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

Supplementary Material

Laura Symul

01 December, 2020 - 15:47

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 [link]. 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 models specifications

1.1 Model for labelling fertility-app data with reproductive events

In this section, we specify a hidden-semi Markov model 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.

1.1.1 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 1.1 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	Ovu	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: 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: 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	0.03	0	0.95	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.

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.

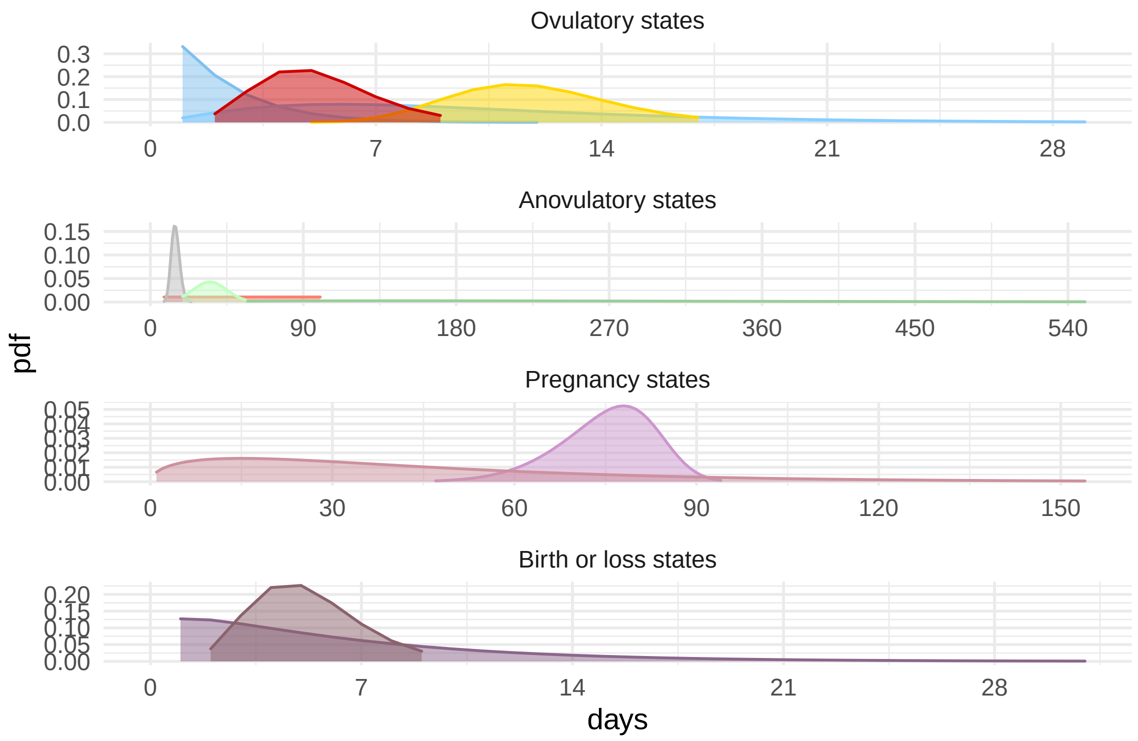


Figure 1.1: Sojourn distributions of the HSMM of reproductive events.

1.1.2 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.

```
## Warning in rm(bleeding_probs, high_bleeding, rare_spotting, frequent_spotting, :
## object 'frequent_spotting' not found
```

1.1.3 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.

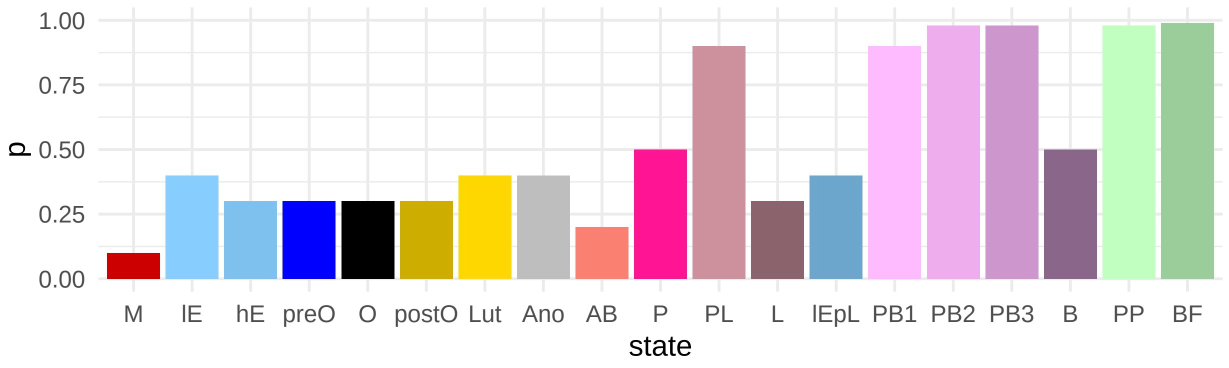


Figure 1.2: Probability of all variable beeing missing simultaneously in each state of our reproductive events HSMM for a hypothetical 'typical' user.

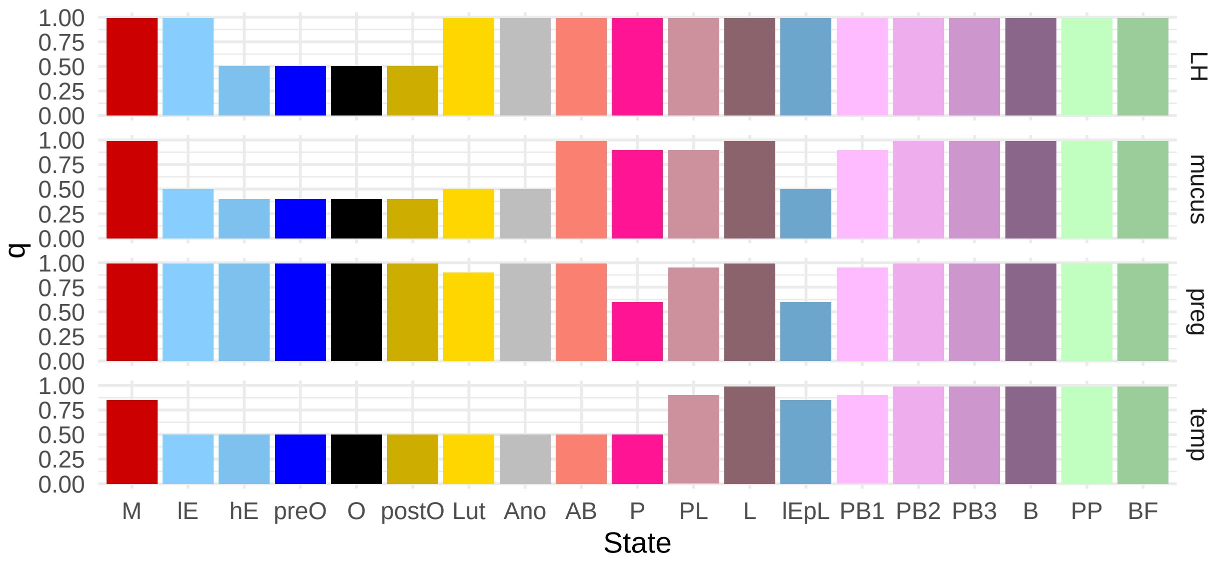


Figure 1.3: Probabilities that a hypothetical 'typical' user would not report a specific variable in each state of our reproductive events HSMM.

1.1.4 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)
```

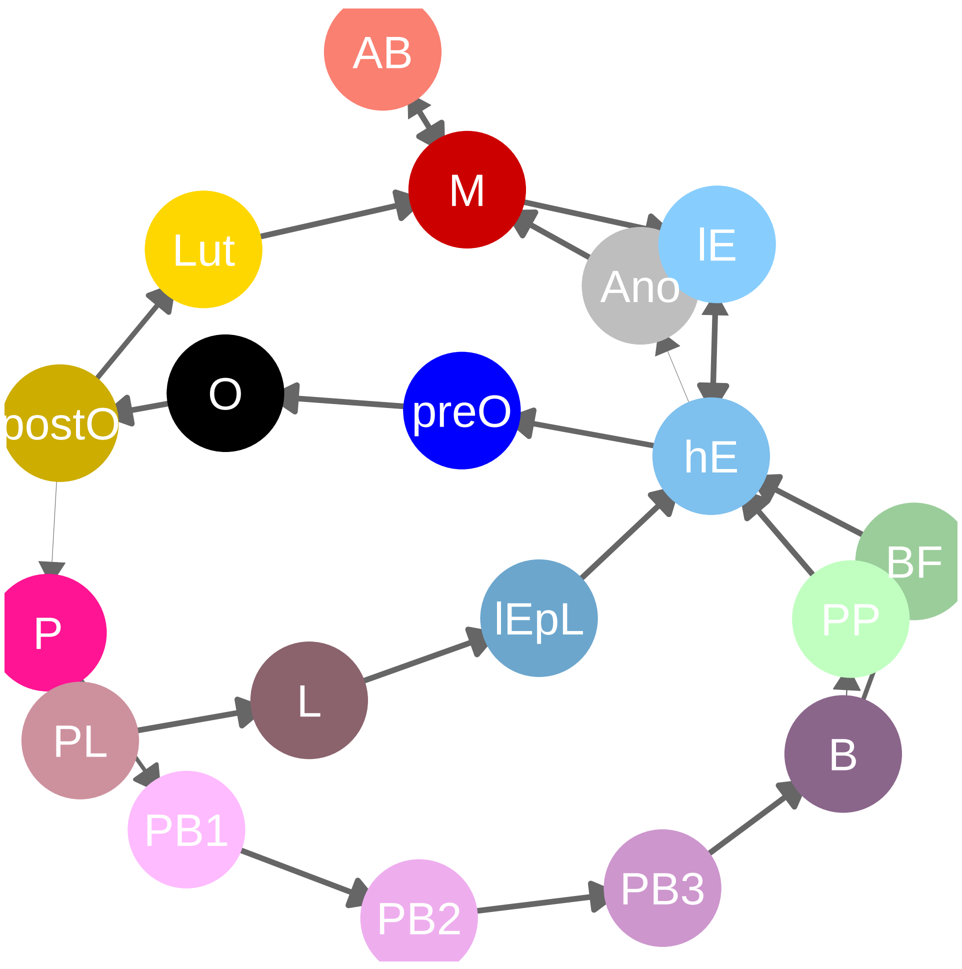


Figure 1.4: Graph of our reproductive events HSMM. The width of the edges are proportional to the transition probabilities between states.

```
plot_hsmm_marg_dist(model = R_hsmm)
```

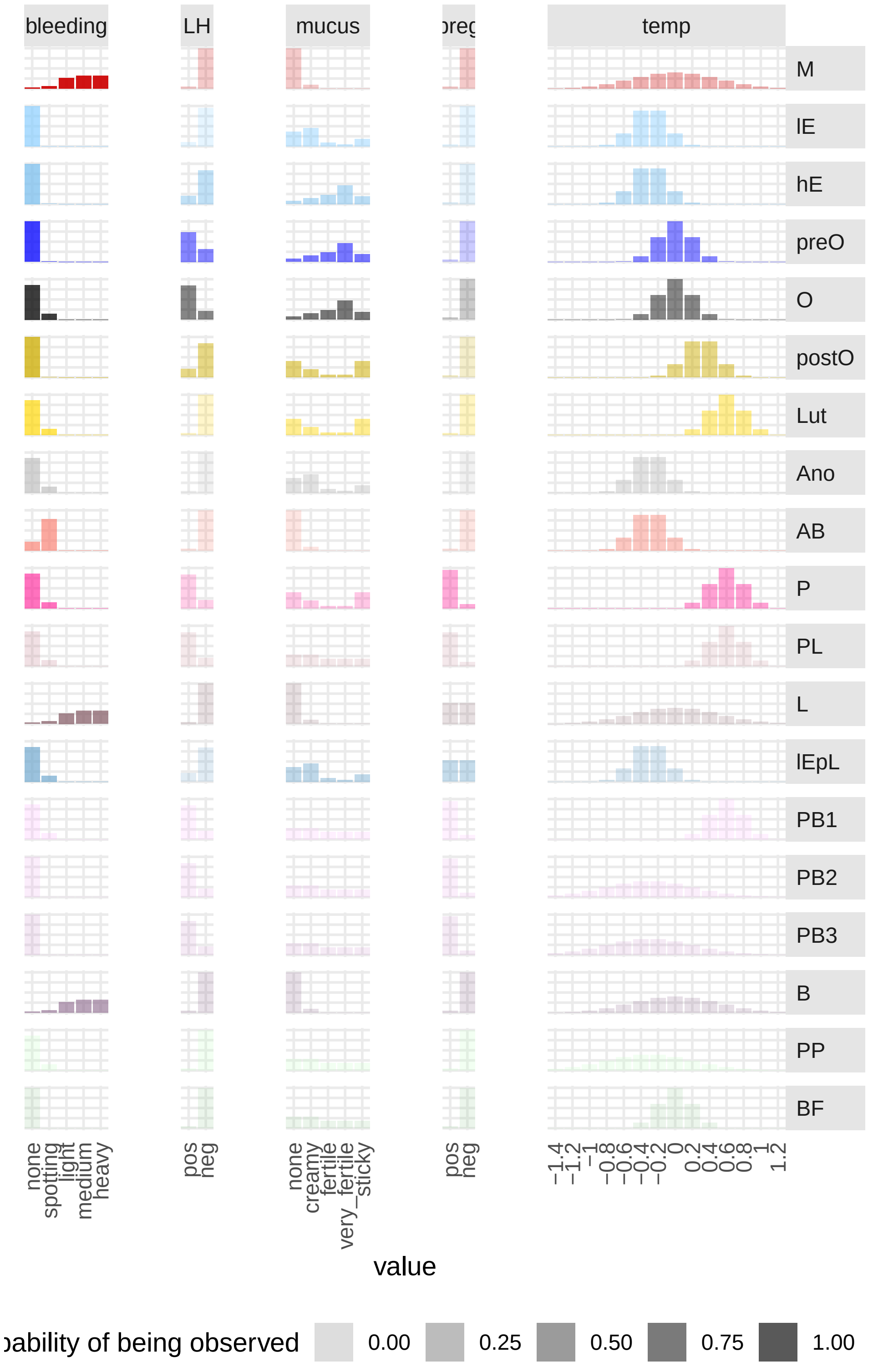
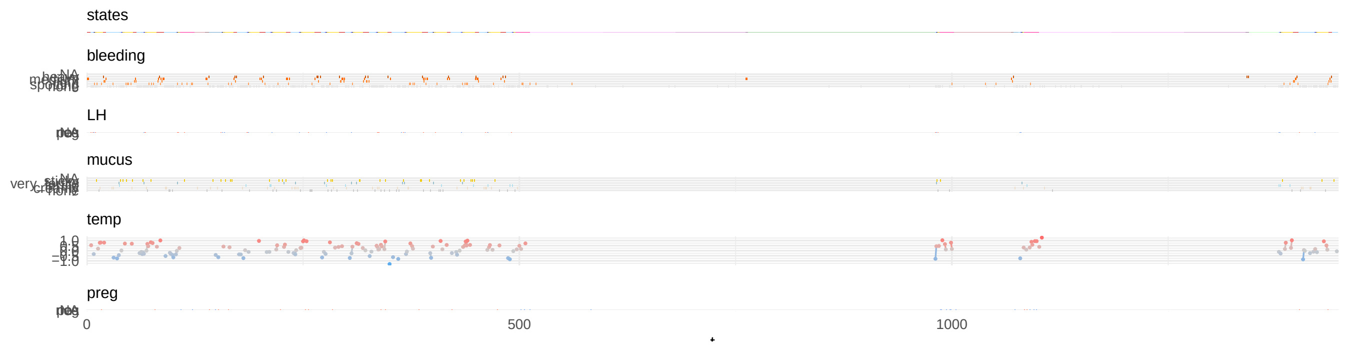


Figure 1.5: Marginal emission probabilities of our reproductive events HSMM.

1.1.5 Simulate a sequence

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

plot_hsmm_seq(X, model = R_hsmm)
```



```
#plot_hsmm_seq(X, model = R_hsmm, compact_view = TRUE)
```

1.1.6 Save model

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

1.2 Model to identify periods from bleeding patterns

The menses-identifying model (`M_hsmm`) is a simplification of the `R_hsmm` model.

```r
load(file = str_c("../Data/models/R_hsmm.Rdata"), verbose = FALSE)

M_hsmm = R_hsmm
# The sojourns of the hE and Lut states are fixed
M = length(M_hsmm$sojourn[[1]]$d)
M_hsmm$sojourn$Lut$d = c(rep(0,10),1,rep(0,M-11))
M_hsmm$sojourn$hE$d = c(rep(0,2),1,rep(0,M-3))

# The transitions from hE to lE and from lE to Ano are prevented
M_hsmm$transition["hE","lE"] = 0
M_hsmm$transition[,"Ano"] = 0
M_hsmm$transition = M_hsmm$transition / rowSums(M_hsmm$transition )
# The model decodes time-series of bleeding reports
M_hsmm$marg_em_probs = list(bleeding = M_hsmm$marg_em_probs$bleeding)

M_hsmm = specify_hsmm(J = M_hsmm$J,
                       state_names = M_hsmm$state_names,
                       init = M_hsmm$init,
                       transition = M_hsmm$transition,
                       sojourn = M_hsmm$sojourn,
                       marg_em_probs = M_hsmm$marg_em_probs)

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

1.3 Model to identify sub-sequences in user's time-series with consistent tracking behavior

The tracking-behavior categories identifying model (`T_hsmm`) is specified to identify long sub-sequences in user's time-series in which they had a consistent tracking behavior. Since we assume, for simplicity of the reproductive event labeling process, that users change behavior around their period, this model decodes time-series of five binary variables. The first one indicates whether a period was reliably identified with the `M_hsmm` and the four other variables indicate whether mucus, temperature, pregnancy tests or LH tests were reported.

The tracking behavior categories, *i.e.* states of this model, are defined based on possible combination of reported variables. In theory, this would lead to $\sum_{k=0}^{4}\binom{4}{k} = 16$ states. However, we simplified to 6 states as some combination of reported variables had similar consequences on the biological states that could or could not be discriminated with these variables. For example, LH tests, temperature and pregnancy tests bring information which can help to discriminate between a cycle with a long follicular phase and a cycle with a short follicular phase but an early pregnancy loss.

To these 6 states, we add a seventh state which is a transition state. This transition state is a short duration state (approximately the same duration as a period) and transitions from any of the 6 other states are done via this transition state.

```r
states = c("b","bp","b_tests","no_temp","no_mucus","full", "transition")
J = length(states)
M = 10*365 # 10 years is the max sojourn

T_hsmm = specify_hsmm(
  J = J,
  state_names = states,
  state_colors = c(
    hsv(0.54, s = 0.6, v = 1),
    hsv(0.54, s = 0.8, v = 1),
    hsv(0.54, s = 0.8, v = 0.9),
    hsv(0.54, s = 0.8, v = 0.8),
    hsv(0.54, s = 0.8, v = 0.7),
    hsv(0.54, s = 0.8, v = 0.6),
    "black"
    ),  #c(rainbow(n = 6, s = 1, v = 0.7), "black"),
  init = c(rep(1, J-1)/(J-1), 0),
  transition = rbind(
    matrix(c(rep(0,J-1),1), nrow = J-1, ncol = J, byrow = TRUE),
    c(rep(1/(J-1),J-1),0)) %>% set_colnames(states) %>% set_rownames(states),
  sojourn = list(type = "ksmoothed_nonparametric",
                 d = cbind(
                   rbind(matrix(0, nrow = 9*32, ncol = J-1), matrix(1/M, nrow = M, ncol = J-1)),
                   matrix(c(rep(1/5,5),rep(0,9*32+M-5)), nrow = 9*32 + M, ncol = 1))),
  marg_em_probs = list(
                                                                                   #b      bp      b_test  no_temp
    any.menses = list(type = "binom", params = list(size = rep(1,J), prob = c(1/5,    1/5,    1/5,    1/5,
    any.preg   = list(type = "binom", params = list(size = rep(1,J), prob = c(0,      1/100,  1/100,  1/100,
    any.LH     = list(type = "binom", params = list(size = rep(1,J), prob = c(0,      1/1000, 1/60,   1/100,
    any.mucus  = list(type = "binom", params = list(size = rep(1,J), prob = c(0,      1/1000, 1/1000, 1/100,
    any.temp   = list(type = "binom", params = list(size = rep(1,J), prob = c(0,      1/2000, 1/2000, 1/2000,
  )
)

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

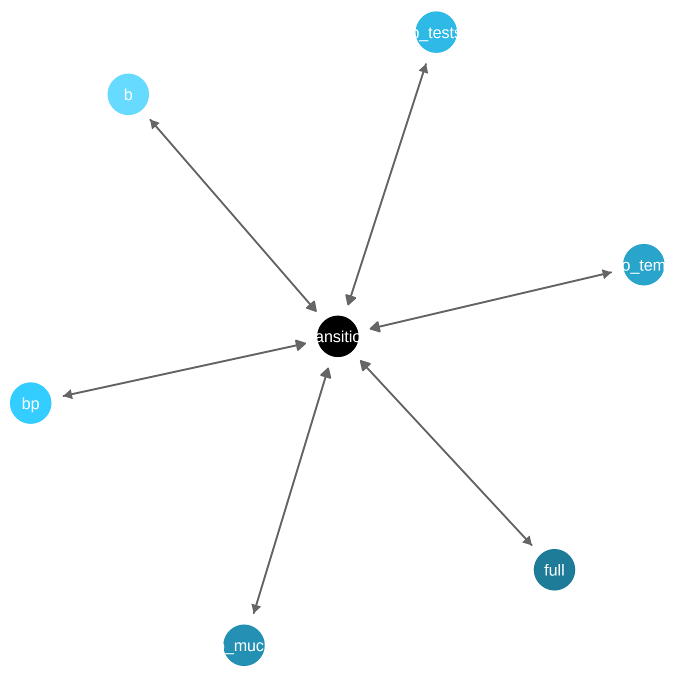


Figure 1.6: Graph of the T-hsmm model

```
Ti_sim = simulate_hsmm(model = T_hsmm, n_state_transitions = 7, seed = 2)
plot_hsmm_seq(X = Ti_sim, model = T_hsmm, add_state_color_legend = TRUE)
```

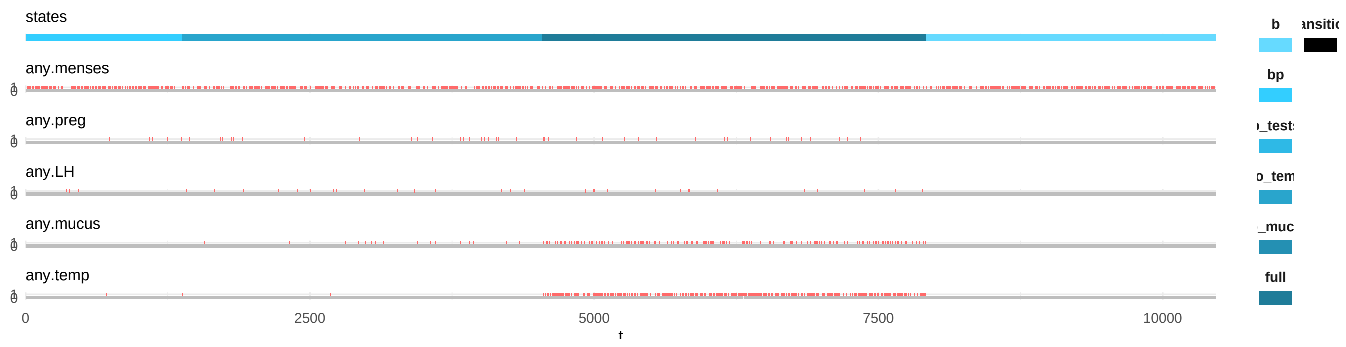


Figure 1.7: Simulated sequence with the T-hsmm.

2 Datasets

2.1 Synthetic data generation

To generate synthetic data, we first define a number of users and some characteristics defining these fictive users such as their highest tracking frequency, controlled with a parameter α, and menstrual characteristics such as the length and regularity of their follicular and luteal phases or the noise in their temperature measurements.

Because we want to evaluate the performance of our framework given different level of missing data in a realistic context, we define the effect of the parameter α on the missing probabilities as follow:

$$q^*_{j,k} = q_{j,k} - \beta\, q_{j,k} + \beta\, \delta$$

where

$q_{j,k}$ are realistic (expected in real-life data, see fig 2.1) estimates of the missing probabilities with $q_{j,k} = P(X_k \text{ is missing}|S = j)$, $\beta = \|\frac{3^\alpha-1}{3^\alpha+1}\|$ and $\delta = 0$ if $\alpha < 0$ and 1 otherwise.

The parameter α takes values in $\mathbb{R}$. When α is zero, the missing probabilities are identical to the expected missing probabilities from a typical app users whose purpose is to identify their fertile window. When α is negative, the missing probabilities are smaller than the expected ones, when alpha is positive they are bigger. In the extreme cases, a value of $-\infty$ means that the variables are never missing and a value of $+\infty$ means that the variables are always missing. When $\alpha = -1$, $q^* = \frac{1}{2}q$: the variables are missing only about half the time they were expected to be missing. When $\alpha = 1$, variables are missing with a probability of at least 50%.

Expected missingness

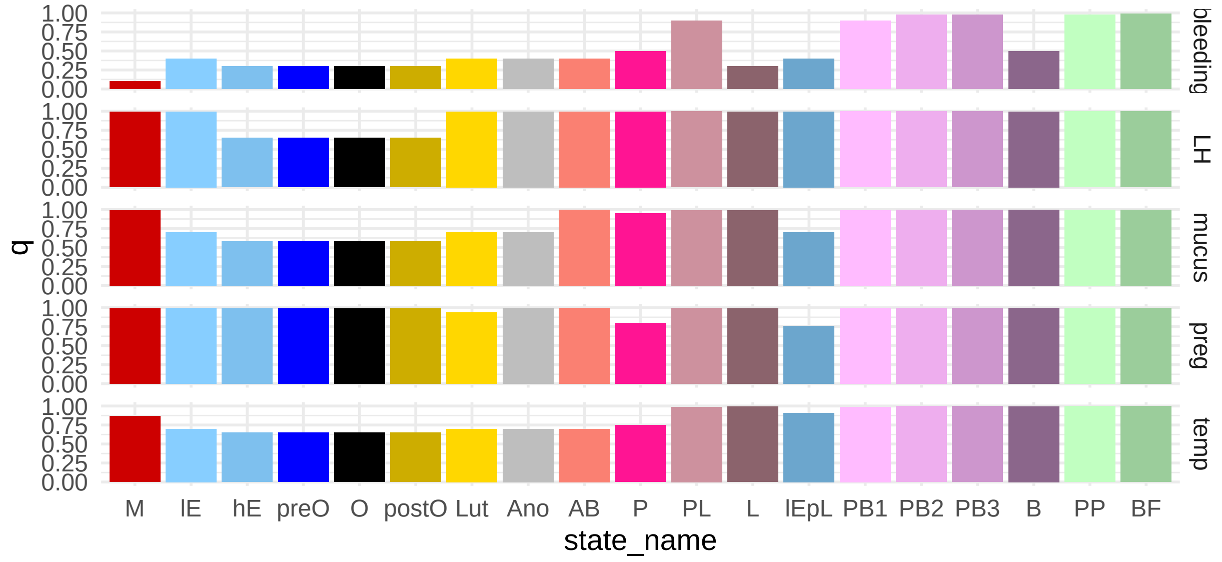


Figure 2.1: Missing probabilities for each variable and state expected for a typical users whose tracking purpose is to identify their fertile window and detect pregnancies early on.

Synthetic users characteristics

```r
N_per_alpha = 30
alpha = rep(c(-Inf, -1, -0.5, 0, 0.5, 1, 2, 4), N_per_alpha)
N_users = length(alpha)

s_users = data.frame(
  seq_id = 1:N_users,
  alpha = alpha,
  temp_sd = runif(N_users, min = 0.05, max = 0.3),
  mean_lE_sojourn = rnorm(N_users, mean = 10, sd = 3),
  sd_lE_sojourn = 0.2 + rpois(N_users, lambda = 2),
  mean_Lut_sojourn = rnorm(N_users, mean = 11, sd = 1),
  sd_Lut_sojourn = 0.2 + rpois(N_users, lambda = 0.75)
  )

rm(alpha)
```

Generating time-series

```r
# 139 sec for 64 time-series
Xsim = purrr::map_dfr(
  .x = s_users$seq_id,
  .f = function(sid){
    #cat("\n",sid, "\n")
    modified_hsmm = R_hsmm

    # modifying censoring probabilities
    alpha = s_users$alpha[s_users$seq_id == sid]
    beta = abs((3^alpha - 1)/(3^alpha + 1))
    delta = ifelse(alpha > 0, 1, 0)
    modified_hsmm$censoring_probs$p = 0 *modified_hsmm$censoring_probs$p
    modified_hsmm$censoring_probs$q = qjk * (1 - beta) + beta * delta

    # modifying emission distributions
    modified_hsmm$marg_em_probs$temp$params$sd = rep(s_users$temp_sd,R_hsmm$J)
    # modifying sojourns
    M = length(modified_hsmm$sojourn$M$d)
    modified_hsmm$sojourn$lE$d = dnorm(1:M,
                                       mean = s_users$mean_lE_sojourn,
                                       sd = s_users$sd_lE_sojourn)
    modified_hsmm$sojourn$Lut$d = dnorm(1:M,
                                        mean = s_users$mean_Lut_sojourn,
                                        sd = s_users$sd_Lut_sojourn)

    # specifying modified model
    modified_hsmm = specify_hsmm(J = modified_hsmm$J,
                                 init = modified_hsmm$init, trans = modified_hsmm$transition,
                                 sojourn = modified_hsmm$sojourn,
                                 marg_em_probs = modified_hsmm$marg_em_probs,
                                 censoring_probs = modified_hsmm$censoring_probs)

    # simulating sequence
    Xsim_i = simulate_hsmm(model = modified_hsmm, seq_id = as.character(sid), n_state_transitions = 100)
  })
```

Tracking behavior categories

```r
Xsim2  = Xsim %>%
  mutate(transitions = rbinom(nrow(Xsim), size = 1, prob = 0.001)) %>%
  group_by(seq_id) %>%
  mutate(tracking_sequence = 1+cumsum(transitions)) %>%
  group_by(seq_id, tracking_sequence) %>%
  mutate(tracking_category = sample(c("b","bp","btm","full"),1)) %>%
  mutate(temp = ifelse(tracking_category %in% c("b","bp"), NA, temp),
         mucus = ifelse(tracking_category %in% c("b","bp"), NA, mucus),
         LH = ifelse(tracking_category %in% c("b","bp","btm"), NA, LH),
         preg = ifelse(tracking_category %in% c("b","btm"), NA, preg)) %>%
  ungroup() %>%
  left_join(., s_users %>% mutate(seq_id = as.character(seq_id)), by = c("seq_id")) %>%
  select(-transitions, -tracking_sequence)
```

Table 2.1: Number of data-point in each tracking category.

tracking_category	Freq
b	61821
bp	50470
btm	54395
full	62771

```r
write_feather(Xsim2, path = "../Data/synthetic_data/Xsim.feather")
```

Visualization of a synthetic time-series example

```r
plot_hsmm_seq(X = Xsim2 %>%  filter(seq_id == 4), model = R_hsmm)
```

```
## Warning: Removed 284 row(s) containing missing values (geom_path).

## Warning: Removed 431 rows containing missing values (geom_point).
```

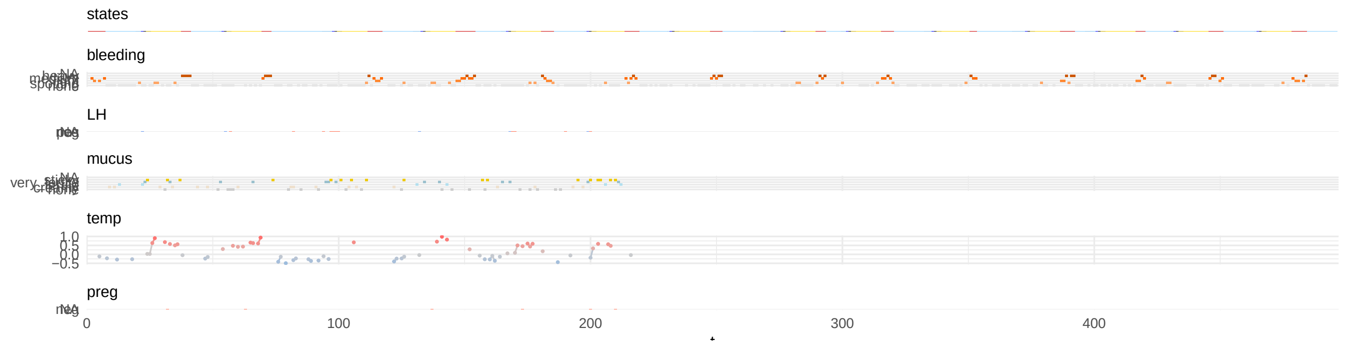


Figure 2.2: Example of a synthetic time-series.

Synthetic dataset characteristics

```r
table(tracking_category = Xsim2$tracking_category)  %>%
  kable(., format = "latex", booktabs = T,
        caption = "Number of data-point in each tracking category.")
```

```r
table(seq_id = Xsim2$seq_id) %>%
  data.frame() %>%
  set_colnames(c("seq_id","sequence_length")) %>%
  ggplot(., aes(x = sequence_length)) +
  geom_histogram(bins = 50) + expand_limits(x = 0)
```

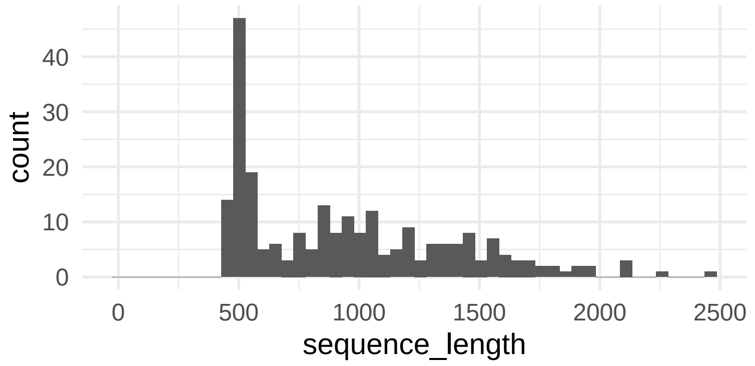


Figure 2.3: Histogram of simulated sequence length.

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"))
```

3 Decoding time-series

3.1 Decoding functions

```r
source("Scripts/00_FAM_decoding_functions.R")
```

3.2 Loading models

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

3.3 Synthetic data

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

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

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

  RES = purrr::map_dfr(.x = unique(X$seq_id),
                       .f = function(sid){
                         # cat(sid, "\n")
                         get_most_likely_sequence_with_prob(
                           X = X %>%
                             filter(seq_id == sid) %>%
                             select(-alpha, -state, -tracking_category),
                           verbose = FALSE
                         )
                       }
  )

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

3.4 App data

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

RES.file = paste0(IO$output_data, "RES_app_data.feather")

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

  tic()
  RES = purrr::map_dfr(.x = unique(X$seq_id),
                       .f = function(sid){
                         cat(sid, "\n")
                         get_most_likely_sequence_with_prob(
                           X = X %>% filter(seq_id == sid),
                           fit_models = FALSE,
                           verbose = FALSE)
```

```
                              }
  )
  toc()

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

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) %>% rename(state_GT = state), b
```

Overall accuracy

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

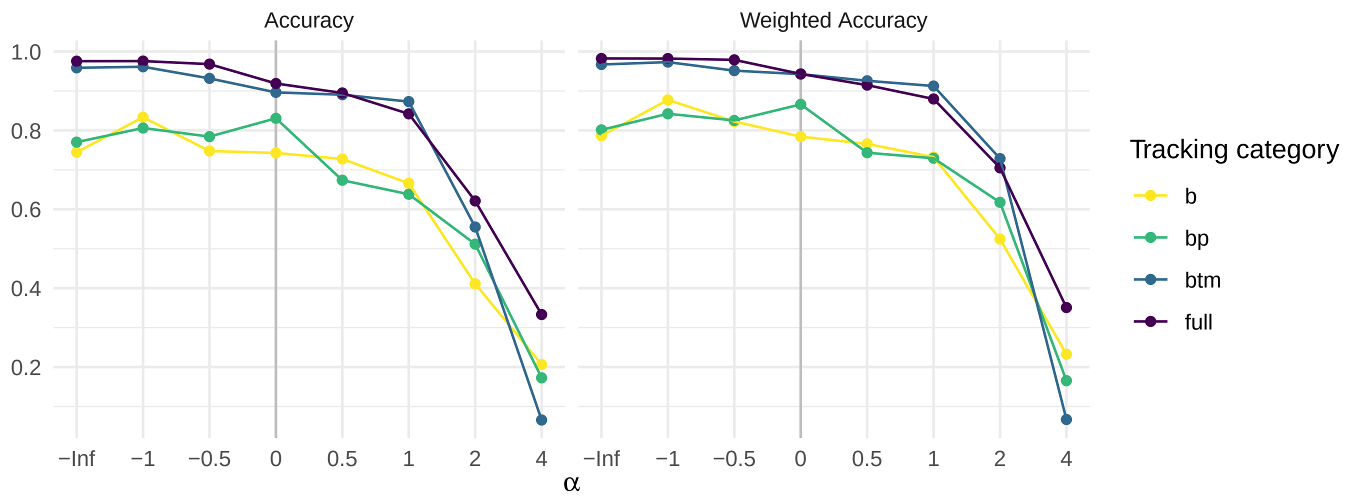


Figure 4.1: Accuracy on synthetic data.

State-specific accuracy

```
Accuracies_per_state = XP %>% group_by(alpha, tracking_category, state_GT) %>%
  summarize(Accuracy = mean(state == state_GT, na.rm = TRUE),
            Weighted_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_Accuracy"), names_to = "accuracy_type", values_to = "accuracy")
  mutate(accuracy_type = accuracy_type %>%  str_replace(., "_"," "))
```

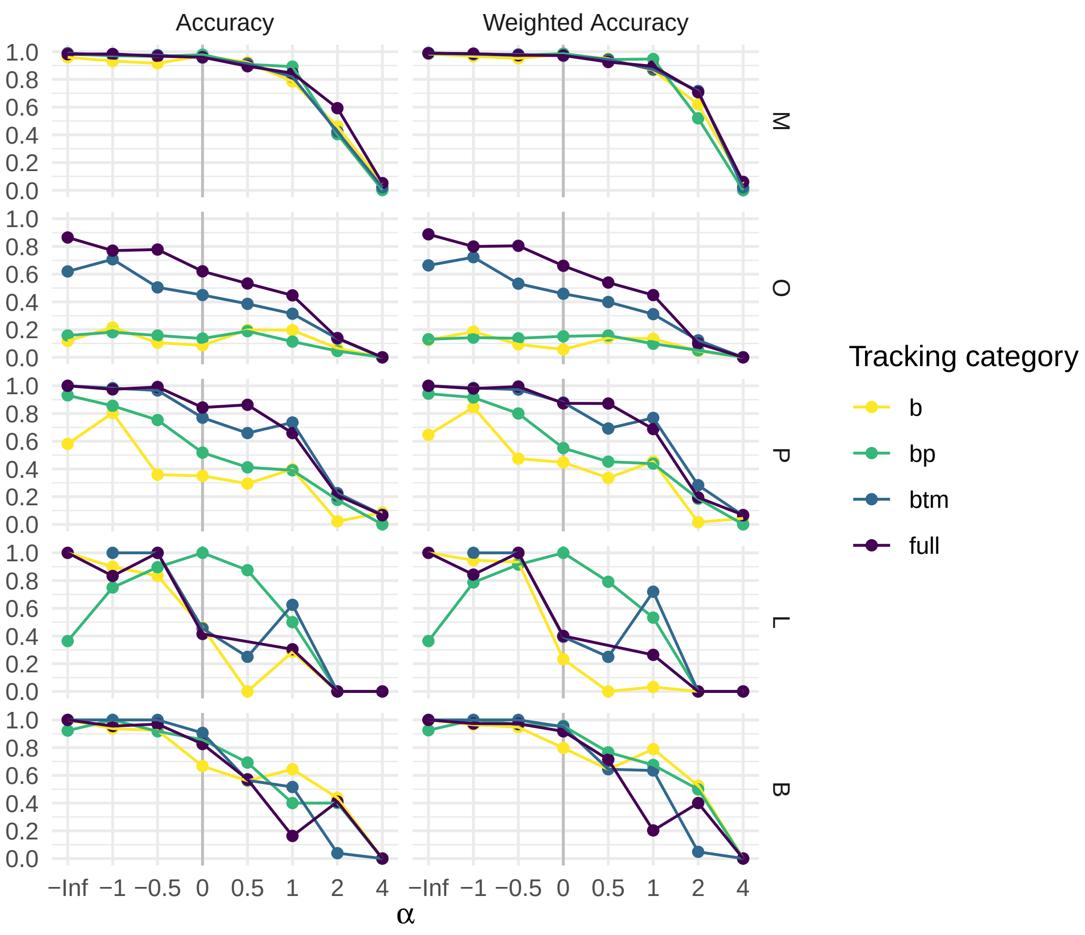


Figure 4.2: Per state accuracy on synthetic data.

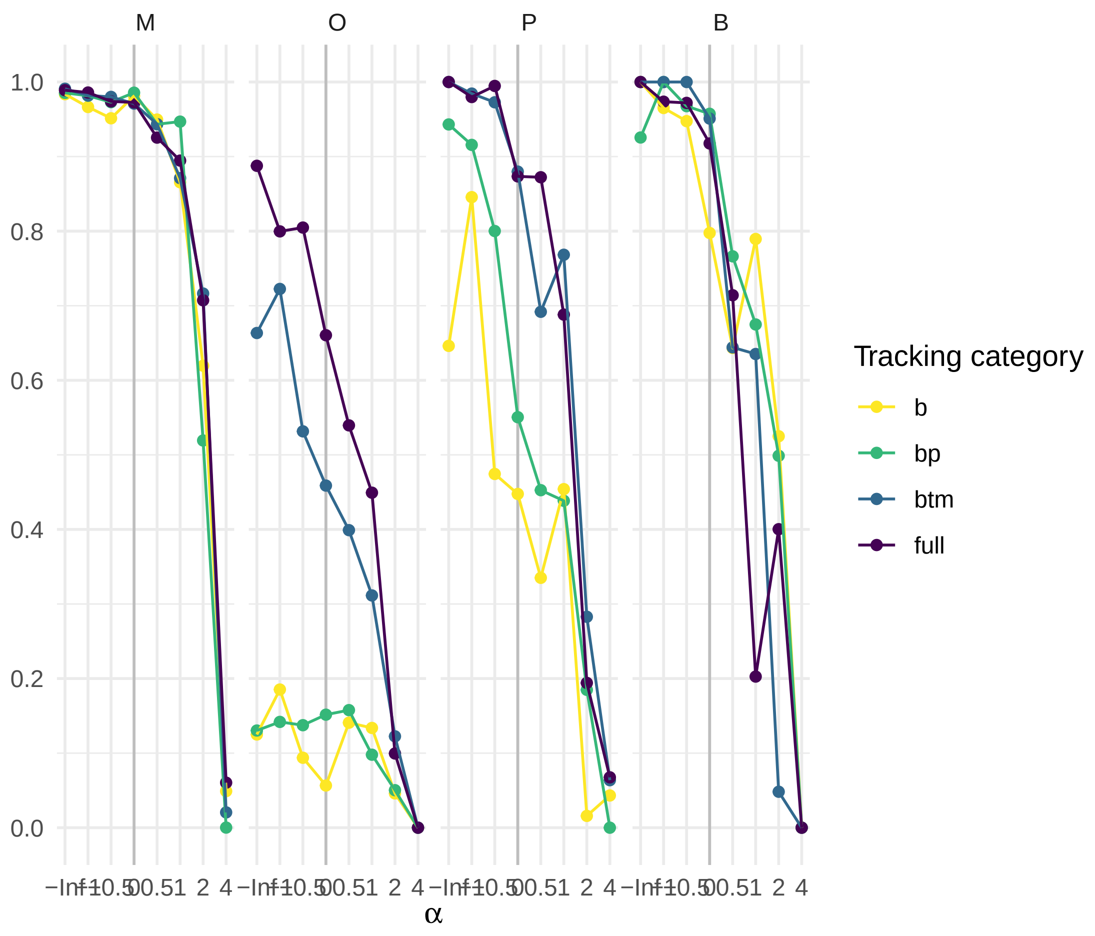


Figure 4.3: Per state accuracy on synthetic data.

Confusion matrices

```
confusion_matrix_df = XP %>%
  filter(!is.na(state)) %>%
  group_by(alpha, tracking_category, state_GT, state) %>%
  summarize(n = n(),
            wn = sum(prob),
            .groups = "drop") %>%
  group_by(alpha, 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) +
  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
```

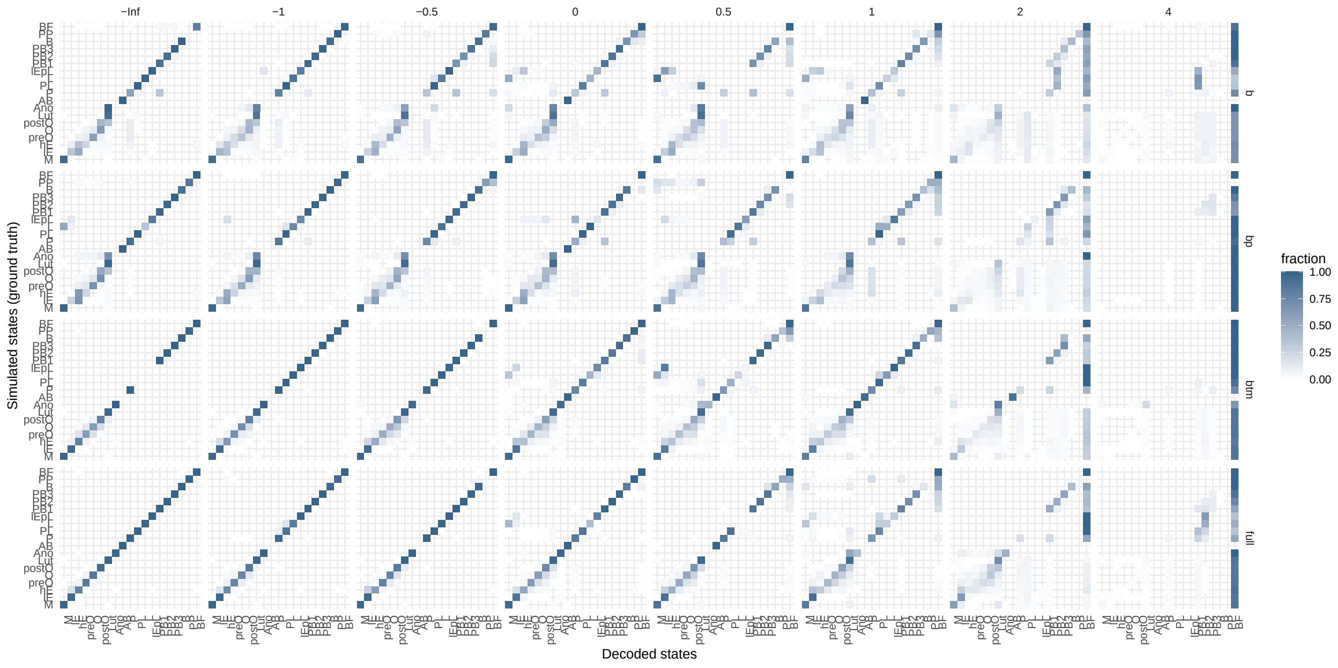


Figure 4.4: Confusion matrix for the decoding on synthetic data.

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

4.2 On Kindara data

Loading data and formatting results

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

```
ML = ML %>% rename(state_GT = state)

XP = full_join(X, RES, by = c("seq_id", "t")) %>%
  left_join(., ML, by = c("seq_id", "t") )

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

Overall accuracy

```
Accuracy = mean(XP$state == XP$state_GT, na.rm = TRUE)
Accuracy
```

```
## [1] 0.9301058
```

```
Weighted_Accuracy = weighted.mean(x = XP$state == XP$state_GT, w = XP$prob, na.rm = TRUE)
Weighted_Accuracy
```

```
## [1] 0.9451277
```

Confusion matrices

```
confusion_matrix_df = XP %>%
  filter(!is.na(state), !is.na(state_GT)) %>%
  group_by(state_GT, state) %>%
  summarize(n = n(),
            wn = sum(prob),
            .groups = "drop") %>%
  group_by(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_conf_mat = ggplot(confusion_matrix_df, aes(x = decoded_state_name, y = GT_state_name, fill = fraction)) +
  geom_tile() +
  facet_grid(. ~ type) +
  scale_fill_gradient("State-specific accuracy", 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))

g_conf_mat
```

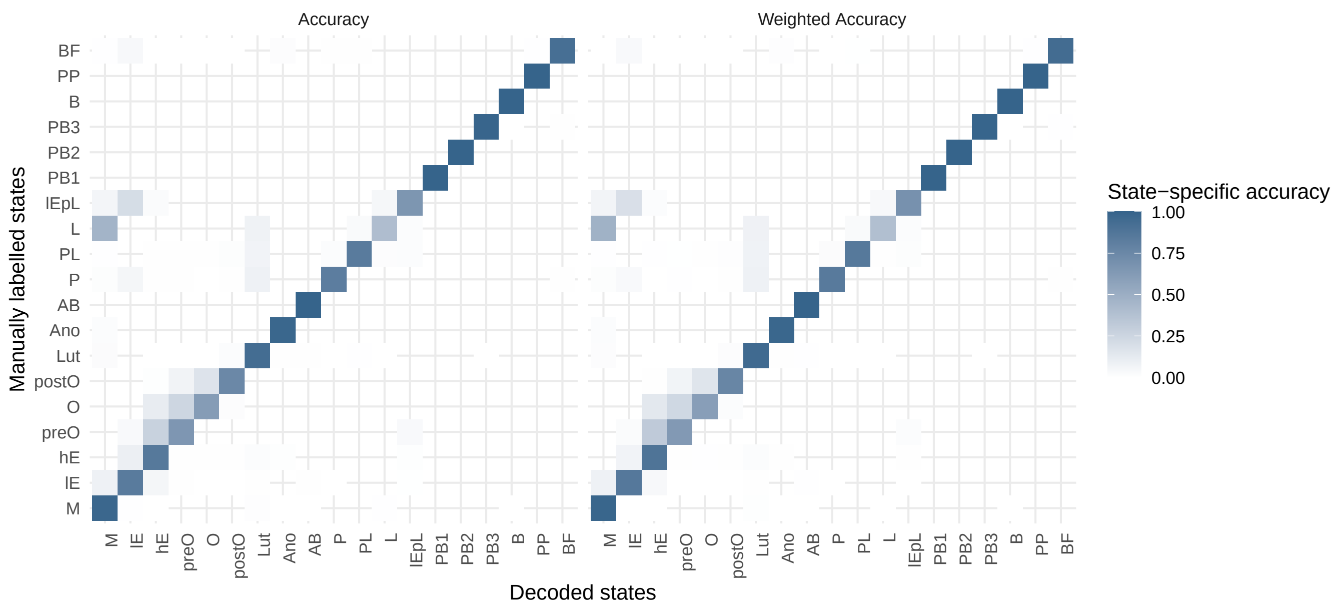
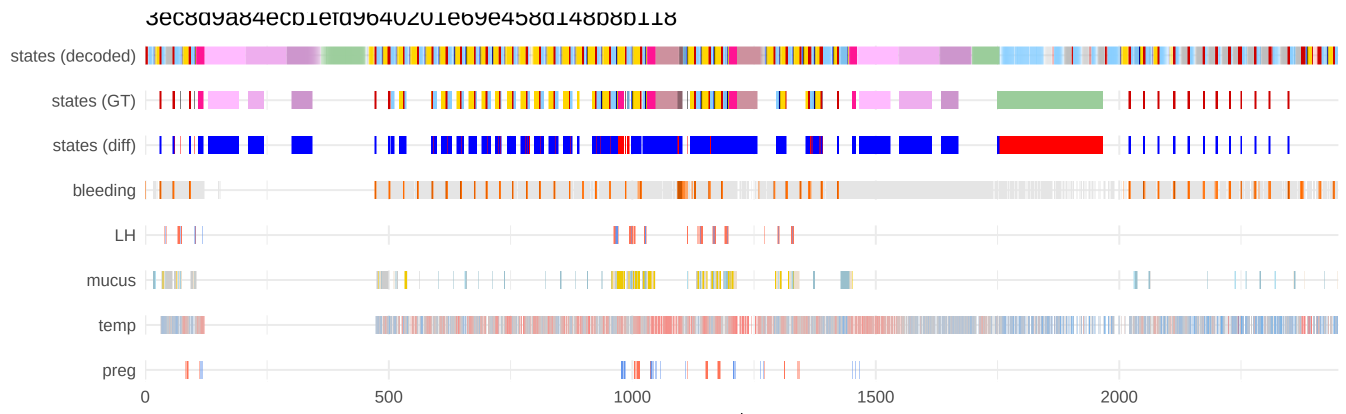


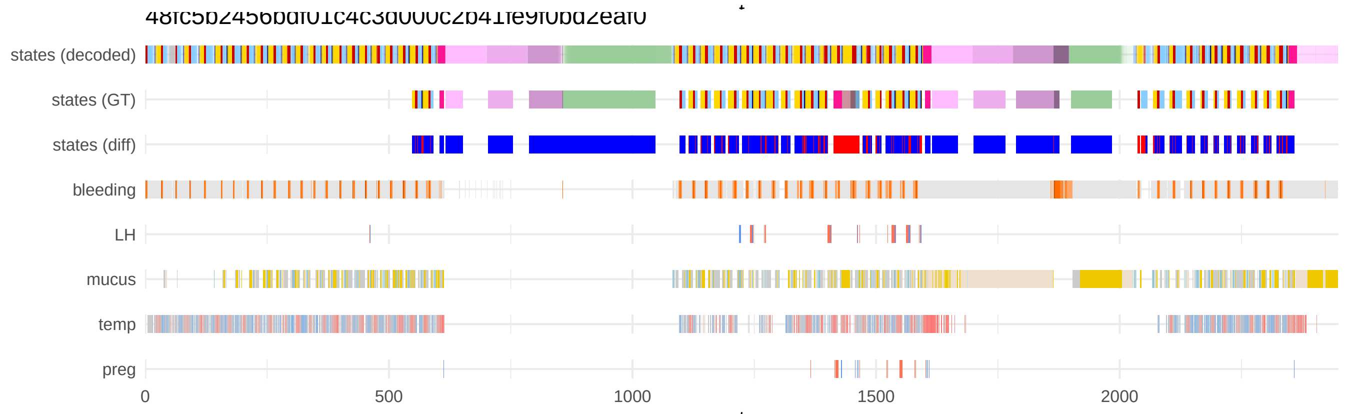
Figure 4.5: Confusion matrix between manually labelled and decoded states.

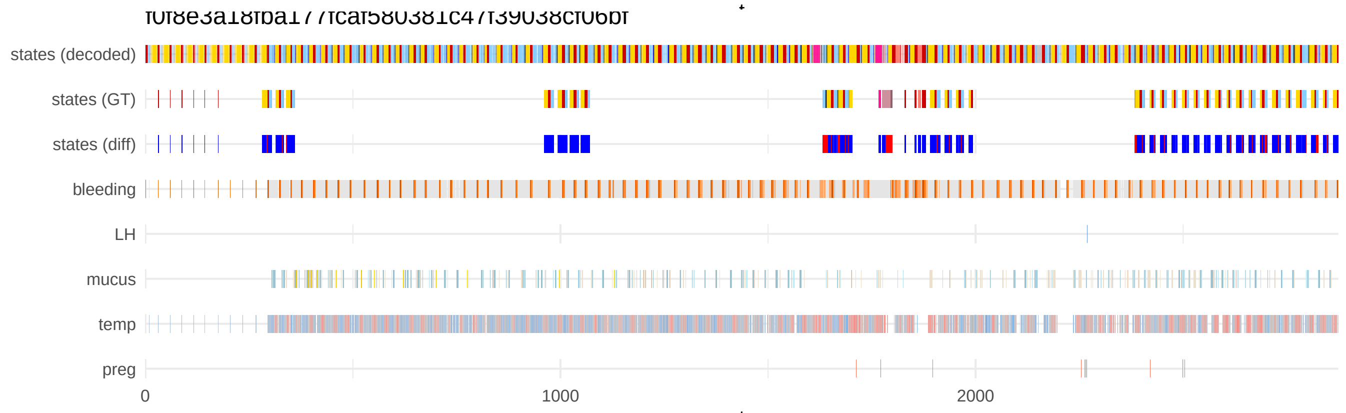
```r
save(g_conf_mat, file = "../Data/plots/g_conf_mat.Rdata")
```

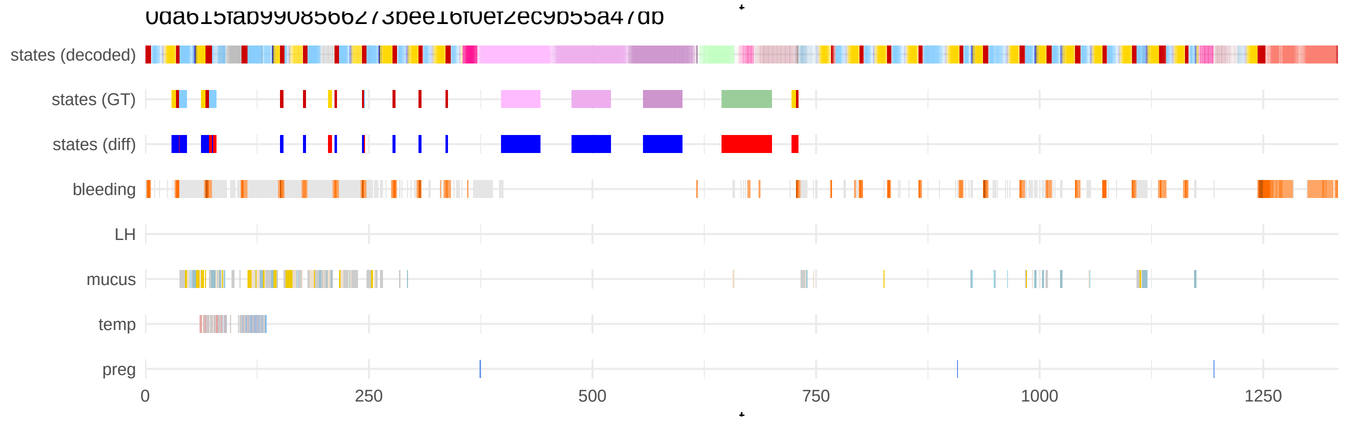
Decoded sequences

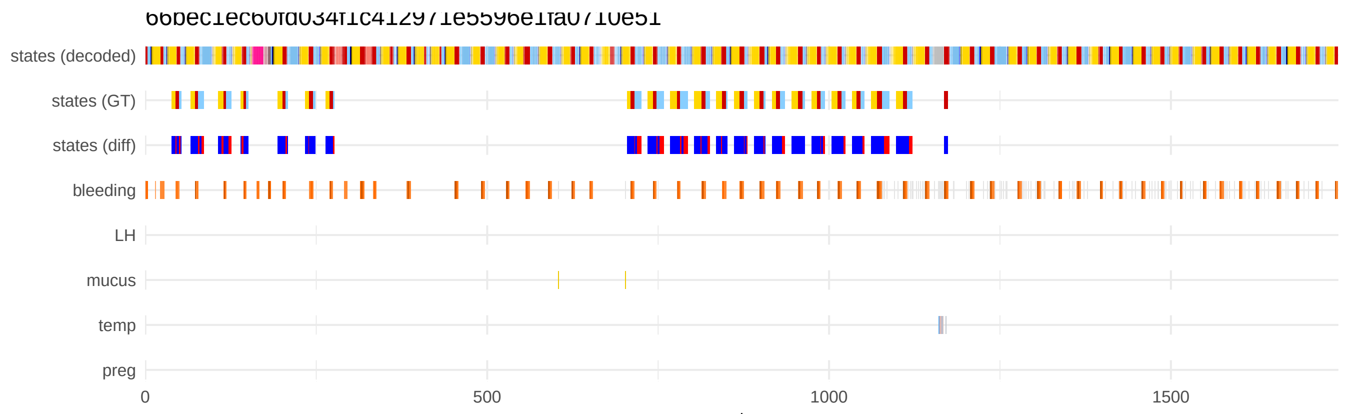
```r
ordered_sequences =
  XP %>%
  group_by(seq_id) %>%
  filter(seq_id %in% unique(RES$seq_id),
         !is.na(state_GT)) %>%
  summarize(
    n_wrong = sum(state != state_GT),
    accuracy = mean(state == state_GT),
    .groups = "drop"
  ) %>%
  arrange(-n_wrong)

for(i in 1:10){ # nrow(ordered_sequences) # length(unique(RES$seq_id))
  uid = ordered_sequences$seq_id[i]  # unique(RES$seq_id)[i]
  this_user_XP = XP %>% filter(seq_id == uid) %>%
    rename(state_decoded = state,
           state_prob_decoded = prob)

  plot_hsmm_seq(X = this_user_XP, model = R_hsmm, title = uid,
                compact_view = TRUE,
                add_color_legend_in_compact_view = FALSE) %>%
    print()

}
```

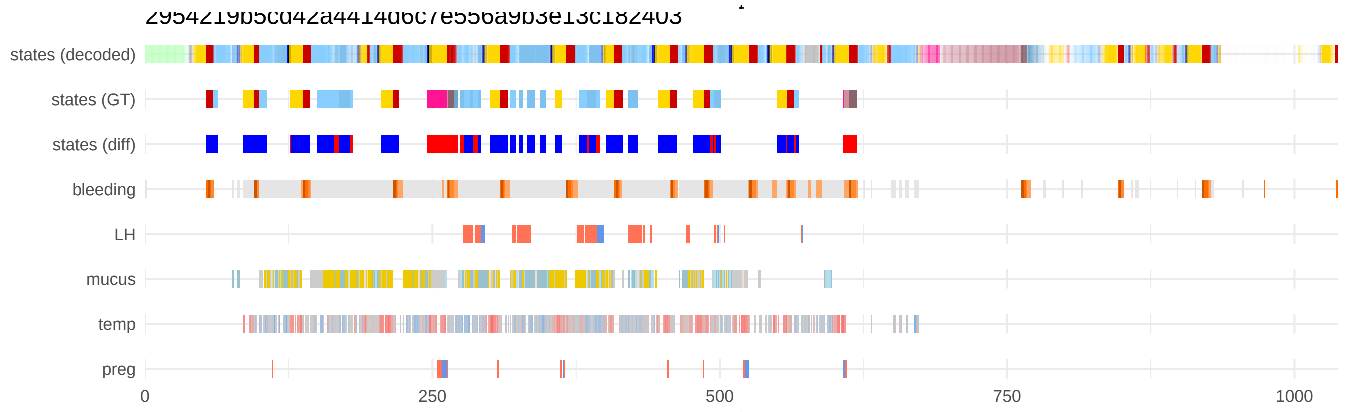

3ec8d9a84ecb1efd9640201e69e458d148b8b118
states (decoded)
states (GT)
states (diff)
bleeding
LH
mucus
temp
preg
0
500
1000
1500
2000

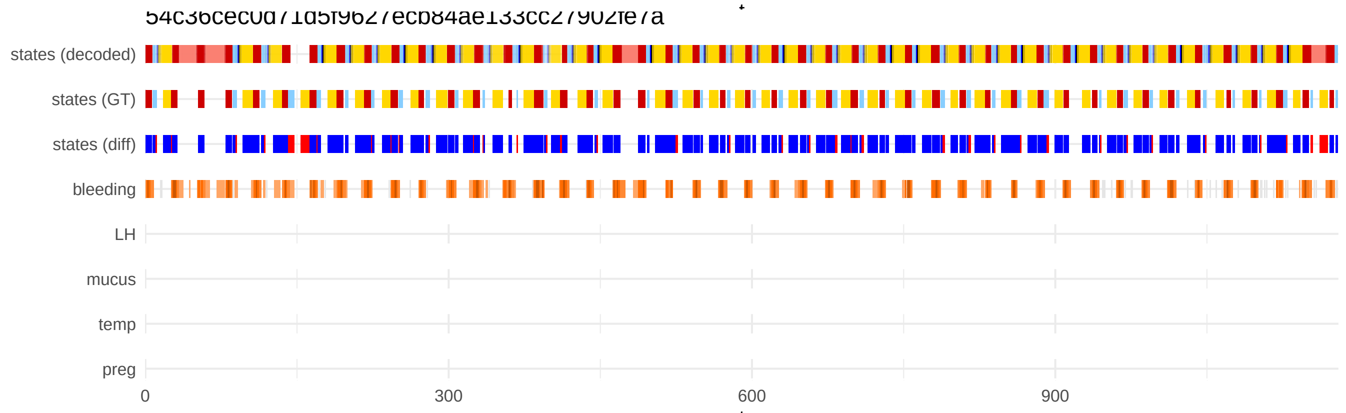

48fc5b2456bdf01c4c3d000c2b41fe9f0bd2eaf0
states (decoded)
states (GT)
states (diff)
bleeding
LH
mucus
temp
preg
0
500
1000
1500
2000

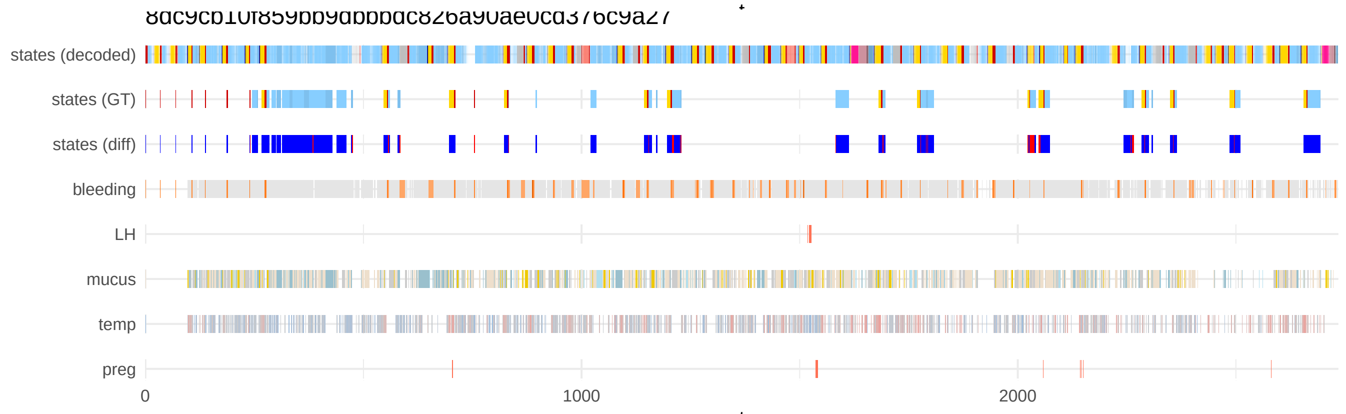

f0f8e3a18fba177fcaf580381c47f39038cf06bf
states (decoded)
states (GT)
states (diff)
bleeding
LH
mucus
temp
preg
0
1000
2000

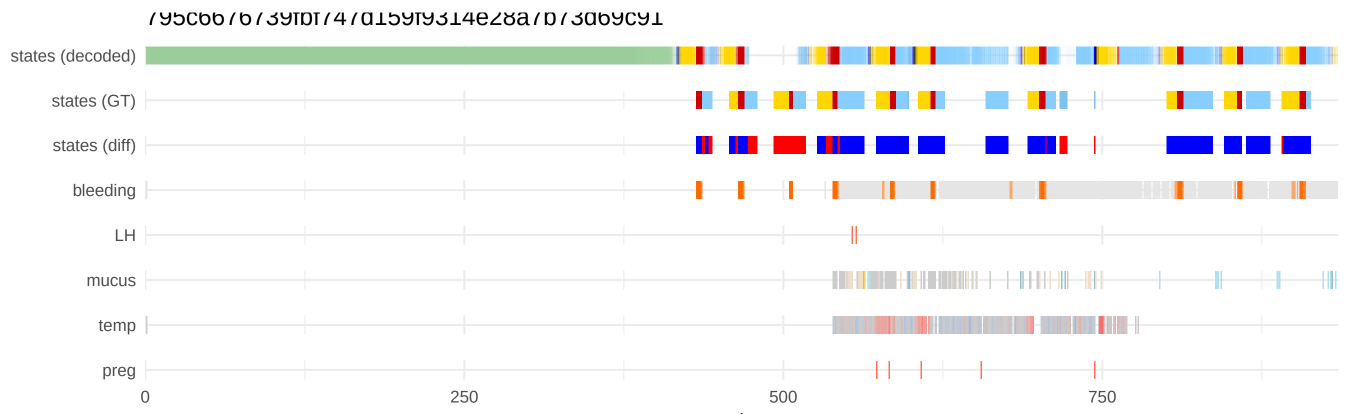

0da615fab9908566273bee16f0ef2ec9b55a47db
states (decoded)
states (GT)
states (diff)
bleeding
LH
mucus
temp
preg
0
250
500
750
1000
1250

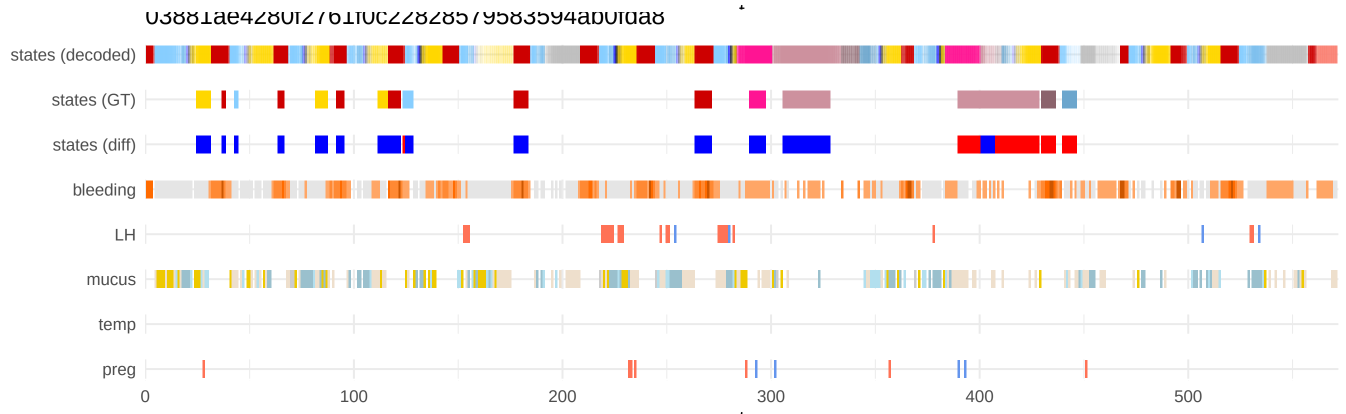

66bec1ec60fd034f1c412971e5596e1fa0710e51
states (decoded)
states (GT)
states (diff)
bleeding
LH
mucus
temp
preg
0
500
1000
1500


2954219b5cd42a4414d6c7e556a9b3e13c182403
states (decoded)
states (GT)
states (diff)
bleeding
LH
mucus
temp
preg
0
250
500
750
1000


54c36cec0d71d5f9627ecb84ae133cc27902fe7a
states (decoded)
states (GT)
states (diff)
bleeding
LH
mucus
temp
preg
0
300
600
900


8dc9cb10f859bb9dbbbdc826a90ae0cd376c9a27
states (decoded)
states (GT)
states (diff)
bleeding
LH
mucus
temp
preg
0
1000
2000


795c6676739fbf747d159f9314e28a7b73d69c91
states (decoded)
states (GT)
states (diff)
bleeding
LH
mucus
temp
preg
0
250
500
750


03881ae4280f2761f0c22828579583594ab0fda8
states (decoded)
states (GT)
states (diff)
bleeding
LH
mucus
temp
preg
0
100
200
300
400
500

5 Predicting next period

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

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

```r
RES.file = paste0(IO$output_data, "RES_app_data.feather")
RES = read_feather(path = RES.file)
rm(RES.file)
```

5.2 Selecting users and cycles

```r
RES = RES %>%
  arrange(seq_id, t) %>%
  group_by(seq_id) %>%
  mutate(cycle_start = (state == 1) & ((lag(state) != 1) | (t == 1)),
         cycle_end = (state != 1) & (lead(state) == 1),
         cycle_number = cumsum(cycle_start)) %>%
  group_by(seq_id, cycle_number) %>%
  mutate(cycle_length = n()) %>%
  ungroup()

cycles = RES %>% filter(cycle_start) %>%
  select(-state, -cycle_start, -cycle_end)

users = cycles %>%
  group_by(seq_id) %>%
  summarize(n_cycles = n(),
            median_cycle_length_all = median(cycle_length, na.rm = TRUE),
            median_cycle_length = median(cycle_length[cycle_length < 50],
                                         na.rm = TRUE),
            sd_cycle_length_all = sd(cycle_length, na.rm = TRUE),
            sd_cycle_length = sd(cycle_length[cycle_length < 50], na.rm = TRUE),
            .groups = "drop")
```

```r
df = RES %>%
  filter(
    # filter for tracking behavior that's not just bleeding and/or pregnancy tests
    str_detect(tracking_behavior,
               str_c(c("b_tests","no_temp","no_mucus","full"), collapse = "|")),
    # filter for ovulatory cycles
    state <= 7,
    # filter for ovulatory cycles in which the state prob is not too low (i.e. there was enough data to estim
    prob > 0.3
  ) %>%
  # we look for consecutive cycles fulfilling the criteria
  group_by(seq_id) %>%
  mutate(new_cons_stretch = (t == min(t)) | ((t - lag(t)) > 1),
         cons_stretch_id = cumsum(new_cons_stretch)) %>%
  # we make sure that cons_stretches start on a cycle start and end on a cycle end
  group_by(seq_id, cons_stretch_id) %>%
  mutate(n_cycles = sum(cycle_start)) %>%
  filter(n_cycles >= 5) %>%
  mutate(t_first_cycle_start = min(t[cycle_start]),
         t_last_cycle_end = max(t[cycle_end])) %>%
  ungroup() %>%
  filter(t >= t_first_cycle_start,
         t <= t_last_cycle_end) %>%
  # we ensure that we have enough cycles in a stretch and that the cycle lengths show some variability
  group_by(seq_id, cons_stretch_id) %>%
  mutate(n_cycles = length(unique(cycle_number)),
         cycle_length_diff = max(cycle_length) - min(cycle_length)) %>%
  filter(
    # filter for at least 5 consecutive cycles
    n_cycles >= 5,
    # filter for some irregularity in cycle length in this stretch
    cycle_length_diff >= 5) %>%
  ungroup()


df %>% select(seq_id, cons_stretch_id, cycle_number) %>% distinct() %>% nrow()
```

```
## [1] 620
```

```r
df %>% select(seq_id, cons_stretch_id) %>% distinct() %>% nrow()
```

```
## [1] 49
```

```r
df %>% select(seq_id) %>% distinct() %>% nrow()
```

```
## [1] 24
```

5.3 Predicting cycle length

5.3.1 Building the input dataset

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

```r
# for each user
# for each consecutive stretch
# for each consecutive 5-cycles group,
# we keep the 4 first cycles (= training cycles)

input = data.frame()
input_baseline = data.frame()
output = data.frame()
fifth_cycle = data.frame()

for(u in unique(df$seq_id)){
  cat(u, "\t")
  dfu = df %>% filter(seq_id == u)
  for(s in unique(dfu$cons_stretch_id)){
    dfs = dfu %>% filter(cons_stretch_id == s)
    n_cycles = length(unique(dfs$cycle_number))
    first_cycle = min(dfs$cycle_number)
    for(cn in first_cycle:(first_cycle+n_cycles-5)){
      dfc = dfu %>% filter(cycle_number %in% cn:(cn+3))
      this_input = X %>%
        filter(seq_id == u,
               t >= min(dfc$t),
               t <= max(dfc$t)+5) %>%  # we add 5 days so that the next period is included
        rename(o_seq_id = seq_id) %>%
        mutate(seq_id = str_c(o_seq_id, "_",s,"_",cn))

      input = bind_rows(input, this_input)

      this_input_baseline = dfc %>%
        select(seq_id, t, cycle_start, cycle_number, cycle_length) %>%
        filter(cycle_start) %>%
        rename(o_seq_id = seq_id) %>%
        mutate(seq_id = str_c(o_seq_id, "_",s,"_",cn))

      input_baseline = bind_rows(input_baseline, this_input_baseline)

      dfc5 = dfu %>% filter(cycle_number %in% (cn+4))
      this_fifth_cycle = X %>%
        filter(seq_id == u,
               t >= min(dfc5$t),
               t <= max(dfc5$t)) %>%
        rename(o_seq_id = seq_id) %>%
        mutate(seq_id = str_c(o_seq_id, "_",s,"_",cn))

      fifth_cycle = bind_rows(fifth_cycle, this_fifth_cycle)

      this_output = dfu %>%  filter(cycle_number == cn+4) %>%
        select(seq_id, t, cycle_number, cycle_length) %>%
        group_by(seq_id, cycle_number, cycle_length) %>%
        summarize(t_cycle_end = max(t),
                  t_cycle_start = min(t),
                  .groups = "drop") %>%
        rename(o_seq_id = seq_id) %>%
        mutate(seq_id = str_c(o_seq_id, "_",s,"_",cn))
      output = bind_rows(output, this_output)
    }
  }
}
```

```
## 003aebf48b9b97201d27a2bd890f3afa08027455    008a7d168d2ae8ff5fafc410f31cedbff386dcba   01f5b1c00959364daf86
```

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

5.3.2 Training and predictions

```
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]]$params$
  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 = rep(0, J),
                         q = matrix(0.5, nrow = length(marg_em_probs), ncol = 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.
```

```
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]
      this_output = output %>% 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.
      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)

      # HSMM
      # Train the model on the 4 cycles
      trained_model = fit_hsmm(model = P_hsmm, X = input %>% filter(seq_id == sid),
                               use_sojourn_prior = FALSE)
      # dec = predict_states_hsmm(model = trained_model$model,
      #                           X = input %>% filter(seq_id == sid),
      #                           method = "viterbi")
      # Predict from each day of the next cycle
      this_fifth_cycle = fifth_cycle %>% filter(seq_id == sid)
      t_cycle_start = min(this_fifth_cycle$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$state_seq %>%
            group_by(state) %>%
            summarize(t_last_transition = min(t), .groups = "drop") %>%
            arrange(-t_last_transition) %>%
            slice_head()
          # we simulate X times and we compute the mean and sd of the predicted cycle lengths
          X = 100
          hsmm_pred_this_cd = purrr::map_dfr(
            .x = last_state_transition$state:7,
            .f = function(s){
              data.frame(state = s, i = 1:X,
                         sojourn = sample(1:length(trained_model$model$sojourn[[s]]$d),
                                          X, prob = trained_model$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$t_last_transition
            summarize(uncertainty = sd(cycle_length),
                      cycle_length = mean(cycle_length)) %>%
            mutate(cycleday = cd, diff = cycle_length - this_output$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.
```

5.4 Results

```
# ggplot(pred %>% filter(model == "HSMM", cycleday_backward >= -25), aes(x = cycleday_backward, y = diff, col
#   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 58 rows containing missing values (geom_tile).
```

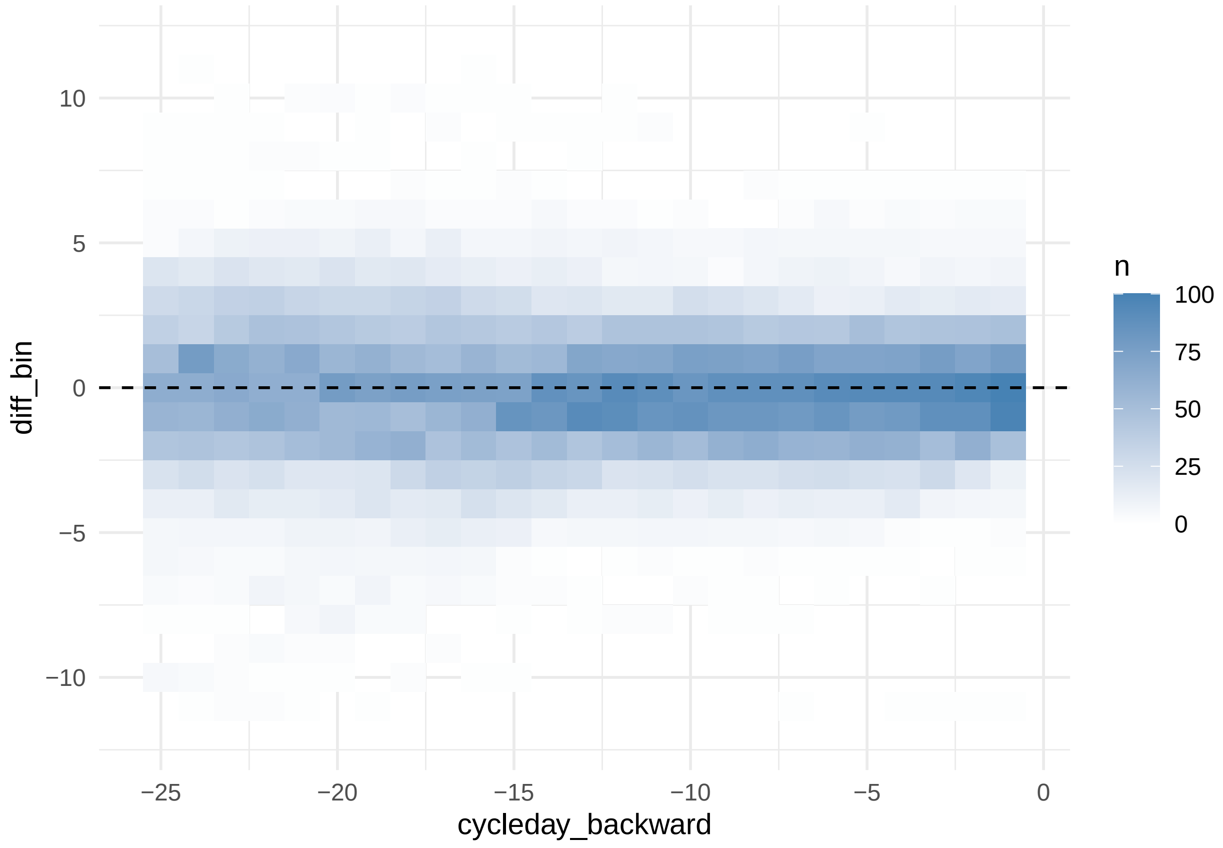


Figure 5.1: Distribution of the errors on the cycle length prediction from the HSMM simulations as one progresses through the cycles.

```
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	17.56	16.99
After ovulation (10 days before next cycle)	17.56	6.10

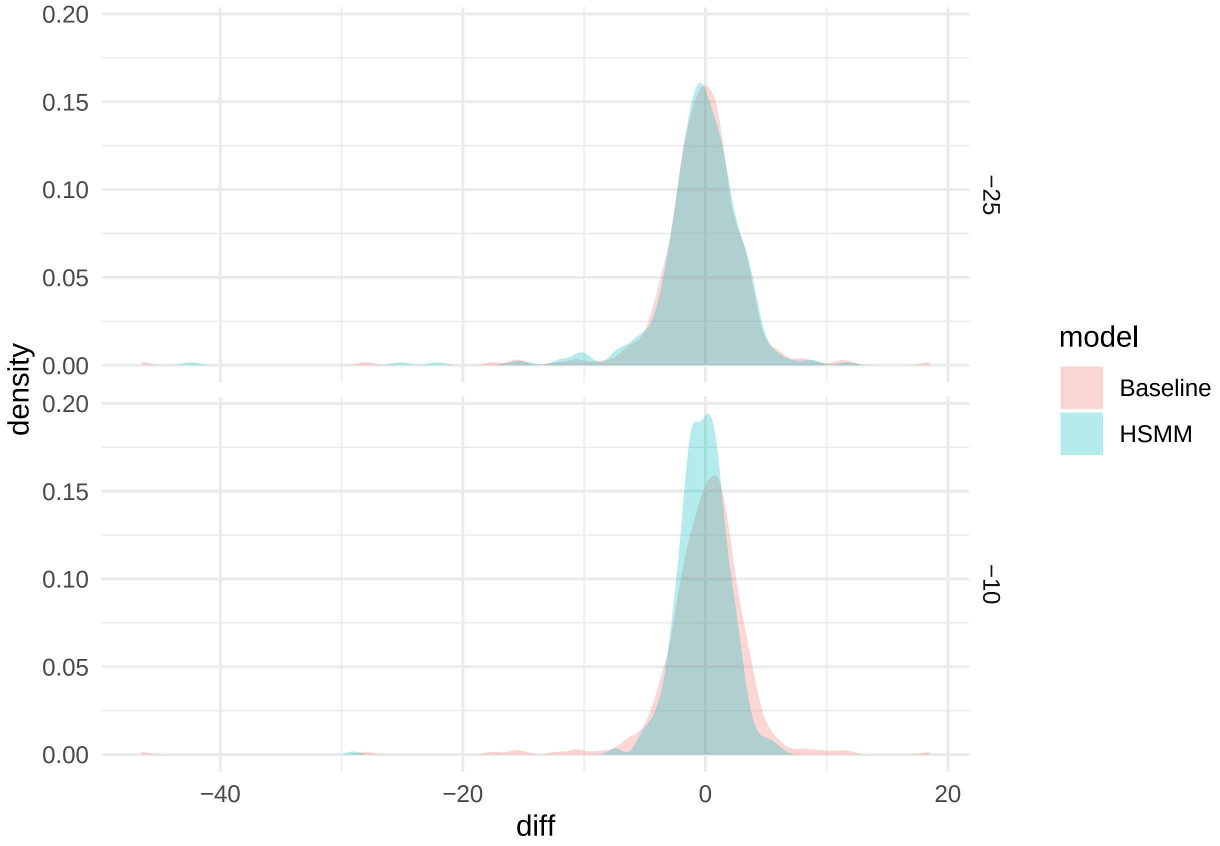


Figure 5.2: Comparison of the distributions of the error on the cycle length prediction between the HSMM and the baseline method.

```
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
```

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)
```

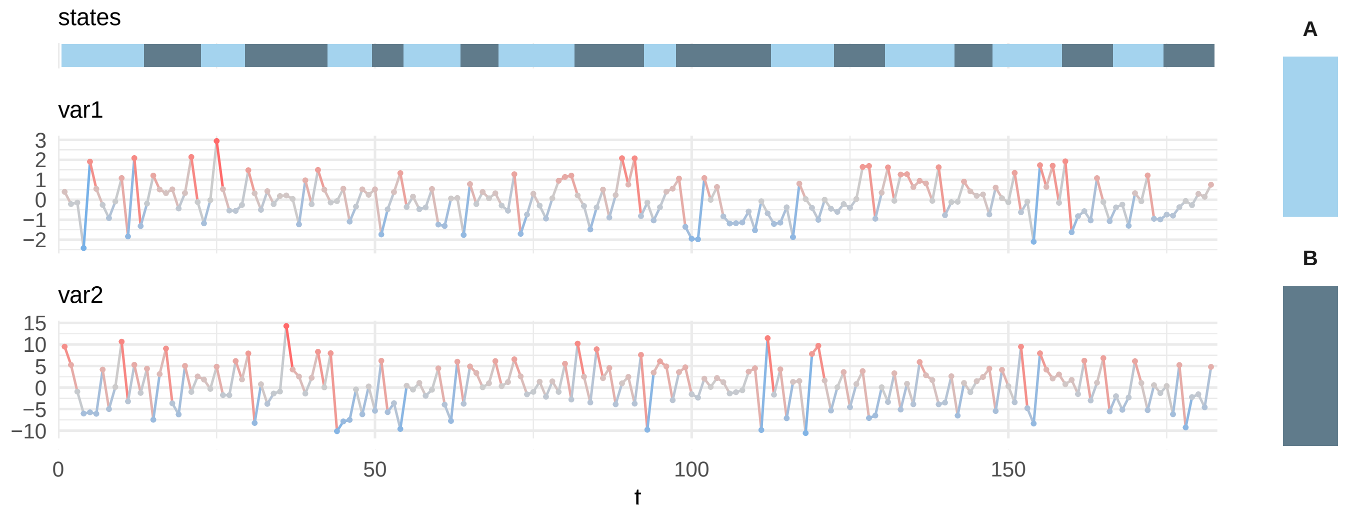


Figure 6.1: Simulated data with the specified model.

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)
```

```r
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
```

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

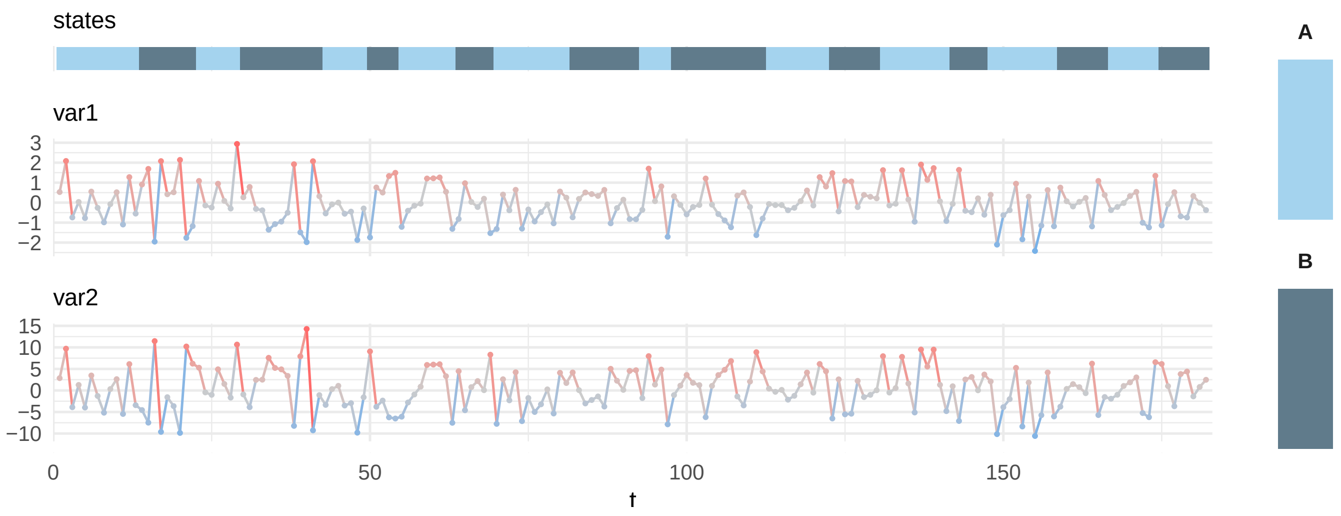


Figure 6.2: Modified data to introduce within-state correlations.

```r
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)
```

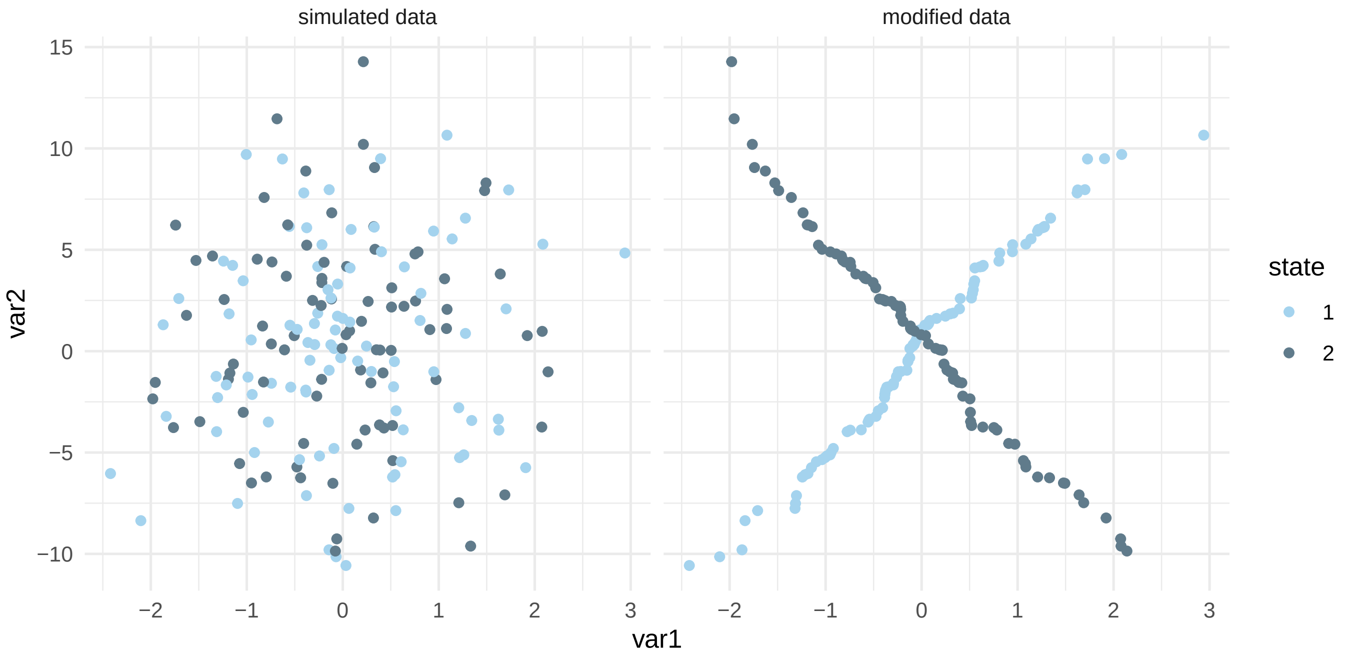


Figure 6.3: Scatter-plot between the two variables of the original and modified time-series.

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)
```

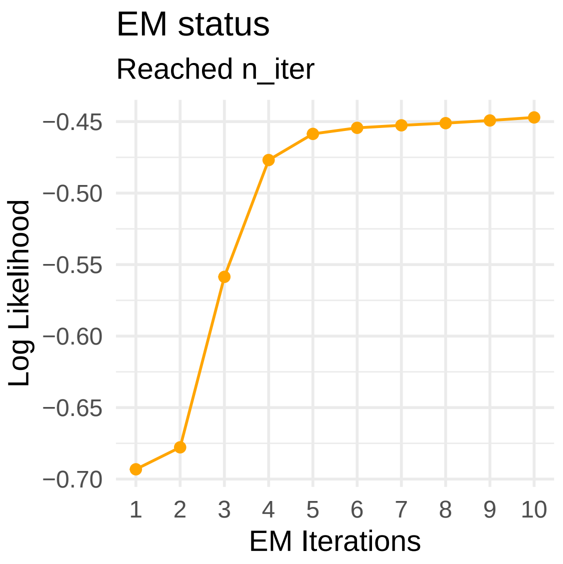


Figure 6.4: Fitting the specified model to the modified data: likelihood at each EM iteration.

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: 0.532967

Decoding accuracy on the modified data with the fitted model: 0.9395604

```
b_original = m$b %>% filter(!is.na(var1), !is.na(var2)) %>% pivot_longer(col = starts_with("p_"), names_to =
b_fit = model_fit$model$b %>% filter(!is.na(var1), !is.na(var2))%>% pivot_longer(col = starts_with("p_"), nam

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)
```

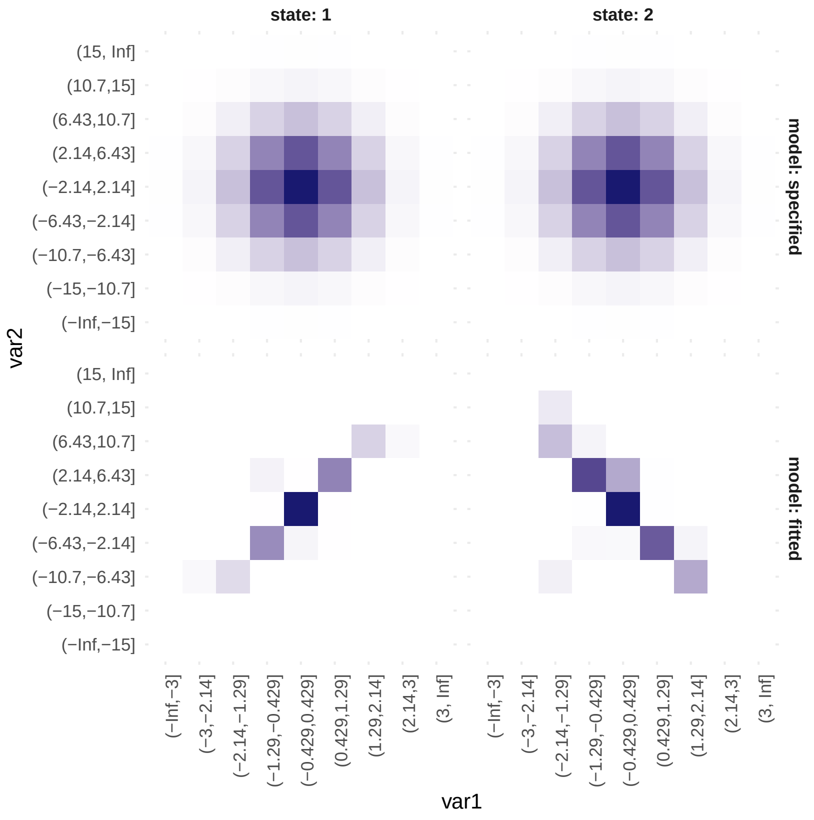


Figure 6.5: Joint emission probability densities per state and model (normalized densities).

7 Reproducibility receipt

```
## Execution datetime:
```

```
## [1] "2020-12-01 15:47:49 PST"
```

```
## -------------------------------
```

```
## sessionInfo :
```

```
## R version 4.0.2 (2020-06-22)
## Platform: x86_64-apple-darwin17.0 (64-bit)
## Running under: macOS  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_0.5.1          viridisLite_0.3.0      HiddenSemiMarkov_0.1.0
##  [4] sn_1.6-2               magrittr_2.0.1         ggthemes_4.2.0
##  [7] forcats_0.5.0          stringr_1.4.0          dplyr_1.0.2
## [10] purrr_0.3.4            readr_1.4.0            tidyr_1.1.2
## [13] tibble_3.0.4           tidyverse_1.3.0        feather_0.3.5
## [16] kableExtra_1.3.1       styler_1.3.2           bookdown_0.21
## [19] knitr_1.30             shiny_1.5.0            igraph_1.2.6
## [22] tictoc_1.0             ggpubr_0.4.0           ggplot2_3.3.2
## [25] cowplot_1.1.0          data.table_1.13.2
##
## loaded via a namespace (and not attached):
##  [1] fs_1.5.0             lubridate_1.7.9.2    webshot_0.5.2
##  [4] httr_1.4.2           numDeriv_2016.8-1.1  tools_4.0.2
##  [7] backports_1.2.0      R6_2.5.0             lazyeval_0.2.2
## [10] DBI_1.1.0            colorspace_2.0-0     withr_2.3.0
## [13] gridExtra_2.3        tidyselect_1.1.0     mnormt_2.0.2
## [16] curl_4.3             compiler_4.0.2       cli_2.2.0
## [19] rvest_0.3.6          network_1.16.1       xml2_1.3.2
## [22] plotly_4.9.2.1       labeling_0.4.2       scales_1.1.1
## [25] digest_0.6.27        foreign_0.8-80       rmarkdown_2.5
## [28] rio_0.5.16           pkgconfig_2.0.3      htmltools_0.5.0
## [31] highr_0.8            dbplyr_2.0.0         fastmap_1.0.1
## [34] htmlwidgets_1.5.2    rlang_0.4.9          readxl_1.3.1
## [37] rstudioapi_0.13      farver_2.0.3         generics_0.1.0
## [40] jsonlite_1.7.1       statnet.common_4.4.1 zip_2.1.1
## [43] car_3.0-10           Rcpp_1.0.5           munsell_0.5.0
## [46] fansi_0.4.1          abind_1.4-5          lifecycle_0.2.0
## [49] geomnet_0.3.1        stringi_1.5.3        yaml_2.2.1
## [52] carData_3.0-4        grid_4.0.2           promises_1.1.1
## [55] crayon_1.3.4         lattice_0.20-41      haven_2.3.1
## [58] hms_0.5.3            sna_2.6              tmvnsim_1.0-2
## [61] pillar_1.4.7         ggsignif_0.6.0       codetools_0.2-18
## [64] rle_0.9.2            reprex_0.3.0         glue_1.4.2
## [67] evaluate_0.14        modelr_0.1.8         vctrs_0.3.5
## [70] httpuv_1.5.4        cellranger_1.1.0     gtable_0.3.0
## [73] assertthat_0.2.1    xfun_0.19            openxlsx_4.2.3
## [76] mime_0.9            xtable_1.8-4         broom_0.7.2
## [79] coda_0.19-4         rstatix_0.6.0        later_1.1.0.1
## [82] ellipsis_0.3.1
```

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).