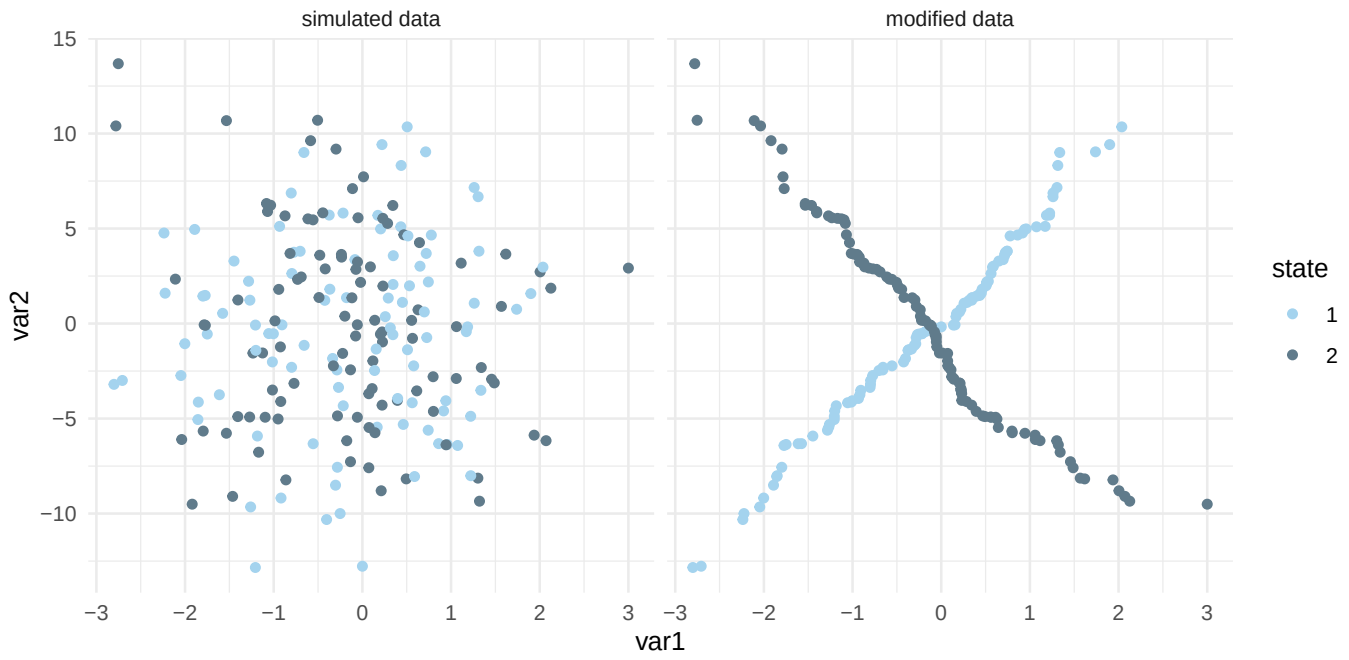


Figure 6.3: Scatter-plot between the two variables of the original and modified time-series.

Fitting the model to the modified time-series

```
model_fit = fit_hsmm(model = m, X = X2, rel_tol = 1/10000)
plot_hsmm_fit_status(fit_output = model_fit)
```

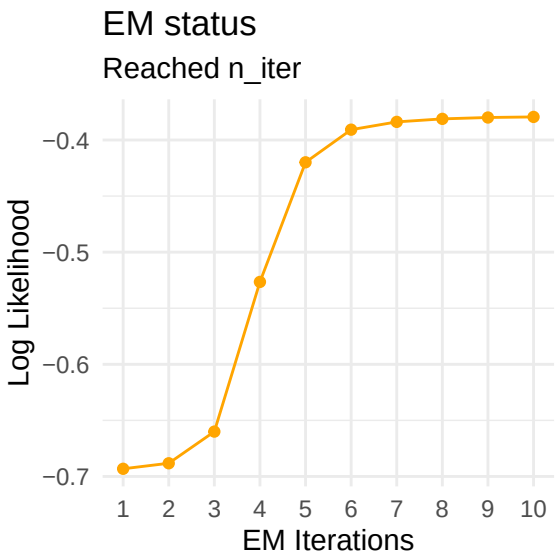


Figure 6.4: Fitting the specified model to the modified data: likelihood at each EM iteration.

Predicting sequence of hidden states with both models: the specified one and the fitted one

```
vit_original = predict_states_hsmm(model = m, X = X2, method = "Viterbi")

vit_fit = predict_states_hsmm(model = model_fit$model, X = X2, method = "Viterbi")
```

Performances

Decoding accuracy on the modified data with the specified model: 0.4656863

Decoding accuracy on the modified data with the fitted model: 0.9803922

```r
b_original = m$b %>% filter(!is.na(var1), !is.na(var2)) %>%
  pivot_longer(col = starts_with("p_"),
               names_to = "state", values_to = "prob",
               names_prefix = "p_")

b_fit = model_fit$model$b %>% filter(!is.na(var1), !is.na(var2))%>%
  pivot_longer(col = starts_with("p_"),
               names_to = "state", values_to = "prob",
               names_prefix = "p_")

b = bind_rows(
  b_original %>% mutate(model = "specified") %>% group_by(state) %>%
    mutate(rel_prob = prob/max(prob)),
  b_fit %>% mutate(model = "fitted") %>% group_by(state) %>%
    mutate(rel_prob = prob/max(prob))
) %>%
  mutate(model = model %>% factor(., levels = c("specified","fitted"))) %>%
  ungroup()

ggplot(b, aes(x = var1, y = var2, fill = rel_prob))+
  geom_tile() +
  coord_fixed() +
  facet_grid(model ~ state, labeller = "label_both") +
  scale_fill_gradient(low = "white", high = "midnightblue") +
  #scale_fill_gradient(low = "black", high = "mediumspringgreen") +
  theme(axis.text.x = element_text(angle = 90, hjust = 1),
        strip.text = element_text(face = 2)) +
  guides(fill = FALSE)
```

```
## Warning: `guides(<scale> = FALSE)` is deprecated. Please use `guides(<scale> =
## "none")` instead.
```

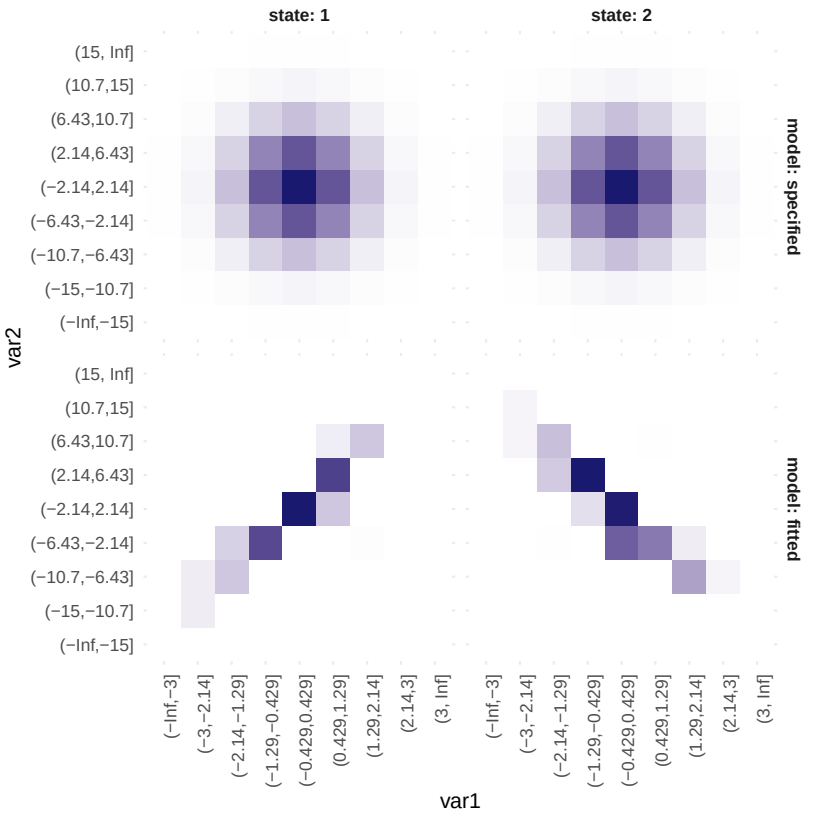


Figure 6.5: Joint emission probability densities per state and model (normalized densities).

7 Reproducibility receipt

```
## Execution datetime:

## [1] "2021-07-27 10:47:15 PDT"

## ------------------------------

## sessionInfo :

## R version 4.0.5 (2021-03-31)
## Platform: x86_64-apple-darwin17.0 (64-bit)
## Running under: macOS Big Sur 10.16
##
## Matrix products: default
## BLAS:   /Library/Frameworks/R.framework/Versions/4.0/Resources/lib/libRblas.dylib
## LAPACK: /Library/Frameworks/R.framework/Versions/4.0/Resources/lib/libRlapack.dylib
##
## locale:
## [1] en_US.UTF-8/en_US.UTF-8/en_US.UTF-8/C/en_US.UTF-8/en_US.UTF-8
##
## attached base packages:
## [1] stats4    stats     graphics  grDevices utils     datasets  methods
## [8] base
##
## other attached packages:
##  [1] viridis_0.6.1          viridisLite_0.4.0      HiddenSemiMarkov_0.1.0
##  [4] sn_2.0.0               magrittr_2.0.1         ggthemes_4.2.4
##  [7] forcats_0.5.1          stringr_1.4.0          dplyr_1.0.7
## [10] purrr_0.3.4            readr_1.4.0            tidyr_1.1.3
## [13] tibble_3.1.2           tidyverse_1.3.1        feather_0.3.5
## [16] kableExtra_1.3.4       styler_1.4.1           bookdown_0.22
## [19] knitr_1.33             shiny_1.6.0            igraph_1.2.6
## [22] tictoc_1.0.1           ggpubr_0.4.0           ggplot2_3.3.5
## [25] cowplot_1.1.1          data.table_1.14.0
##
## loaded via a namespace (and not attached):
##  [1] fs_1.5.0             lubridate_1.7.10     webshot_0.5.2
##  [4] httr_1.4.2           numDeriv_2016.8-1.1  tools_4.0.5
##  [7] backports_1.2.1      utf8_1.2.1           R6_2.5.0
## [10] lazyeval_0.2.2       DBI_1.1.1            colorspace_2.0-2
## [13] withr_2.4.2          gridExtra_2.3        tidyselect_1.1.1
## [16] mnormt_2.0.2         curl_4.3.2           compiler_4.0.5
## [19] cli_3.0.0            rvest_1.0.0          network_1.17.1
## [22] xml2_1.3.2           plotly_4.9.4.1       scales_1.1.1
## [25] systemfonts_1.0.2    digest_0.6.27        foreign_0.8-81
## [28] rmarkdown_2.9        svglite_2.0.0        rio_0.5.27
## [31] pkgconfig_2.0.3      htmltools_0.5.1.1    dbplyr_2.1.1
## [34] fastmap_1.1.0        htmlwidgets_1.5.3    rlang_0.4.11
## [37] readxl_1.3.1         rstudioapi_0.13      generics_0.1.0
## [40] jsonlite_1.7.2       statnet.common_4.5.0 zip_2.2.0
## [43] car_3.0-11           Rcpp_1.0.6           munsell_0.5.0
## [46] fansi_0.5.0          abind_1.4-5          lifecycle_1.0.0
## [49] geomnet_0.3.1        stringi_1.6.2        yaml_2.2.1
## [52] carData_3.0-4        grid_4.0.5           promises_1.2.0.1
## [55] crayon_1.4.1         lattice_0.20-44      haven_2.4.1
## [58] hms_1.1.0            sna_2.6              tmvnsim_1.0-2
## [61] pillar_1.6.1         ggsignif_0.6.2       codetools_0.2-18
## [64] reprex_2.0.0         glue_1.4.2           evaluate_0.14
## [67] modelr_0.1.8         vctrs_0.3.8          httpuv_1.6.1
```

```
## [70] cellranger_1.1.0     gtable_0.3.0         assertthat_0.2.1
## [73] xfun_0.24            openxlsx_4.2.4       mime_0.11
## [76] xtable_1.8-4         broom_0.7.8          coda_0.19-4
## [79] rstatix_0.7.0        later_1.2.0          ellipsis_0.3.2
```

References

1. I. S. Fraser, H. O. D. Critchley, M. Broder, M. G. Munro, The FIGO recommendations on terminologies and definitions for normal and abnormal uterine bleeding. *Seminars in Reproductive Medicine*. **29,** 383–390 (2011).

2. C. J. Munro, G. H. Stabenfeldt, J. R. Cragun, L. A. Addiego, J. W. Overstreet, B. L. Lasley, Relationship of serum estradiol and progesterone concentrations to the excretion profiles of their major urinary metabolites as measured by enzyme immunoassay and radioimmunoassay. *Clinical Chemistry*. **37,** 838–844 (1991).

3. L. Faust, D. Bradley, E. Landau, K. Noddin, L. V. Farland, A. Baron, A. Wolfberg, Findings from a mobile application–based cohort are consistent with established knowledge of the menstrual cycle, fertile window, and conception. *Fertility and Sterility*. **112,** 450–457.e3 (2019).

4. L. Symul, K. Wac, P. Hillard, M. Salathé, Assessment of Menstrual Health Status and Evolution through Mobile Apps for Fertility Awareness. *npj Digital Medicine* (2019), doi:10.1038/s41746-019-0139-4.

5. S. D. Harlow, S. A. Ephross, Epidemiology of Menstruation and Its Relevance to Women ' s Health. *Public Health*. **17,** 265–286 (1995).

6. L. A. Cole, D. G. Ladner, F. W. Byrn, The normal variabilities of the menstrual cycle. *Fertility and Sterility*. **91,** 522–527 (2009).

7. E. A. Lenton, B. M. Landgren, L. Sexton, Normal variation in the length of the luteal phase of the menstrual cycle: identification of the short luteal phase. *British journal of obstetrics and gynaecology*. **91,** 685–9 (1984).

8. E. A. Lenton, B. M. Landgren, L. Sexton, R. Harper, Normal variation in the length of the follicular phase of the menstrual cycle: effect of chronological age. *British journal of obstetrics and gynaecology*. **91,** 681–4 (1984).

9. C. E. Malcolm, D. C. Cumming, Does anovulation exist in eumenorrheic women? *Obstetrics and Gynecology*. **102,** 317–318 (2003).

10. J. C. Prior, M. Naess, A. Langhammer, S. Forsmo, Ovulation prevalence in women with spontaneous normal-length menstrual cycles - A population-based cohort from HUNT3, Norway. *PLoS ONE*. **10,** 1–14 (2015).

11. A. J. Wilcox, D. D. Baird, C. R. Weinberg, Time of implantation of the conceptus and loss of pregnancy. *The New England journal of medicine*. **340,** 1796–9 (1999).

12. L. M. Rossen, K. A. Ahrens, A. M. Branum, Trends in Risk of Pregnancy Loss Among US Women, 1990–2011. *Paediatric and Perinatal Epidemiology*. **32,** 19–29 (2018).

13. ACOG, FAQ: Early pregnancy loss.

14. M. Delnord, J. Zeitlin, Epidemiology of late preterm and early term births – An international perspective. *Seminars in Fetal and Neonatal Medicine*. **24,** 3–10 (2019).

15. E. L. Billings, J. B. Brown, J. J. Billings, H. G. Burger, Symptoms and Hormonal Changes Accompanying Ovulation. *The Lancet*. **299,** 282–284 (1972).

16. J. R. L. Ehrenkranz, Home and point-of-care pregnancy tests: A review of the technology. *Epidemiology*. **13,** 18–22 (2002).